Clustering by Compression

Rudi Cilibrasi*
CWI

Paul Vitanyi†
CWI, University of Amsterdam
and National ICT of Australia

Abstract

We present a new method for clustering based on compression. The method doesn't use subject-specific features or background knowledge, and works as follows: First, we determine a parameter-free, universal, similarity distance, the normalized compression distance or NCD , computed from the lengths of compressed data files (singly and in pairwise concatenation). Second, we apply a hierarchical clustering method. The NCD is not restricted to a specific application area, and works across application area boundaries. A theoretical precursor, the normalized information distance, co-developed by one of the authors, is provably optimal. However, the optimality comes at the price of using the non-computable notion of Kolmogorov complexity. We propose axioms to capture the real-world setting, and show that the NCD approximates optimality. To extract a hierarchy of clusters from the distance matrix, we determine a dendrogram (binary tree) by a new quartet method and a fast heuristic to implement it. The method is implemented and available as public software, and is robust under choice of different compressors. To substantiate our claims of universality and robustness, we report evidence of successful application in areas as diverse as genomics, virology, languages, literature, music, handwritten digits, astronomy, and combinations of objects from completely different domains, using statistical, dictionary, and block sorting compressors. In genomics we presented new evidence for major questions in Mammalian evolution, based on whole-mitochondrial genomic analysis: the Eutherian orders and the Marsupionta hypothesis against the Theria hypothesis.

1 Introduction

All data are created equal but some data are more alike than others. We propose a method expressing this alikeness, using a new similarity metric based on compression. It is parameter-free in that it doesn't use any features or background knowledge, and can without changes be applied to different areas and across area boundaries. It is universal in that it approximates the parameter expressing similarity of the dominant feature in all pairwise comparisons. It is robust in the sense that its success appears independent from the type of compressor used. The clustering we use is hierarchical clustering in dendrograms based on a new fast heuristic for the quartet method. The method is available as an open-source software tool. Below we explain the method, the theory underpinning it, and present evidence for its universality and robustness by experiments and results in a plethora of different areas using different types of compressors.

Feature-Based Similarities: We are presented with unknown data and the question is to determine the similarities among them and group like with like together. Commonly, the data are of a certain type: music fi les, transaction records of ATM machines, credit card applications, genomic data. In these data there are hidden relations that we would like to get out in the open. For example, from genomic data one can extract letter- or block frequencies (the blocks are over the four-letter alphabet); from music fi les one can extract various specifi c numerical features, related to pitch, rhythm, harmony etc. One can extract such features using for instance Fourier transforms [43] or wavelet transforms [17], to quantify parameters expressing similarity. The resulting vectors corresponding to the various fi les are then classifi ed or clustered using existing classifi cation software, based on various standard statistical pattern recognition classifi ers [43], Bayesian classifi ers [15], hidden Markov models [13], ensembles of nearest-neighbor classifi ers [17] or neural networks [15, 39]. For example, in music one feature would be to look for rhythm in the sense of beats per minute. One can make a histogram where each histogram bin corresponds to a particular tempo

*Supported in part by the Netherlands BSIK/BRICKS project. Address: CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands. Email: Rudi.Cilibrasi@cwi.nl.

†Part of this work was done while the author was on leave at the National ICT of Australia, Sydney Laboratory at UNSW. Supported in part by the EU project RESQ, IST–2001–37559, the NoE QUIPROCONE IST–1999–29064, the ESF QiT Programmme, and the EU NoE PASCAL, and the Netherlands BSIK/BRICKS project. Address: CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands. Email: Paul.Vitanyi@cwi.nl.

in beats-per-minute and the associated peak shows how frequent and strong that particular periodicity was over the entire piece. In [43] we see a gradual change from a few high peaks to many low and spread-out ones going from hip-hip, rock, jazz, to classical. One can use this similarity type to try to cluster pieces in these categories. However, such a method requires specific and detailed knowledge of the problem area, since one needs to know what features to look for.

Non-Feature Similarities: Our aim is to capture, in a single similarity metric, *every effective distance*: effective versions of Hamming distance, Euclidean distance, edit distances, alignment distance, Lempel-Ziv distance [11], and so on. This metric should be so general that it works in every domain: music, text, literature, programs, genomes, executables, natural language determination, equally and simultaneously. It would be able to simultaneously detect *all* similarities between pieces that other effective distances can detect.

Compression-based Similarity: Such a "universal" metric was co-developed by us in [29, 30, 31], as a normalized version of the "information metric" of [32, 4]. Roughly speaking, two objects are deemed close if we can significantly "compress" one given the information in the other, the idea being that if two pieces are more similar, then we can more succinctly describe one given the other. The mathematics used is based on Kolmogorov complexity theory [32]. In [31] we defined a new class of (possibly non-metric) distances, taking values in $[0,1]$ and appropriate for measuring effective similarity relations between sequences, say one type of similarity per distance, and *vice versa*. It was shown that an appropriately "normalized" information distance minorizes every distance in the class. It discovers all effective similarities in the sense that if two objects are close according to some effective similarity, then they are also close according to the normalized information distance. Put differently, the normalized information distance represents similarity according to the dominating shared feature between the two objects being compared. In comparisons of more than two objects, different pairs may have different dominating features. The normalized information distance too is a metric and takes values in $[0,1]$; hence it may be called *"the" similarity metric*. To apply this ideal precise mathematical theory in real life, we have to replace the use of the noncomputable Kolmogorov complexity by an approximation using a standard real-world compressor. Earlier approaches resulted in the first completely automatic construction of the phylogeny tree based on whole mitochondrial genomes, [29, 30, 31], a completely automatic construction of a language tree for over 50 Euro-Asian languages [31], detects plagiarism in student programming assignments [8], gives phylogeny of chain letters [5], and clusters music [10]. Moreover, the method turns out to be robust under change of the underlying compressor-types: statistical (PPMZ), Lempel-Ziv based dictionary (gzip), block based (bzip2), or special purpose (Gencompress).

Related Work: In view of the simplicity and naturalness of our proposal, it is perhaps surprising that compression based clustering and classification approaches did not arise before. But recently there have been several partially independent proposals in that direction: [1, 2] for author attribution and building language trees—while citing [32, 4]—is by more *ad hoc* arguments related to the compressibility of a target file after first compressing a candidate reference file. The better the candidate file compresses, the more we feel it is similar to the reference file in question. This approach is used to cluster music MIDI files by Kohonen maps in [33]. Another recent offshoot based on our work is [23] hierarchical clustering based on mutual information. In a related, but considerably simpler feature-based approach, one can compare the word frequencies in text files to assess similarity. In [42] the word frequencies of words common to a pair of text files are used as entries in two vectors, and the similarity of the two files is based on the distance between those vectors. The authors attribute authorship to Shakespeare plays, the Federalist Papers, and the Chinese classic "The Dream of the Red Chamber." The approach to similarity distances based on block occurrence statistics is standard in genomics, and in an experiment below it gives inferior phylogeny trees compared to our compression method (and wrong ones according to current biological wisdom). The possibly new feature in the cited work is that it uses statistics of only the words that the files being compared have in common. A related, opposite, approach was taken in [22], where literary texts are clustered by author gender or fact versus fiction, essentially by first identifying distinguishing features, like gender dependent word usage, and then classifying according to those features. Recently, [20] has experimented with our method on all time sequence data used in all the major data-mining conferences in the last decade. Comparing the compression method with all major methods used in those conferences they established clear superiority of the compression method for clustering heterogenous data, and for anomaly detection.

Outline: Here we propose a first comprehensive theory of real-world compressor-based normalized compression distance, a novel hierarchical clustering heuristic, together with several applications. First, we propose mathematical notions of "admissible distance" (a wider class than in [31]), "normalized admissible distance" or "similarity distance," "normal compressor," and "normalized compression distance." We then prove the normalized compression distance based on a normal compressor to be a similarity metric. The normalized compression distance is shown to be quasi-universal in the sense that it minorizes every computable similarity distance up to an error term that depends on the quality of the compressor's approximation of the true Kolmogorov complexities of the files concerned. This

means that the NCD captures the dominant similarity over all possible features for every pair objects compared, up to the stated precision. Note that different pairs of objects may have different dominant shared features. Next, we present a method of hierarchical clustering based on a novel fast randomized hill-climbing heuristic of a new quartet tree optimization criterion. Given a matrix of the pairwise similarity distances between the objects, we score how well the resulting tree represents the information in the distance matrix on a scale of 0 to 1. Then, as proof of principle, we run the program on three data sets, where we know what the final answer should be: (i) reconstruct a tree from a distance matrix obtained from a randomly generated tree; (ii) reconstruct a tree from files containing artificial similarities; and (iii) reconstruct a tree from natural files of vastly different types. To substantiate our claim of parameter-freeness and universality, we apply the method to different areas, not using any feature analysis at all. We first give an example in whole-genome phylogeny using the whole mitochondrial DNA of the species concerned. We compare the hierarchical clustering of our method with a more standard method of two-dimensional clustering (to show that our dendrogram method of depicting the clusters is more informative). We give a whole-genome phylogeny of fungi and compare this to results using alignment of selected proteins (alignment being often too costly to perform on the whole-mitochondial genome, but the disadvantage of protein selection being that different selections usually result in different phylogenies—so which is right?). We identify the virii that are closest to the sequenced SARS virus; we give an example of clustering of language families; Russian authors in the original Russian, the same pieces in English translation (clustering partially follows the translators); clustering of music in MIDI format; clustering of handwritten digits used for optical character recognition; and clustering of radio observations of a mysterious astronomical object, a microquasar of extremely complex variability. In all these cases the method performs very well in the following sense: The method yields the phylogeny of 24 species precisely according to biological wisdom. The probability that it randomly would hit this one outcome, or anything reasonably close, is very small. In clustering 36 music pieces taken equally many from pop, jazz, classic, so that 12-12-12 is the grouping we understand is correct, we can identify convex clusters so that only six errors are made. (That is, if three items get dislodged then six items get misplaced.) The probability that this happens by chance is extremely small. The reason why we think the method does something remarkable is concisely put by Laplace [28]:

> "If we seek a cause wherever we perceive symmetry, it is not that we regard the symmetrical event as less possible than the others, but, since this event ought to be the effect of a regular cause or that of chance, the first of these suppositions is more probable than the second. On a table we see letters arranged in this order `C o n s t a n t i n o p l e`, and we judge that this arrangement is not the result of chance, not because it is less possible than others, for if this word were not employed in any language we would not suspect it came from any particular cause, but this word being in use among us, it is incomparably more probable that some person has thus arranged the aforesaid letters than that this arrangement is due to chance."

Materials and Methods: The data samples we used were obtained from standard data bases accessible on the world-wide web, generated by ourselves, or obtained from research groups in the field of investigation. We supply the details with each experiment. The method of processing the data was the same in all experiments. First, we preprocessed the data samples to bring them in appropriate format: the genomic material over the four-letter alphabet $\{A,T,G,C\}$ is recoded in a four-letter alphabet; the music MIDI files are stripped of identifying information such as composer and name of the music piece. Then, in all cases the data samples were completely automatically processed by our CompLearn Toolkit, rather than as is usual in phylogeny, by using an ecclectic set of software tools per experiment. Oblivious to the problem area concerned, simply using the distances according to the NCD below, the method described in this paper fully automatically classifies the objects concerned. The method has been released in the public domain as open-source software: The CompLearn Toolkit [9] is a suite of simple utilities that one can use to apply compression techniques to the process of discovering and learning patterns in completely different domains. In fact, this method is so general that it requires no background knowledge about any particular subject area. There are no domain-specific parameters to set, and only a handful of general settings.

The Complearn Toolkit using NCD and not, say, alignment, can cope with full genomes and other large data files and thus comes up with a single distance matrix. The clustering heuristic generates a tree with a certain confidence, called standardized benefit score or $S(T)$ value in the sequel. Generating trees from the same distance matrix many times resulted in the same tree or almost the same tree, for all distance matrices we used, even though the heuristic is randomized. The differences that arose are apparently due to early or late termination with different $S(T)$ values. This is a great difference with previous phylogeny methods, where because of computational limitations one uses only parts of the genome, or certain proteins that are viewed as significant [21]. These are run through a tree reconstruction method like neighbor joining [38], maximum likelihood, maximum evolution, maximum parsimony as in [21], or quartet hypercleaning [6], many times. The percentage-wise agreement on certain branches arising

are called "bootstrap values." Trees are depicted with the best bootstrap values on the branches that are viewed as supporting the theory tested. Different choices of proteins result in different best trees. One way to avoid this ambiguity is to use the full genome, [36, 31], leading to whole-genome phylogeny. With our method we do whole-genome phylogeny, and end up with a single overall best tree, not optimizing selected parts of it.

The quality of the results depends on (a) the NCD distance matrix, and (b) how well the hierarchical tree represents the information in the matrix. The quality of (b) is measured by the $S(T)$ value, and is given with each experiment. In general, the $S(T)$ value deteriorates for large sets. We believe this to be partially an artifact of a low-resolution NCD matrix due to limited compression power, and limited file size. The main reason, however, is the fact that with increasing size of a natural data set the projection of the information in the NCD matrix into a binary tree gets increasingly distorted. Another aspect limiting the quality of the NCD matrix is more subtle. Recall that the method knows nothing about any of the areas we apply it to. It determines the dominant feature as seen through the NCD filter. The dominant feature of alikeness between two files may not correspond to our a priori conception but may have an unexpected cause. The results of our experiments suggest that this is not often the case: In the natural data sets where we have preconceptions of the outcome, for example that works by the same authors should cluster together, or music pieces by the same composers, musical genres, or genomes, the outcomes conform largely to our expectations. For example, in the music genre experiment the method would fail dramatically if genres were evenly mixed, or mixed with little bias. However, to the contrary, the separation in clusters is almost perfect. The few misplacements that are discernable are either errors (the method was not powerful enough to discern the dominant feature), or the dominant feature between a pair of music pieces is not the genre but some other aspect. The surprising news is that we can generally confirm expectations with few misplacements, indeed, that the data don't contain unknown rogue features that dominate to cause spurious (in our preconceived idea) clustering. This gives evidence that where the preconception is in doubt, like with phylogeny hypotheses, the clustering can give true support of one hypothesis against another one.

Figures: We use two styles to display the hierarchical clusters. In the case of genomics of Eutherian orders and fungi, language trees, it is convenient to follow the dendrograms that are customary in that area (suggesting temporal evolution) for easy comparison with the literature. Although there is no temporal relation intended, the dendrogram representation looked also appropriate for the Russian writers, and translations of Russian writers. In the other experiments (even the genomic SARS experiment) it is more informative to display an unrooted ternary tree (or binary tree if we think about incoming and outgoing edges) with explicit internal nodes. This facilitates identification of clusters in terms of subtrees rooted at internal nodes or contiguous sets of subtrees rooted at branches of internal nodes.

2 Similarity Distance

We give a precise formal meaning to the loose distance notion of "degree of similarity" used in the pattern recognition literature.

2.1 Distance and Metric

Let Ω be a nonempty set and $\mathcal{R}^+$ be the set of nonnegative real numbers. A *distance function* on Ω is a function $D : \Omega \times \Omega \to \mathcal{R}^+$. It is a *metric* if it satisfies the metric (in)equalities:

- $D(x,y) = 0$ iff $x = y$,
- $D(x,y) = D(y,x)$ (symmetry), and
- $D(x,y) \leq D(x,z) + D(z,y)$ (triangle inequality).

The value $D(x,y)$ is called the *distance* between $x,y \in \Omega$. A familiar example of a distance that is also metric is the Euclidean metric, the everyday distance $e(a,b)$ between two objects a,b expressed in, say, meters. Clearly, this distance satisfies the properties $e(a,a) = 0$, $e(a,b) = e(b,a)$, and $e(a,b) \leq e(a,c) + e(c,b)$ (for instance, $a =$ Amsterdam, $b =$ Brussels, and $c =$ Chicago.) We are interested in a particular type of distance, the "similarity distance", which we formally define in Definition 2.5. For example, if the objects are classical music pieces then the function $D(a,b) = 0$ if a and b are by the same composer and $D(a,b) = 1$ otherwise, is a similarity distance that is also a metric. This metric captures only one similarity aspect (feature) of music pieces, presumably an important one that subsumes a conglomerate of more elementary features.

2.2 Admissible Distance

In defining a class of admissible distances (not necessarily metric distances) we want to exclude unrealistic ones like $f(x,y) = \frac{1}{2}$ for *every* pair $x \neq y$. We do this by restricting the number of objects within a given distance of an object. As in [4] we do this by only considering effective distances, as follows. Fix a suitable, and for the remainder of the paper, fixed, programming language. This is the *reference programming* language.

Definition 2.1 Let $\Omega = \Sigma^*$, with Σ a finite nonempty alphabet and Σ^* the set of finite strings over that alphabet. Since every finite alphabet can be recoded in binary, we choose $\Sigma = \{0,1\}$. In particular, "files" in computer memory are finite binary strings. A function $D : \Omega \times \Omega \to \mathcal{R}^+$ is an *admissible distance* if for every pair of objects $x, y \in \Omega$ the distance $D(x,y)$ is the length of a binary prefix code-word that is a program that computes x from y, and vice versa, in the reference programming language. This implies that an admissible distance is *symmetric*.

If D is an admissible distance, then for every x the set $\{D(x,y) : y \in \{0,1\}^*\}$ is the length set of a prefix code. Hence, it satisfies by the Kraft inequality, see [12],

$$\sum_y 2^{-D(x,y)} \leq 1. \tag{2.1}$$

Example 2.2 In representing the Hamming distance d between x and y strings of equal length n differing in positions $i_1, \ldots, i_d$, we can use a simple prefix-free encoding of $(n, d, i_1, \ldots, i_d)$ in $2\log n + 4\log\log n + 2 + d\log n$ bits. We encode n and d prefix-free in $\log n + 2\log\log n + 1$ bits each, see e.g. [32], and then the literal indexes of the actual flipped-bit positions. Adding an $O(1)$-bit program to interpret these data, we have defined $H_n(x,y) = 2\log n + 4\log\log n + 2 + d\log n + O(1)$ as the length of a prefix code word (prefix program) to compute x from y and *vice versa*. Then, by the Kraft inequality, $\sum_y 2^{-H_n(x,y)} \leq 1$. It is easy to verify that H_n is a metric in the sense that it satisfies the metric (in)equalities up to $O(\log n)$ additive precision. ♢

2.3 Normalized Admissible Distance

Large objects (in the sense of long strings) that differ by a tiny part are intuitively closer than tiny objects that differ by the same amount. For example, two whole mitochondrial genomes of 18,000 bases that differ by 9,000 are very different, while two whole nuclear genomes of 3×10^9 bases that differ by only 9,000 bases are very similar. Thus, absolute difference between two objects doesn't govern similarity, but relative difference appears to do so.

Definition 2.3 A *compressor* is a lossless encoder mapping Ω into $\{0,1\}^*$ such that the resulting code is a prefix code. "Lossless" means that there is a decompressor that reconstructs the source message from the code message. For convenience of notation we identify "compressor" with a "code word length function" $C : \Omega \to \mathcal{N}$, where $\mathcal{N}$ is the set of nonnegative integers. That is, the compressed version of a file x has length $C(x)$. We only consider compressors such that $C(x) \leq |x| + O(\log|x|)$. (The additive logarithmic term is due to our requirement that the compressed file be a prefix code word.) We fix a compressor C, and call the fixed compressor the *reference compressor*.

Definition 2.4 Let D be an admissible distance. Then $D^+(x,y)$ is defined by $D^+(x,y) = \max\{\max\{D(x,z) : C(z) \leq C(y)\}, \max\{D(z,y) : C(z) \leq C(x)\}\}$. Note that since $D(x,y) = D(y,x)$, also $D^+(x,y) = D^+(y,x)$. Furthermore, define $D^+(x) = \max\{D^+(x,y) : y \in \Omega\}$.

Definition 2.5 Let D be an admissible distance. The *normalized admissible distance*, also called a *similarity distance*, $d(x,y)$, based on D relative to a reference compressor C, is defined by

$$d(x,y) = \frac{D(x,y)}{D^+(x,y)}.$$

It follows from the definitions that a normalized admissible distance is a function $d : \Omega \times \Omega \to [0,1]$ that is symmetric: $d(x,y) = d(y,x)$.

Lemma 2.6 *For every $x \in \Omega$, and constant $e \in [0,1]$, a normalized admissible distance satisfies the density constraint*

$$|\{y : d(x,y) \leq e\}| < 2^{eD^+(x)+1}. \tag{2.2}$$

PROOF. Assume to the contrary that d does not satisfy (2.2). Then, there is an $e \in [0,1]$, such that (2.2) is false. We first note that, since $D(x,y)$ is an admissible distance that satisfies (2.1), $d(x,y)$ satisfies a "normalized" version of the Kraft inequality:

$$\sum_y 2^{-d(x,y)D^+(x)} \leq \sum_y 2^{-d(x,y)D^+(x,y)} \leq 1. \tag{2.3}$$

Starting from (2.3) we obtain the required contradiction:

$$1 \geq \sum_y 2^{-d(x,y)D^+(x)} \geq \sum_{y:d(x,y)\leq e} 2^{-eD^+(x)} \geq 2^{eD^+(x)+1} 2^{-eD^+(x)} > 1.$$

□

If $d(x,y)$ is the normalized version of an admissible distance $D(x,y)$ then (2.3) is equivalent to (2.1). We call a normalized distance a "similarity" distance, because it gives a relative similarity according to the distance (with distance 0 when objects are maximally similar and distance 1 when they are maximally dissimilar) and, conversely, for every well-defined computable notion of similarity we can express it as a metric distance according to our definition. In the literature a distance that expresses lack of similarity (like ours) is often called a "dissimilarity" distance or a "disparity" distance.

Remark 2.7 As far as the authors know, the idea of normalized metric is, surprisingly, not well-studied. An exception is [41], which investigates normalized metrics to account for relative distances rather than absolute ones, and it does so for much the same reasons as in the present work. An example there is the normalized Euclidean metric $|x-y|/(|x|+|y|)$, where $x,y \in \mathcal{R}^n$ ($\mathcal{R}$ denotes the real numbers) and $|\cdot|$ is the Euclidean metric—the L_2 norm. Another example is a normalized symmetric-set-difference metric. But these normalized metrics are not necessarily effective in that the distance between two objects gives the length of an effective description to go from either object to the other one. ♢

Remark 2.8 Our definition of normalized admissible distance is more direct than in [31], and the density constraints (2.2) and (2.3) follow from the definition. In [31] we put a stricter density condition in the definition of "admissible" normalized distance, which is, however, harder to satisfy and maybe too strict to be realistic. The purpose of this stricter density condition was to obtain a stronger "universality" property than the present Theorem 6.3, namely one with $\alpha = 1$ and $\varepsilon = O(1/\max\{C(x),C(y)\})$. Nonetheless, both definitions coincide if we set the length of the compressed version $C(x)$ of x to the ultimate compressed length $K(x)$, the Kolmogorov complexity of x. ♢

Example 2.9 To obtain a normalized version of the Hamming distance of Example 2.2, we define $h_n(x,y) = H_n(x,y)/H_n^+(x,y)$. We can set $H_n^+(x,y) = H_n^+(x) = (n+2)\lceil \log n \rceil + 4\lceil \log\log n \rceil$ since every contemplated compressor C will satisfy $C(x) = C(\overline{x})$, where $\overline{x}$ is x with all bits flipped (so $H_n^+(x,y) \geq H_n^+(z,\overline{z})$ for either $z = x$ or $z = y$). By (2.2), for every x, the number of y in the Hamming ball $h_n(x,y) \leq e$ is less than $2^{eH_n^+(x)+1}$. This upper bound is an obvious overestimate for $e \geq 1/\log n$. For lower values of e, the upper bound is correct by the observation that the number of y's equals $\sum_{i=0}^{en} \binom{n}{en} \leq 2^{nH(e)}$, where $H(e) = e\log e + (1-e)\log(1-e)$, Shannon's entropy function. Then, $eH_n^+(x) > en\log n > enH(e)$ since $e\log n > H(e)$. ♢

3 Normal Compressor

We give axioms determining a large family of compressors that both include most (if not all) real-world compressors and ensure the desired properties of the NCD to be defined later.

Definition 3.1 A compressor C is *normal* if it satisfies, up to an additive $O(\log n)$ term, with n the maximal binary length of an element of Ω involved in the (in)equality concerned, the following:

1. *Idempotency*: $C(xx) = C(x)$ and $C(\lambda) = 0$, where λ is the empty string.
2. *Monotonicity*: $C(xy) \geq C(x)$.
3. *Symmetry*: $C(xy) = C(yx)$.
4. *Distributivity*: $C(xy) + C(z) \leq C(xz) + C(yz)$.

Idempotency: A reasonable compressor will see exact repetitions and obey idempotency up to the required precision. It will also compress the empty string to the empty string.

Monotonicity: A real compressor must have the monotonicity property, at least up to the required precision. The property is evident for stream-based compressors, and only slightly less evident for block-coding compressors.

Symmetry: Stream-based compressors of the Lempel-Ziv family, like gzip and pkzip, and the predictive PPM family, like PPMZ, are possibly not precisely symmetric. This is related to the stream-based property: the initial file x may have regularities to which the compressor adapts; after crossing the border to y it must unlearn those regularities and adapt to the ones of x. This process may cause some imprecision in symmetry that vanishes asymptotically with the length of x,y. A compressor must be poor indeed (and will certainly not be used to any extent) if it doesn't satisfy symmetry up to the required precision. Apart from stream-based, the other major family of compressors is block-coding based, like bzip2. They essentially analyze the full input block by considering all rotations in obtaining the compressed version. It is to a great extent symmetrical, and real experiments show no departure from symmetry.

Distributivity: The distributivity property is not immediately intuitive. In Kolmogorov complexity theory the stronger distributivity property

$$C(xyz)+C(z)\leq C(xz)+C(yz) \tag{3.1}$$

holds (with $K=C$). However, to prove the desired properties of NCD below, only the weaker distributivity property

$$C(xy)+C(z)\leq C(xz)+C(yz) \tag{3.2}$$

above is required, also for the boundary case were $C=K$. In practice, real-world compressors appear to satisfy this weaker distributivity property up to the required precision.

Definition 3.2 Define

$$C(y|x)=C(xy)-C(x). \tag{3.3}$$

This number $C(y|x)$ of bits of information in y, relative to x, can be viewed as the excess number of bits in the compressed version of xy compared to the compressed version of x, and is called the amount of *conditional compressed information*.

In the definition of compressor the decompression algorithm is not included (unlike the case of Kolmorogov complexity, where the decompressing algorithm is given by definition), but it is easy to construct one: Given the compressed version of x in $C(x)$ bits, we can run the compressor on all candidate strings z—for example, in length-increasing lexicographical order, until we find the compressed string $z_0=x$. Since this string decompresses to x we have found $x=z_0$. Given the compressed version of xy in $C(xy)$ bits, we repeat this process using strings xz until we find the string xz_1 of which the compressed version equals the compressed version of xy. Since the former compressed version decompresses to xy, we have found $y=z_1$. By the unique decompression property we find that $C(y|x)$ is the extra number of bits we require to describe y apart from describing x. It is intuitively acceptable that the conditional compressed information $C(x|y)$ satisfies the triangle inequality

$$C(x|y)\leq C(x|z)+C(z|y). \tag{3.4}$$

Lemma 3.3 *Both* (3.1) *and* (3.4) *imply* (3.2).

PROOF. ((3.1) implies (3.2):) By monotonicity.

((3.4) implies (3.2):) Rewrite the terms in (3.4) according to (3.3), cancel $C(y)$ in the left- and right-hand sides, use symmetry, and rearrange. □

Lemma 3.4 *A normal compressor satisfies additionally* subadditivity*:* $C(xy)\leq C(x)+C(y)$.

PROOF. Consider the special case of distributivity with z the empty word so that $xz=x$, $yz=y$, and $C(z)=0$. □

Subadditivity: The subadditivity property is clearly also required for every viable compressor, since a compressor may use information acquired from x to compress y. Minor imprecision may arise from the unlearning effect of crossing the border between x and y, mentioned in relation to symmetry, but again this must vanish asymptotically with increasing length of x,y.

4 Background in Kolmogorov complexity

Technically, the *Kolmogorov complexity* of x given y is the length of the shortest binary program, for the reference universal prefix Turing machine, that on input y outputs x; it is denoted as $K(x|y)$. For precise definitions, theory and applications, see [32]. The Kolmogorov complexity of x is the length of the shortest binary program with no input that outputs x; it is denoted as $K(x) = K(x|\lambda)$ where λ denotes the empty input. Essentially, the Kolmogorov complexity of a file is the length of the ultimate compressed version of the file. In [4] the *information distance* $E(x,y)$ was introduced, defined as the length of the shortest binary program for the reference universal prefix Turing machine that, with input x computes y, and with input y computes x. It was shown there that, up to an additive logarithmic term, $E(x,y) = \max\{K(x|y),K(y|x)\}$. It was shown also that $E(x,y)$ is a metric, up to negligible violations of the metric inequalties. Moreover, it is universal in the sense that for every admissible distance $D(x,y)$ as in Definition 2.1, $E(x,y) \leq D(x,y)$ up to an additive constant depending on D but not on x and y. In [31], the normalized version of $E(x,y)$, called the *normalized information distance*, is defined as

$$\mathrm{NID}(x,y) = \frac{\max\{K(x|y),K(y|x)\}}{\max\{K(x),K(y)\}}. \tag{4.1}$$

It too is a metric, and it is universal in the sense that this single metric minorizes up to an negligible additive error term all normalized admissible distances in the class considered in [31]. Thus, if two files (of whatever type) are similar (that is, close) according to the particular feature described by a particular normalized admissible distance (not necessarily metric), then they are also similar (that is, close) in the sense of the normalized information metric. This justifies calling the latter *the* similarity metric. We stress once more that different pairs of objects may have different dominating features. Yet every such dominant similarity is detected by the NID . However, this metric is based on the notion of Kolmogorov complexity. Unfortunately, the Kolmogorov complexity is non-computable in the Turing sense. Approximation of the denominator of (4.1) by a given compressor C is straightforward: it is $\max\{C(x),C(y)\}$. The numerator is more tricky. It can be rewritten as

$$\max\{K(x,y) - K(x), K(x,y) - K(y)\}, \tag{4.2}$$

within logarithmic additive precision, by the additive property of Kolmogorov complexity [32]. The term $K(x,y)$ represents the length of the shortest program for the pair (x,y). In compression practice it is easier to deal with the concatenation xy or yx. Again, within logarithmic precision $K(x,y) = K(xy) = K(yx)$. Following a suggestion by Steven de Rooij, one can approximate (4.2) best by $\min\{C(xy),C(yx)\} - \min\{C(x),C(y)\}$. Here, and in the later experiments using the CompLearn Toolkit [9], we simply use $C(xy)$ rather than $\min\{C(xy),C(yx)\}$. This is justified by the observation that block-coding based compressors are symmetric almost by definition, and experiments with various stream-based compressors (gzip, PPMZ) show only small deviations from symmetry.

The theory as developed for the Kolmogorov-complexity based NID in [31], may not hold for the (possibly poorly) approximating NCD . It is nonetheless the case that experiments show that the NCD apparently has (some) properties that make the NID so appealing. To fill this gap between theory and practice, we develop the theory of NCD from first principles, based on the axiomatics of Section 3. We show that the NCD is a quasi-universal similarity metric relative to a normal reference compressor C. The theory developed in [31] is the boundary case $C = K$, where the "quasi-universality" below has become full "universality".

5 Compression Distance

We define a compression distance based on a normal compressor and show it is an admissible distance. In applying the approach, we have to make do with an approximation based on a far less powerful real-world reference compressor C. A compressor C approximates the information distance $E(x,y)$, based on Kolmogorov complexity, by the compression distance $E_C(x,y)$ defined as

$$E_C(x,y) = C(xy) - \min\{C(x),C(y)\}. \tag{5.1}$$

Here, $C(xy)$ denotes the compressed size of the concatenation of x and y, $C(x)$ denotes the compressed size of x, and $C(y)$ denotes the compressed size of y.

Lemma 5.1 *If C is a normal compressor, then $E_C(x,y) + O(1)$ is an admissible distance.*

PROOF. **Case 1:** Assume $C(x) \leq C(y)$. Then $E_C(x,y) = C(xy) - C(x)$. Then, given x and a prefix-program of length $E_C(x,y)$ consisting of the suffix of the C-compressed version of xy, and the compressor C in $O(1)$ bits, we can run the compressor C on all xz's, the candidate strings z in length-increasing lexicographical. When we find a z so that the suffix of the compressed version of xz matches the given suffix, then $z = y$ by the unique decompression property.

Case 2: Assume $C(y) \geq C(x)$. By symmetry $C(xy) = C(yx)$. Now follow the proof of Case 1. □

Lemma 5.2 *If C is a normal compressor, then $E_C(x,y)$ satisfies the metric (in)equalities up to logarithmic additive precision.*

PROOF. Only the triangular inequality is non-obvious. By (3.2) $C(xy) + C(z) \leq C(xz) + C(yz)$ up to logarithmic additive precision. There are six possibilities, and we verify the correctness of the triangular inequality in turn for each of them. Assume $C(x) \leq C(y) \leq C(z)$: Then $C(xy) - C(x) \leq C(xz) - C(x) + C(yz) - C(y)$. Assume $C(y) \leq C(x) \leq C(z)$: Then $C(xy) - C(y) \leq C(xz) - C(y) + C(yz) - C(x)$. Assume $C(x) \leq C(z) \leq C(y)$: Then $C(xy) - C(x) \leq C(xz) - C(x) + C(yz) - C(z)$. Assume $C(y) \leq C(z) \leq C(x)$: Then $C(xy) - C(y) \leq C(xz) - C(z) + C(yz) - C(y)$. Assume $C(z) \leq C(x) \leq C(y)$: Then $C(xy) - C(x) \leq C(xz) - C(z) + C(yz) - C(z)$. Assume $C(z) \leq C(y) \leq C(x)$: Then $C(xy) - C(y) \leq C(xz) - C(z) + C(yz) - C(z)$. □

Lemma 5.3 *If C is a normal compressor, then $E_C^+(x,y) = \max\{C(x), C(y)\}$.*

PROOF. Consider a pair (x,y). The $\max\{C(xz) - C(z) : C(z) \leq C(y)\}$ is $C(x)$ which is achieved for $z = \lambda$, the empty word, with $C(\lambda) = 0$. Similarly, the $\max\{C(yz) - C(z) : C(z) \leq C(x)\}$ is $C(y)$. Hence the lemma. □

6 Normalized Compression Distance

The normalized version of the admissible distance $E_C(x,y)$, the compressor C based approximation of the normalized information distance (4.1), is called the *normalized compression distance* or NCD:

$$\mathrm{NCD}(x,y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}. \tag{6.1}$$

This NCD is the main concept of this work. It is the real-world version of the ideal notion of normalized information distance NID in (4.1).

Remark 6.1 In practice, the NCD is a non-negative number $0 \leq r \leq 1 + \varepsilon$ representing how different the two files are. Smaller numbers represent more similar files. The ε in the upper bound is due to imperfections in our compression techniques, but for most standard compression algorithms one is unlikely to see an ε above 0.1 (in our experiments gzip and bzip2 achieved NCD 's above 1, but PPMZ always had NCD at most 1). ♢

There is a natural interpretation to $\mathrm{NCD}(x,y)$: If, say, $C(y) \geq C(x)$ then we can rewrite

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

That is, the distance $\mathrm{NCD}(x,y)$ between x and y is the improvement due to compressing y using x as previously compressed "data base," and compressing y from scratch, expressed as the ratio between the bit-wise length of the two compressed versions. Relative to the reference compressor we can define the information in x about y as $C(y) - C(y|x)$. Then, using (3.3),

$$\mathrm{NCD}(x,y) = 1 - \frac{C(y) - C(y|x)}{C(y)}.$$

That is, the NCD between x and y is 1 minus the ratio of the information x about y and the information in y.

Theorem 6.2 *If the compressor is normal, then the* NCD *is a normalized admissible distance satsifying the metric (in)equalities, that is, a similarity metric.*

PROOF. If the compressor is normal, then by Lemma 5.1 and Lemma 5.3, the NCD is a normalized admissible distance. It remains to show it satisfies the three metric (in)equalities.

1. By idempotency we have $\mathrm{NCD}(x,x) = 0$. By monotonicity we have $\mathrm{NCD}(x,y) \geq 0$ for every x,y, with inequality for $y \neq x$.

2. $\mathrm{NCD}(x,y) = \mathrm{NCD}(y,x)$. The NCD is unchanged by interchanging x and y in (6.1).

3. The difficult property is the triangle inequality. Without loss of generality we assume $C(x) \leq C(y) \leq C(z)$. Since the NCD is symmetrical, there are only three triangle inequalities that can be expressed by $\mathrm{NCD}(x,y), \mathrm{NCD}(x,z), \mathrm{NCD}(y,z)$. We verify them in turn:

   (a) $\mathrm{NCD}(x,y) \leq \mathrm{NCD}(x,z) + \mathrm{NCD}(z,y)$: By distributivity, the compressor itself satisfies $C(xy) + C(z) \leq C(xz) + C(zy)$. Subtracting $C(x)$ from both sides and rewriting, $C(xy) - C(x) \leq C(xz) - C(x) + C(zy) - C(z)$. Dividing by $C(y)$ on both sides we find

   $$\frac{C(xy) - C(x)}{C(y)} \leq \frac{C(xz) - C(x) + C(zy) - C(z)}{C(y)}.$$

   The left-hand side is ≤ 1.

      i. Assume the right-hand side is ≤ 1. Setting $C(z) = C(y) + \Delta$, and adding Δ to both the numerator and denominator of the right-hand side, it can only increase and draw closer to 1. Therefore,

      $$\begin{aligned} \frac{C(xy) - C(x)}{C(y)} &\leq \frac{C(xz) - C(x) + C(zy) - C(z) + \Delta}{C(y) + \Delta} \\ &= \frac{C(zx) - C(x)}{C(z)} + \frac{C(zy) - C(y)}{C(z)}, \end{aligned}$$

      which was what we had to prove.

      ii. Assume the right-hand side is > 1. We proceed like in the previous case, and add Δ to both numerator and denominator. Although now the right-hand side decreases, it must still be greater than 1, and therefore the right-hand side remains at least as large as the left-hand side.

   (b) $\mathrm{NCD}(x,z) \leq \mathrm{NCD}(x,y) + \mathrm{NCD}(y,z)$: By distributivity we have $C(xz) + C(y) \leq C(xy) + C(yz)$. Subtracting $C(x)$ from both sides, rearranging, and dividing both sides by $C(z)$ we obtain

   $$\frac{C(xz) - C(x)}{C(z)} \leq \frac{C(xy) - C(x)}{C(z)} + \frac{C(yz) - C(y)}{C(z)}.$$

   The right-hand side doesn't decrease when we substitute $C(y)$ for the denominator $C(z)$ of the first term, since $C(y) \leq C(z)$. Therefore, the inequality stays valid under this substitution, which was what we had to prove.

   (c) $\mathrm{NCD}(y,z) \leq \mathrm{NCD}(y,x) + \mathrm{NCD}(x,z)$: By distributivity we have $C(yz) + C(x) \leq C(yx) + C(xz)$. Subtracting $C(y)$ from both sides, rearanging, and dividing both sides by $C(z)$ we obtain

   $$\frac{C(yz) - C(y)}{C(z)} \leq \frac{C(yx) - C(x)}{C(z)} + \frac{C(yz) - C(y)}{C(z)}.$$

   The right-hand side doesn't decrease when we substitute $C(y)$ for the denominator $C(z)$ of the first term, since $C(y) \leq C(z)$. Therefore, the inequality stays valid under this substitution, which was what we had to prove.

□

Quasi-Universality: We now digress to the theory developed in [31], which formed the motivation for developing the NCD . If, instead of the result of some real compressor, we substitute the Kolmogorov complexity for the lengths of the compressed files in the NCD formula, the result is the NID as in (4.1). It is universal in the following sense: Every admissible distance expressing similarity according to some feature, that can be computed from the objects concerned, is comprised (in the sense of minorized) by the NID . Note that every feature of the data gives rise to a similarity, and, conversely, every similarity can be thought of as expressing some feature: being similar in that sense. Our actual practice in using the NCD falls short of this ideal theory in at least three respects:

(i) The claimed universality of the NID holds only for indefinitely long sequences x,y. Once we consider strings x,y of definite length n, it is only universal with respect to "simple" computable normalized admissible distances,

where "simple" means that they are computable by programs of length, say, logarithmic in n. This reflects the fact that, technically speaking, the universality is achieved by summing the weighted contribution of all similarity distances in the class considered with respect to the objects considered. Only similarity distances of which the complexity is small (which means that the weight is large), with respect to the size of the data concerned, kick in.

(ii) The Kolmogorov complexity is not computable, and it is in principle impossible to compute how far off the NCD is from the NID . So we cannot in general know how well we are doing using the NCD .

(iii) To approximate the NCD we use standard compression programs like gzip, PPMZ, and bzip2. While better compression of a string will always approximate the Kolmogorov complexity better, this may not be true for the NCD . Due to its arithmetic form, subtraction and division, it is theoretically possible that while all items in the formula get better compressed, the improvement is not the same for all items, and the NCD value moves away from the NID value. In our experiments we have not observed this behavior in a noticable fashion. Formally, we can state the following:

Theorem 6.3 *Let d be a computable normalized admissible distance and C be a normal compressor. Then,* $\mathrm{NCD}(x,y) \leq \alpha d(x,y) + \varepsilon$, *where for $C(x) \geq C(y)$ we have $\alpha = D^+(x)/C(x)$ and $\varepsilon = (C(x|y) - K(x|y))/C(x)$, with $C(x|y)$ according to* (3.3).

PROOF. Fix d, C, x, y in the statement of the theorem. Since the NCD is symmetrical, we can, without loss of generality, let $C(x) \geq C(y)$. By (3.3) and the symmetry property $C(xy) = C(yx)$ we have $C(x|y) \geq C(y|x)$. Therefore, $\mathrm{NCD}(x,y) = C(x|y)/C(x)$. Let $d(x,y)$ be the normalized version of the admissible distance $D(x,y)$; that is, $d(x,y) = D(x,y)/D^+(x,y)$. Let $d(x,y) = e$. By (2.2), there are $< 2^{eD^+(x)+1}$ many (x,v) pairs, such that $d(x,v) \leq e$. Since d is computable, we can compute and enumerate all these pairs. The initially fixed pair (x,y) is an element in the list and its index takes $\leq eD^+(x) + 1$ bits. Therefore, given x, the y can be described by at most $eD^+(x) + O(1)$ bits—its index in the list and an $O(1)$ term accounting for the lengths of the programs involved in reconstructing y given its index in the list, and algorithms to compute functions d and C. Since the Kolmogorov complexity gives the length of the shortest effective description, we have $K(y|x) \leq eD^+(x) + O(1)$. Substitution and rewriting yields $\mathrm{NCD}(x,y) = C(x|y)/C(x) \leq \alpha e + \varepsilon$, which was what we had to prove.

□

Remark 6.4 Clustering according to NCD will group sequences together that are similar according to features that are not explicitly known to us. Analysis of what the compressor actually does, still may not tell us which features that make sense to us can be expressed by conglomerates of features analyzed by the compressor. This can be exploited to track down unknown features implicitly in classification: forming automatically clusters of data and see in which cluster (if any) a new candidate is placed.

Another aspect that can be exploited is exploratory: Given that the NCD is small for a pair x,y of specific sequences, what does this really say about the sense in which these two sequences are similar? The above analysis suggests that close similarity will be due to a dominating feature (that perhaps expresses a conglomerate of subfeatures). Looking into these deeper causes may give feedback about the appropriateness of the realized NCD distances and may help extract more intrinsic information about the objects, than the oblivious division into clusters, by looking for the common features in the data clusters. ♢

7 Clustering

Given a set of objects, the pairwise NCD 's form the entries of a distance matrix. This distance matrix contains the pairwise relations in raw form. But in this format that information is not easily usable. Just as the distance matrix is a reduced form of information representing the original data set, we now need to reduce the information even further in order to achieve a cognitively acceptable format like data clusters. To extract a hierarchy of clusters from the distance matrix, we determine a dendrogram (binary tree) that agrees with the distance matrix according to a cost measure. This allows us to extract more information from the data than just flat clustering (determining disjoint clusters in dimensional representation).

Clusters are groups of objects that are similar according to our metric. There are various ways to cluster. Our aim is to analyze data sets for which the number of clusters is not known a priori, and the data are not labeled. As stated in [16], conceptually simple, hierarchical clustering is among the best known unsupervised methods in this setting, and the most natural way is to represent the relations in the form of a dendrogram, which is customarily a directed binary tree or undirected ternary tree. To construct the tree from a distance matrix with entries consisting of the pairwise distances between objects, we use a quartet method. This is a matter of choice only, other methods may work equally

well. The distances we compute in our experiments are often within the range 0.85 to 1.2. That is, the distinguishing features are small, and we need a sensitive method to extract as much information contained in the distance matrix as is possible. For example, our experiments showed that reconstructing a minimum spanning tree is not sensitive enough and gives poor results. With increasing number of data items, the projection of the NCD matrix information into the tree representation format gets increasingly distorted. A similar situation arises in using alignment cost in genomic comparisons. Experience shows that in both cases the hierarchical clustering methods seem to work best for small sets of data, up to 25 items, and to deteriorate for larger sets, say 40 items or more. A standard solution to hierarchically cluster larger sets of data is to first cluster nonhierarchically, by say multidimensional scaling of *k*-means, available in standard packages, for instance *Matlab*, and then apply hierarchical clustering on the emerging clusters.

The quartet method: We consider every group of four elements from our set of n elements; there are $\binom{n}{4}$ such groups. From each group u,v,w,x we construct a tree of arity 3, which implies that the tree consists of two subtrees of two leaves each. Let us call such a tree a *quartet topology*. There are three possibilities denoted (i) $uv|wx$, (ii) $uw|vx$, and (iii) $ux|vw$, where a vertical bar divides the two pairs of leaf nodes into two disjoint subtrees (Figure 1).

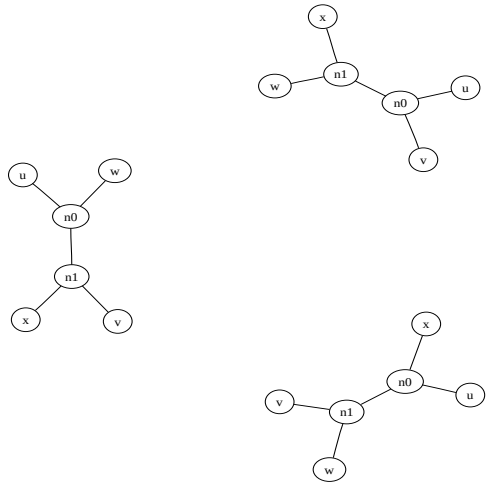

Figure 1: The three possible quartet topologies for the set of leaf labels *u,v,w,x*

For any given tree T and any group of four leaf labels u,v,w,x, we say T is *consistent* with $uv|wx$ if and only if the path from u to v does not cross the path from w to x. Note that exactly one of the three possible quartet topologies for any set of 4 labels must be consistent for any given tree. We may think of a large tree having many smaller

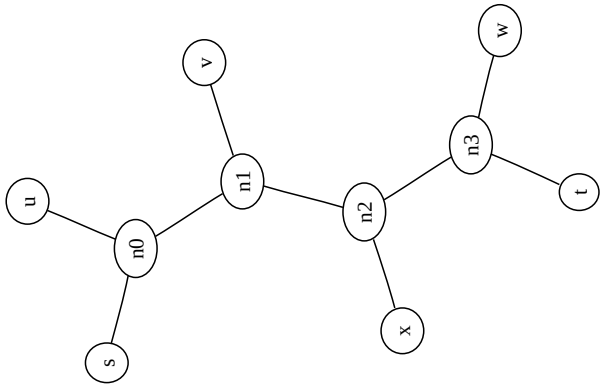

Figure 2: An example tree consistent with quartet topology $uv|wx$

quartet topolgies embedded within its structure. Commonly the goal in the quartet method is to find (or approximate as closely as possible) the tree that embeds the maximal number of consistent (possibly weighted) quartet topologies from a given set Q of quartet topologies [19] (Figure 2). This is called the (weighted) *Maximum Quartet Consistency*

(MQC) problem.

We propose a new optimization problem: the *Minimum Quartet Tree Cost (MQTC)*, as follows: The cost of a quartet topology is defined as the sum of the distances between each pair of neighbors; that is, $C_{uv|wx} = d(u,v) + d(w,x)$. The total cost C_T of a tree T with a set N of leaves (external nodes of degree 1) is defined as $C_T = \sum_{\{u,v,w,x\}\subseteq N}\{C_{uv|wx} : T \text{ is consistent with } uv|wx\}$—the sum of the costs of all its consistent quartet topologies. First, we generate a list of all possible quartet topolgies for all four-tuples of labels under consideration. For each group of three possible quartet topologies for a given set of four labels u,v,w,x, calculate a best (minimal) cost $m(u,v,w,x) = \min\{C_{uv|wx}, C_{uw|vx}, C_{ux|vw}\}$, and a worst (maximal) cost $M(u,v,w,x) = \max\{C_{uv|wx}, C_{uw|vx}, C_{ux|vw}\}$. Summing all best quartet toplogies yields the best (minimal) cost $m = \sum_{\{u,v,w,x\}\subseteq N} m(u,v,w,x)$. Conversely, summing all worst quartet toplogies yields the worst (maximal) cost $M = \sum_{\{u,v,w,x\}\subseteq N} M(u,v,w,x)$. For some distance matrices, these minimal and maximal values can not be attained by actual trees; however, the score C_T of every tree T will lie between these two values. In order to be able to compare tree scores in a more uniform way, we now rescale the score linearly such that the worst score maps to 0, and the best score maps to 1, and term this the *normalized tree benefit score* $S(T) = (M - C_T)/(M - m)$. Our goal is to find a full tree with a maximum value of $S(T)$, which is to say, the lowest total cost.

To express the notion of computational difficulty one uses the notion of "nondeterministic polynomial time (NP)". If a problem concerning n objects is NP-hard this means that the best known algorithm for this (and a wide class of significant problems) requires computation time exponential in n. That is, it is infeasible in practice. The *MQC decision problem* is the following: Given n objects, let T be a tree of which the n leaves are labeled by the objects, and let Q_T be the set of quartet topologies embedded in T. Given a set of quartet topologies Q, and an integer k, the problem is to decide whether there is a binary tree T such that $Q \bigcap Q_T > k$. In [19] it is shown that the MQC decision problem is NP-hard. For every MQC decision problem one can define an MQTC problem that has the same solution: give the quartet topologies in Q cost 0 and the other ones cost 1. This way the MQC decision problem can be reduced to the MQTC decision problem, which shows also the latter to be NP-hard. Hence, it is infeasible in practice, but we can sometimes solve it, and always approximate it. (The reduction also shows that the quartet problems reviewed in [19], are subsumed by our problem.) Adapting current methods in [6] to our MQTC optimization problem, results in far too computationally intensive calculations; they run many months or years on moderate-sized problems of 30 objects. Therefore, we have designed a simple, feasible, heuristic method for our problem based on randomization and hill-climbing. First, a random tree with $2n-2$ nodes is created, consisting of n leaf nodes (with 1 connecting edge) labeled with the names of the data items, and $n-2$ non-leaf or *internal* nodes labeled with the lowercase letter "n" followed by a unique integer identifier. Each internal node has exactly three connecting edges. For this tree T, we calculate the total cost of all embedded quartet toplogies, and invert and scale this value to find $S(T)$. A tree is consistent with precisely $\frac{1}{3}$ of all quartet topologies, one for every quartet. A random tree may be consistent with about $\frac{1}{3}$ of the best quartet topologies—but because of dependencies this figure is not precise. The initial random this tree is chosen as the currently best known tree, and is used as the basis for further searching. We define a simple mutation on a tree as one of the three possible transformations:

1. A *leaf swap*, which consists of randomly choosing two leaf nodes and swapping them.

2. A *subtree swap*, which consists of randomly choosing two internal nodes and swapping the subtrees rooted at those nodes.

3. A *subtree transfer*, whereby a randomly chosen subtree (possibly a leaf) is detached and reattached in another place, maintaining arity invariants.

Each of these simple mutations keeps the number of leaf nodes and internal nodes in the tree invariant; only the structure and placements change. Define a full mutation as a sequence of at least one but potentially many simple mutations, picked according to the following distribution. First we pick the number k of simple mutations that we will perform with probability 2^{-k}. For each such simple mutation, we choose one of the three types listed above with equal probability. Finally, for each of these simple mutations, we pick leaves or internal nodes, as necessary. Notice that trees which are close to the original tree (in terms of number of simple mutation steps in between) are examined often, while trees that are far away from the original tree will eventually be examined, but not very frequently. In order to search for a better tree, we simply apply a full mutation on T to arrive at T', and then calculate $S(T')$. If $S(T') > S(T)$, then keep T' as the new best tree. Otherwise, try a new different tree and repeat. If $S(T')$ ever reaches 1, then halt, outputting the best tree. Otherwise, run until it seems no better trees are being found in a reasonable amount of time, in which case the approximation is complete.

Note that if a tree is ever found such that $S(T) = 1$, then we can stop because we can be certain that this tree is optimal, as no tree could have a lower cost. In fact, this perfect tree result is achieved in our artificial tree

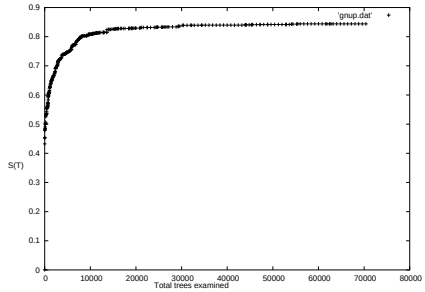

Figure 3: Progress of a 60-item data set experiment over time

reconstruction experiment (Section 7.1) reliably in a few minutes. For real-world data, $S(T)$ reaches a maximum somewhat less than 1, presumably reflecting distortion of the information in the distance matrix data by the best possible tree representation, as noted above, or indicating getting stuck in a local optimum or a search space too large to find the global optimum. On many typical problems of up to 40 objects this tree-search gives a tree with $S(T) \geq 0.9$ within half an hour. For large numbers of objects, tree scoring itself can be slow (as this takes order n^4 computation steps), and the space of trees is also large, so the algorithm may slow down substantially. For larger experiments, we use a C++/Ruby implementation with MPI (Message Passing Interface, a common standard used on massively parallel computers) on a cluster of workstations in parallel to find trees more rapidly. We can consider the graph mapping the achieved $S(T)$ score as a function of the number of trees examined. Progress occurs typically in a sigmoidal fashion towards a maximal value ≤ 1, Figure 3.

7.1 Three controlled experiments

With the natural data sets we use, one may have the preconception (or prejudice) that, say, music by Bach should be clustered together, music by Chopin should be clustered together, and so should music by rock stars. However, the preprocessed music files of a piece by Bach and a piece by Chopin, or the Beatles, may resemble one another more than two different pieces by Bach—by accident or indeed by design and copying. Thus, natural data sets may have ambiguous, conflicting, or counterintuitive outcomes. In other words, the experiments on natural data sets have the drawback of not having an objective clear "correct" answer that can function as a benchmark for assessing our experimental outcomes, but only intuitive or traditional preconceptions. We discuss three experiments that show that our program indeed does what it is supposed to do—at least in artificial situations where we know in advance what the correct answer is. The similarity machine consists of two parts: (i) extracting a distance matrix from the data, and (ii) constructing a tree from the distance matrix using our novel quartet-based heuristic.

Testing the quartet-based tree construction: We first test whether the quartet-based tree construction heuristic is trustworthy: We generated a ternary tree T with 18 leaves, using the pseudo-random number generator "rand" of the Ruby programming language, and derived a metric from it by defining the distance between two nodes as follows: Given the length of the path from a to b, in an integer number of edges, as $L(a,b)$, let

$$d(a,b) = \frac{L(a,b)+1}{18},$$

except when $a = b$, in which case $d(a,b) = 0$. It is easy to verify that this simple formula always gives a number between 0 and 1, and is monotonic with path length. Given only the 18×18 matrix of these normalized distances, our quartet method exactly reconstructed the original tree T represented in Figure 4, with $S(T) = 1$.

Testing the similarity machine on artificial data: Given that the tree reconstruction method is accurate on clean consistent data, we tried whether the full procedure works in an acceptable manner when we know what the outcome should be like. We used the "rand" pseudo-random number generator from the C programming language standard library under Linux. We randomly generated 11 separate 1-kilobyte blocks of data where each byte was equally probable and called these *tags*. Each tag was associated with a different lowercase letter of the alphabet. Next, we generated 22 files of 80 kilobyte each, by starting with a block of purely random bytes and applying one, two, three, or four different tags on it. Applying a tag consists of ten repetitions of picking a random location in the 80-kilobyte file, and overwriting that location with the globally consistent tag that is indicated. So, for instance, to create the file referred to in the diagram by "a," we start with 80 kilobytes of random data, then pick ten places to copy over this random data with the arbitrary 1-kilobyte sequence identified as tag a. Similarly, to create file "ab,"

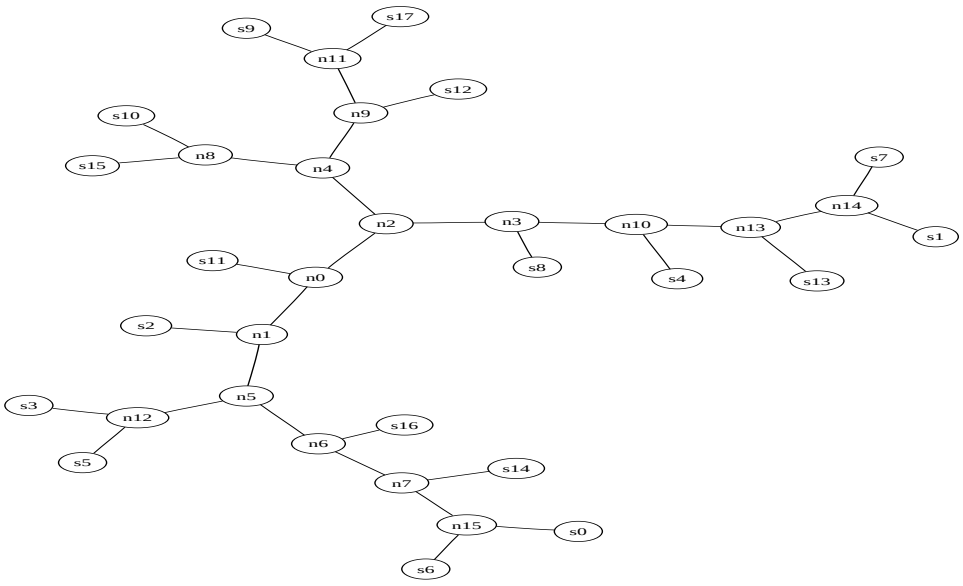

Figure 4: The randomly generated tree that our algorithm reconstructed. $S(T) = 1$.

we start with 80 kilobytes of random data, then pick ten places to put copies of tag *a*, then pick ten more places to put copies of tag *b* (perhaps overwriting some of the *a* tags). Because we never use more than four different tags, and therefore never place more than 40 copies of tags, we can expect that at least half of the data in each file is random and uncorrelated with the rest of the files. The rest of the file is correlated with other files that also contain tags in common; the more tags in common, the more related the files are. The compressor used to compute the NCD matrix was bzip2. The resulting tree is given in Figure 5; it can be seen that the clustering has occured exactly as we would expect. The $S(T)$ score is 0.905.

Testing the similarity machine on natural data: We test gross classification of files based on markedly different file types. Here, we chose several files: (i) Four mitochondrial gene sequences, from a black bear, polar bear, fox, and rat obtained from the GenBank Database on the world-wide web; (ii) Four excerpts from the novel *The Zeppelin's Passenger* by E. Phillips Oppenheim, obtained from the Project Gutenberg Edition on the World-Wide web; (iii) Four MIDI files without further processing; two from Jimi Hendrix and two movements from Debussy's Suite Bergamasque, downloaded from various repositories on the world-wide web; (iv) Two Linux x86 ELF executables (the *cp* and *rm* commands), copied directly from the RedHat 9.0 Linux distribution; and (v) Two compiled Java class files, generated by ourselves. The compressor used to compute the NCD matrix was bzip2. As expected, the program correctly classifies each of the different types of files together with like near like. The result is reported in Figure 6 with $S(T)$ equal to the very high confidence value 0.984. This experiment shows the power and universality of the method: no features of any specific domain of application are used.

8 Experimental Validation

We developed the CompLearn Toolkit, Section 1, and performed experiments in vastly different application fields to test the quality and universality of the method. The success of the method as reported below depends strongly on the judicious use of encoding of the objects compared. Here one should use common sense on what a real world compressor can do. There are situations where our approach fails if applied in a straightforward way. For example: comparing text files by the same authors in different encodings (say, Unicode and 8-bit version) is bound to fail. For the ideal similarity metric based on Kolmogorov complexity as defined in [31] this does not matter at all, but for practical compressors used in the experiments it will be fatal. Similarly, in the music experiments below we use symbolic MIDI music file format rather than wave format music files. The reason is that the strings resulting from straightforward discretizing the wave form files may be too sensitive to how we discretize.

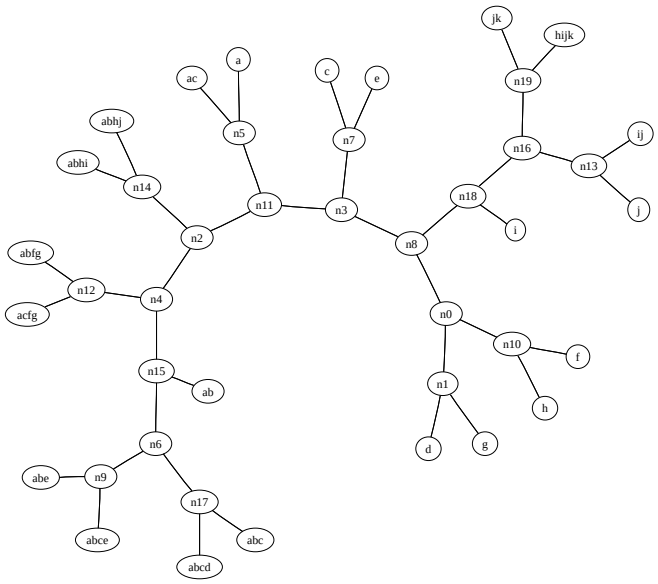

Figure 5: Classification of artificial files with repeated 1-kilobyte tags. Not all possiblities are included; for example, file "*b*" is missing. $S(T) = 0.905$.

8.1 Genomics and Phylogeny

In recent years, as the complete genomes of various species become available, it has become possible to do whole genome phylogeny (this overcomes the problem that useing different targeted parts of the genome, or proteins, may give different trees [36]). Traditional phylogenetic methods on individual genes depended on multiple alignment of the related proteins and on the model of evolution of individual amino acids. Neither of these is practically applicable to the genome level. In absence of such models, a method which can compute the shared information between two sequences is useful because biological sequences encode information, and the occurrence of evolutionary events (such as insertions, deletions, point mutations, rearrangements, and inversions) separating two sequences sharing a common ancestor will result in the loss of their shared information. Our method (in the form of the CompLearn Toolkit) is a fully automated software tool based on such a distance to compare two genomes.

Mammalian Evolution: In evolutionary biology the timing and origin of the major extant placental clades (groups of organisms that have evolved from a common ancestor) continues to fuel debate and research. Here, we provide evidence by whole mitochondrial genome phylogeny for competing hypotheses in two main questions: the grouping of the Eutherian orders, and the Therian hypothesis versus the Marsupionta hypothesis.

Eutherian Orders: We demonstrate (already in [31]) that a whole mitochondrial genome phylogeny of the Eutherians (placental mammals) can be reconstructed automatically from *unaligned* complete mitochondrial genomes by use of an early form of our compression method, using standard software packages. As more genomic material has become available, the debate in biology has intensified concerning which two of the three main groups of placental mammals are more closely related: Primates, Ferungulates, and Rodents. In [7], the maximum likelihood method of phylogeny tree reconstruction gave evidence for the (Ferungulates, (Primates, Rodents)) grouping for half of the proteins in the mitochondial genomes investigated, and (Rodents, (Ferungulates, Primates)) for the other halves of the mt genomes. In that experiment they aligned 12 concatenated mitochondrial proteins, taken from 20 species: rat (*Rattus norvegicus*), house mouse (*Mus musculus*), grey seal (*Halichoerus grypus*), harbor seal (*Phoca vitulina*), cat (*Felis catus*), white rhino (*Ceratotherium simum*), horse (*Equus caballus*), finback whale (*Balaenoptera physalus*), blue whale (*Balaenoptera musculus*), cow (*Bos taurus*), gibbon (*Hylobates lar*), gorilla (*Gorilla gorilla*), human (*Homo sapiens*), chimpanzee (*Pan troglodytes*), pygmy chimpanzee (*Pan paniscus*), orangutan (*Pongo pygmaeus*), Sumatran orangutan (*Pongo pygmaeus abelii*), using opossum (*Didelphis virginiana*), wallaroo (*Macropus robustus*), and the platypus (*Ornithorhynchus anatinus*) as outgroup. In [30, 31] we used the whole mitochondrial genome of the same 20 species, computing the NCD distances (or a closely related distance in [30]), using the GenCompress compressor, followed by tree reconstruction using the neighbor joining program in the MOLPHY package [38] to

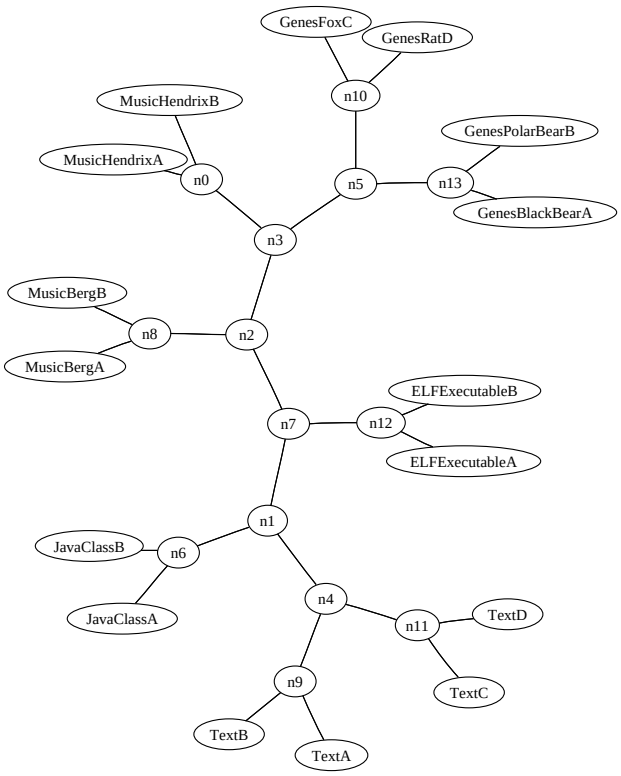

Figure 6: Classification of different file types. Tree agrees exceptionally well with NCD distance matrix: $S(T) = 0.984$.

confirm the commonly believed morphology-supported hypothesis (Rodents, (Primates, Ferungulates)). Repeating the experiment using the hypercleaning method [6] of phylogeny tree reconstruction gave the same result. Here, we repeated this experiment several times using the CompLearn Toolkit using our new quartet method for reconstructing trees, and computing the NCD with various compressors (gzip, bzip2, PPMZ), again always with the same result. These experiments are not reported since they are subsumed by the larger experiment of Figure 7. This is a far larger experiment than the one in [30, 31], and aimed at testing two distinct hypotheses simulataniously: the one in the latter references about the Eutherian orders, and the far more general one about thge orders of the placental mammals (Eutheria, Metatheria, and Prototheria). Note also that adding the extra species from 20 to 24 is an addition that biologists are loath to do: both for computational reasons and fear of destabilizing a realistic phylogeny by adding even one more species to the computation. Furthermore, in the last mentioned references we used the special-purpose genome compressor GenCompress to determine the distance matrix, and the standard biological software MOLPHY package to reconstruct the phylogeny tree from the distance matrix. In contrast, in this paper we conduct a larger experiment than before, using just the general-purpose compressor bzip2 to obtain the distance matrix, and our new quartet tree reconstruction method to obtain the phylogeny tree—that is, our own Complearn package [9], used without any change in all the other experiments.

Marsupionta and Theria: The extant monophyletic divisions of the class Mammalia are the Prototheria (monotremes: mammals that procreate using eggs), Metatheria (marsupials: mammals that procreate using pouches), and Eutheria (placental mammals: mammals that procreate using placentas). The sister relationships between these groups is viewed as the most fundamental question in mammalian evolution [21]. Phylogenetic comparison by either anatomy or mitochondrial genome has resulted in two conflicting hypotheses: the gene-isolation-supported *Marsupionta hypothesis*: ((Prototheria, Metatheria), Eutheria) versus the morphology-supported *Theria hypothesis*: (Prototheria, (Methateria, Eutheria)), the third possiblity apparently not being held seriously by anyone. There has been a lot of support for either hypothesis; recent support for the Theria hypothesis was given in [21] by analyzing a large nuclear gene (M6P/IG2R), viewed as important across the species concerned, and even more recent support for the Marsupionta hypothesis was given in [18] by phylogenetic analysis of another sequence from the nuclear gene (18S rRNA) and by the whole mitochondrial genome.

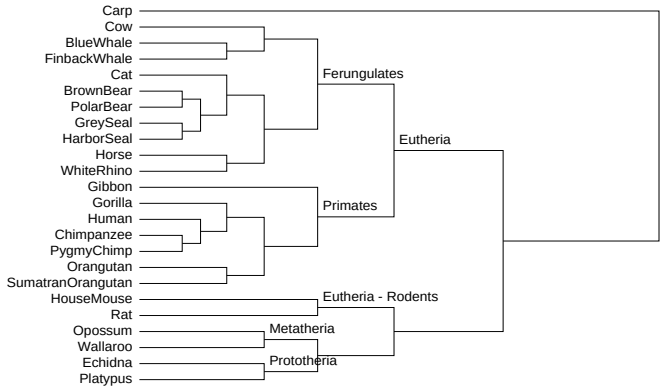

Figure 7: The evolutionary tree built from complete mammalian mtDNA sequences of 24 species, using the NCD matrix of Figure 9. We have redrawn the tree from our output to agree better with the customary phylogeny tree format. The tree agrees exceptionally well with the NCD distance matrix: $S(T) = 0.996$.

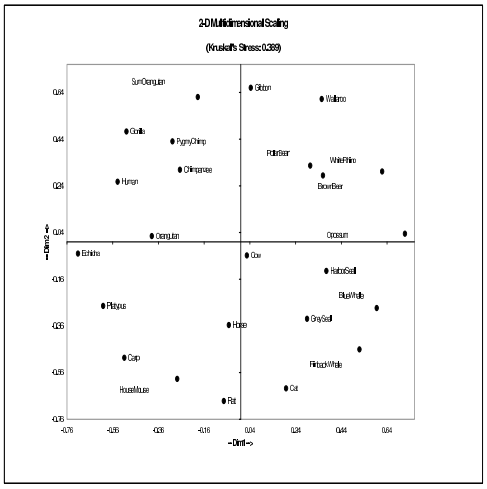

Figure 8: Multidimensional clustering of same NCD matrix (Figure 9) as used for Figure 7. Kruskal's stress-1 = 0.389.

	BlueWhale	BrownBear	Carp	Cat	Chimpanzee	Cow	Echidna	FinWhale	Gibbon	Gorilla	GreySeal	HarborSeal	Horse	HouseMouse	Human	Opossum	Orangutan	PolarBear	Platypus	PygmyChimp	Rat	SumOrang	Wallaroo	WhiteRhino
BlueWhale	0.005	0.906	0.943	0.897	0.925	0.883	0.936	0.616	0.928	0.931	0.901	0.898	0.896	0.926	0.920	0.936	0.928	0.929	0.907	0.930	0.927	0.929	0.925	0.902
BrownBear	0.906	0.002	0.943	0.887	0.935	0.906	0.944	0.915	0.939	0.940	0.875	0.872	0.910	0.934	0.930	0.936	0.938	0.937	0.269	0.940	0.935	0.936	0.923	0.915
Carp	0.943	0.943	0.006	0.946	0.954	0.947	0.955	0.952	0.951	0.957	0.949	0.950	0.952	0.956	0.946	0.956	0.953	0.954	0.945	0.960	0.950	0.953	0.942	0.960
Cat	0.897	0.887	0.946	0.003	0.926	0.897	0.942	0.905	0.928	0.931	0.870	0.872	0.885	0.919	0.922	0.933	0.932	0.931	0.885	0.929	0.920	0.934	0.919	0.897
Chimpanzee	0.925	0.935	0.954	0.926	0.006	0.926	0.948	0.926	0.849	0.731	0.925	0.922	0.921	0.943	0.667	0.943	0.841	0.946	0.931	0.441	0.933	0.835	0.934	0.930
Cow	0.883	0.906	0.947	0.897	0.926	0.006	0.936	0.885	0.931	0.927	0.890	0.888	0.893	0.925	0.920	0.931	0.930	0.929	0.905	0.931	0.921	0.930	0.923	0.899
Echidna	0.936	0.944	0.955	0.942	0.948	0.936	0.005	0.936	0.947	0.947	0.940	0.937	0.942	0.941	0.939	0.936	0.947	0.855	0.935	0.949	0.941	0.947	0.929	0.948
FinbackWhale	0.616	0.915	0.952	0.905	0.926	0.885	0.936	0.005	0.930	0.931	0.911	0.908	0.901	0.933	0.922	0.936	0.933	0.934	0.910	0.932	0.928	0.932	0.927	0.902
Gibbon	0.928	0.939	0.951	0.928	0.849	0.931	0.947	0.930	0.005	0.859	0.932	0.930	0.927	0.948	0.844	0.951	0.872	0.952	0.936	0.854	0.939	0.868	0.933	0.929
Gorilla	0.931	0.940	0.957	0.931	0.731	0.927	0.947	0.931	0.859	0.006	0.927	0.929	0.924	0.944	0.737	0.944	0.835	0.943	0.928	0.732	0.938	0.836	0.934	0.929
GreySeal	0.901	0.875	0.949	0.870	0.925	0.890	0.940	0.911	0.932	0.927	0.003	0.399	0.888	0.924	0.922	0.933	0.931	0.936	0.863	0.929	0.922	0.930	0.920	0.898
HarborSeal	0.898	0.872	0.950	0.872	0.922	0.888	0.937	0.908	0.930	0.929	0.399	0.004	0.888	0.922	0.922	0.933	0.932	0.937	0.860	0.930	0.922	0.928	0.919	0.900
Horse	0.896	0.910	0.952	0.885	0.921	0.893	0.942	0.901	0.927	0.924	0.888	0.888	0.003	0.928	0.913	0.937	0.923	0.936	0.903	0.923	0.912	0.924	0.924	0.848
HouseMouse	0.926	0.934	0.956	0.919	0.943	0.925	0.941	0.933	0.948	0.944	0.924	0.922	0.928	0.006	0.932	0.923	0.944	0.930	0.924	0.942	0.860	0.945	0.921	0.928
Human	0.920	0.930	0.946	0.922	0.667	0.920	0.939	0.922	0.844	0.737	0.922	0.922	0.913	0.932	0.005	0.949	0.834	0.949	0.931	0.681	0.938	0.826	0.934	0.929
Opossum	0.936	0.936	0.956	0.933	0.943	0.931	0.936	0.936	0.951	0.944	0.933	0.933	0.937	0.923	0.949	0.006	0.960	0.938	0.939	0.954	0.941	0.960	0.891	0.952
Orangutan	0.928	0.938	0.953	0.932	0.841	0.930	0.947	0.933	0.872	0.835	0.931	0.932	0.923	0.944	0.834	0.960	0.006	0.954	0.933	0.843	0.943	0.585	0.945	0.934
Platypus	0.929	0.937	0.954	0.931	0.946	0.929	0.855	0.934	0.952	0.943	0.936	0.937	0.936	0.930	0.949	0.938	0.954	0.003	0.932	0.948	0.937	0.949	0.920	0.948
PolarBear	0.907	0.269	0.945	0.885	0.931	0.905	0.935	0.910	0.936	0.928	0.863	0.860	0.903	0.924	0.931	0.939	0.933	0.932	0.002	0.942	0.940	0.936	0.927	0.917
PygmyChimp	0.930	0.940	0.960	0.929	0.441	0.931	0.949	0.932	0.854	0.732	0.929	0.930	0.923	0.942	0.681	0.954	0.843	0.948	0.942	0.007	0.935	0.838	0.931	0.929
Rat	0.927	0.935	0.950	0.920	0.933	0.921	0.941	0.928	0.939	0.938	0.922	0.922	0.912	0.860	0.938	0.941	0.943	0.937	0.940	0.935	0.006	0.939	0.922	0.922
SumOrangutan	0.929	0.936	0.953	0.934	0.835	0.930	0.947	0.932	0.868	0.836	0.930	0.928	0.924	0.945	0.826	0.960	0.585	0.949	0.936	0.838	0.939	0.007	0.942	0.937
Wallaroo	0.925	0.923	0.942	0.919	0.934	0.923	0.929	0.927	0.933	0.934	0.920	0.919	0.924	0.921	0.934	0.891	0.945	0.920	0.927	0.931	0.922	0.942	0.005	0.935
WhiteRhino	0.902	0.915	0.960	0.897	0.930	0.899	0.948	0.902	0.929	0.929	0.898	0.900	0.848	0.928	0.929	0.952	0.934	0.948	0.917	0.929	0.922	0.937	0.935	0.002

Figure 9: Distance matrix of pairwise NCD . For display purpose, we have truncated the original entries from 15 decimals to 3 decimals precision.

Experimental Evidence: To test the Eutherian orders simultaneously with the Marsupionta- versus Theria hypothesis, we added four animals to the above twenty: Australian echidna (*Tachyglossus aculeatus*), brown bear (*Ursus arctos*), polar bear (*Ursus maritimus*), using the common carp (*Cyprinus carpio*) as the outgroup. Interestingly, while there are many species of Eutheria and Metatheria, there are only three species of now living Prototheria known: platypus, and two types of echidna (or spiny anteater). So our sample of the Prototheria is large. The addition of the new species might be risky in that the addition of new relations is known to distort the previous phylogeny in traditional computational genomics practice. With our method, using the full genome and obtaining a single tree with a very high confi dence $S(T)$ value, that risk is not as great as in traditional methods obtaining ambiguous trees with bootstrap (statistic support) values on the edges. The mitochondrial genomes of the total of 24 species we used were downloaded from the GenBank Database on the world-wide web. Each is around 17,000 bases. The NCD distance matrix was computed using the compressor PPMZ. The resulting phylogeny, with an almost maximal $S(T)$ score of 0.996 supports anew the currently accepted grouping (Rodents, (Primates, Ferungulates)) of the Eutherian orders, and additionally the Marsupionta hypothesis ((Prototheria, Metatheria), Eutheria), see Figure 7. Overall, our whole-mitochondrial NCD analysis supports the following hypothesis:

$$\overbrace{(\underbrace{(primates, ferungulates)(rodents}_{\text{Eutheria}}, (Metatheria, Prototheria))}^{\text{Mammalia}}),$$

which indicates that the rodents, and the branch leading to the Metatheria and Prototheria, split off early from the branch that led to the primates and ferungulates. Inspection of the distance matrix shows that the primates are very close together, as are the rodents, the Metatheria, and the Prototheria. These are tightly-knit groups with relatively close NCD 's. The ferungulates are a much looser group with generally distant NCD 's. The intergroup distances show that the Prototheria are furthest away from the other groups, followed by the Metatheria and the rodents. Also the fi ne-structure of the tree is consistent with biological wisdom.

Hierarchical versus Flat Clustering: This is a good place to contrast the informativeness of hierarchical clustering with multidimensional clustering using the same NCD matrix, exhibited in Figure 9. The entries give a good example of typical NCD values; we truncated the number of decimals from 15 to 3 signifi cant digits to save space. Note that the majority of distances bunches in the range $[0.9,1]$. This is due to the regularities the compressor can perceive. The diagonal elements give the self-distance, which, for PPMZ, is not actually 0, but is off from 0 only in the third decimal. In Figure 8 we clustered the 24 animals using the NCD matrix by multidimenional scaling as points in 2-dimensional Euclidean space. In this method, the NCD matrix of 24 animals can be viewed as a set of distances between points in n-dimensional Euclidean space ($n \leq 24$), which we want to project into a 2-dimensional Euclidean space, trying to distort the distances between the pairs as little as possible. This is akin to the problem of projecting the surface of the earth globe on a two-dimensional map with minimal distance distortion. The main feature is the choice of the measure of distortion to be minimized, [16]. Let the original set of distances be $d_1,\ldots,d_k$ and the projected distances be $d'_1,\ldots,d'_k$. In Figure 8 we used the distortion measure *Kruskall's stress-1*,

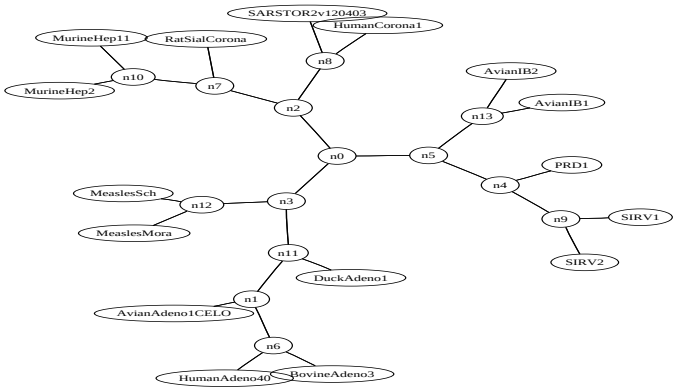

Figure 10: SARS virus among other virii. Legend: AvianAdeno1CELO.inp: Fowl adenovirus 1; AvianIB1.inp: Avian infectious bronchitis virus (strain Beaudette US); AvianIB2.inp: Avian infectious bronchitis virus (strain Beaudette CK); BovineAdeno3.inp: Bovine adenovirus 3; DuckAdeno1.inp: Duck adenovirus 1; HumanAdeno40.inp: Human adenovirus type 40; HumanCorona1.inp: Human coronavirus 229E; MeaslesMora.inp: Measles virus strain Moraten; MeaslesSch.inp: Measles virus strain Schwarz; MurineHep11.inp: Murine hepatitis virus strain ML-11; MurineHep2.inp: Murine hepatitis virus strain 2; PRD1.inp: Enterobacteria phage PRD1; RatSialCorona.inp: Rat sialodacryoadenitis coronavirus; SARS.inp: SARS TOR2v120403; SIRV1.inp: Sulfolobus virus SIRV-1; SIRV2.inp: Sulfolobus virus SIRV-2. $S(T) = 0.988$.

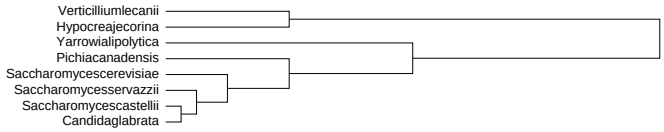

Figure 11: Dendrogram of mitochondrial genomes of fungi using NCD . This represents the distance matrix precisely with $S(T) = 0.999$.

[24], which minimizes $\sqrt{(\sum_{i\leq k}(d_i - d_i')^2)/\sum_{i\leq k} d_i^2}$. Kruskall's stress-1 equal 0 means no distortion, and the worst value is at most 1 (unless you have a really bad projection). In the projection of the NCD matrix according to our quartet method one minimizes the more subtle distortion $S(T)$ measure, where 1 means perfect representation of the relative relations between every 4-tuple, and 0 means minimal representation. Therefore, we should compare distortion Kruskall stress-1 with $1 - S(T)$. Figure 7 has a very good $1 - S(T) = 0.04$ and Figure 8 has a poor Kruskal stress 0.389. Assuming that the comparison is signifi cant for small values (close to perfect projection), we fi nd that the multidimensional scaling of this experiment's NCD matrix is formally inferior to that of the quartet tree. This conclusion formally justifi es the impression conveyed by the fi gures on visual inspection.

SARS Virus: In another experiment we clustered the SARS virus after its sequenced genome was made publicly available, in relation to potential similar virii. The 15 virus genomes were downloaded from The Universal Virus Database of the International Committee on Taxonomy of Viruses, available on the world-wide web. The SARS virus was downloaded from Canada's Michael Smith Genome Sciences Centre which had the fi rst public SARS Coronovirus draft whole genome assembly available for download (SARS TOR2 draft genome assembly 120403). The NCD distance matrix was computed using the compressor bzip2. The relations in Figure 10 are very similar to the defi nitive tree based on medical-macrobio-genomics analysis, appearing later in the New England Journal of Medicine, [25]. We depicted the fi gure in the ternary tree style, rather than the genomics-dendrogram style, since the former is more precise for visual inspection of proximity relations.

Analysis of Mitochondrial Genomes of Fungi: As a pilot for applications of the CompLearn Toolkit in fungi genomics reasearch, the group of T. Boekhout, E. Kuramae, V. Robert, of the Fungal Biodiversity Center, Royal Netherlands Academy of Sciences, supplied us with the mitochondrial genomes of *Candida glabrata, Pichia canadensis, Saccharomyces cerevisiae, S. castellii, S. servazzii, Yarrowia lipolytica* (all yeasts), and two fi lamentous ascomycetes *Hypocrea jecorina* and *Verticillium lecanii*. The NCD distance matrix was computed using the compressor PPMZ. The resulting tree is depicted in Figure 11. The interpretation of the fungi researchers is "the tree clearly clustered

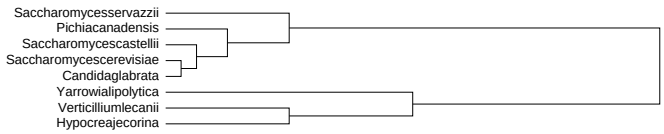

Figure 12: Dendrogram of mitochondrial genomes of fungi using block frequencies. This represents the distance matrix precisely with $S(T) = 0.999$.

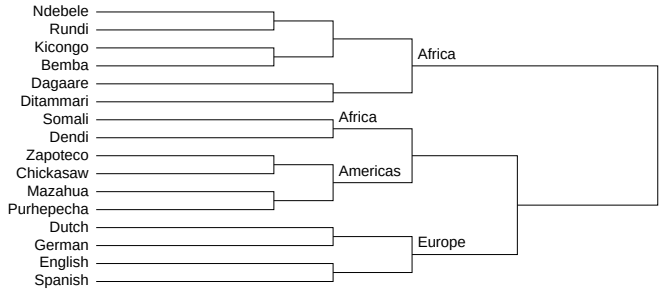

Figure 13: Clustering of Native-American, Native-African, and Native-European languages. $S(T) = 0.928$.

the ascomycetous yeasts versus the two filamentous Ascomycetes, thus supporting the current hypothesis on their classification (for example, see [26]). Interestingly, in a recent treatment of the Saccharomycetaceae, S. servazii, S. castellii and C. glabrata were all proposed to belong to genera different from Saccharomyces, and this is supported by the topology of our tree as well ([27])."

To compare the veracity of the NCD clustering with a more feature-based clustering, we also calculated the pairwise distances as follows: Each file is converted to a 4096-dimensional vector by considering the frequency of all (overlapping) 6-byte contiguous blocks. The l2-distance (Euclidean distance) is calculated between each pair of files by taking the square root of the sum of the squares of the component-wise differences. These distances are arranged into a distance matrix and linearly scaled to fit the range $[0, 1.0]$. Finally, we ran the clustering routine on this distance matrix. The results are in Figure 12. As seen by comparing with the NCD -based Figure 11 there are apparent misplacements when using the Euclidean distance in this way. Thus, in this simple experiment, the NCD performed better, that is, agreed more precisely with accepted biological knowledge.

8.2 Language Trees

Our method improves the results of [1], using a linguistic corpus of "The Universal Declaration of Human Rights (UDoHR)" [35] in 52 languages. Previously, [1] used an asymmetric measure based on relative entropy, and the full matrix of the pair-wise distances between all 52 languages, to build a language classification tree. This experiment was repeated (resulting in a somewhat better tree) using the compression method in [31] using standard biological software packages to construct the phylogeny. We have redone this experiment, and done new experiments, using the CompLearn Toolkit. Here, we report on an experiment to separate radically different language families. We downloaded the language versions of the UDoHR text in English, Spanish, Dutch, German (Native-European), Pemba, Dendi, Ndbele, Kicongo, Somali, Rundi, Ditammari, Dagaare (Native African), Chikasaw, Perhupecha, Mazahua, Zapoteco (Native-American), and didn't preprocess them except for removing initial identifying information. We used an Lempel-Ziv-type compressor *gzip* to compress text sequences of sizes not exceeding the length of the sliding window *gzip* uses (32 kilobytes), and compute the NCD for each pair of language sequences. Subsequently we clustered the result. We show the outcome of one of the experiments in Figure 13. Note that three groups are correctly clustered, and that even the subclusters of the European languages are correct (English is grouped with the Romance languages because it contains up to 40% admixture of words from Latin origine).

8.3 Literature

The texts used in this experiment were down-loaded from the world-wide web in original Cyrillic-lettered Russian and in Latin-lettered English by L. Avanasiev (Moldavian MSc student at the University of Amsterdam). The compressor used to compute the NCD matrix was bzip2. We clustered Russian literature in the original (Cyrillic) by Gogol, Dostojevski, Tolstoy, Bulgakov,Tsjechov, with three or four different texts per author. Our purpose was to

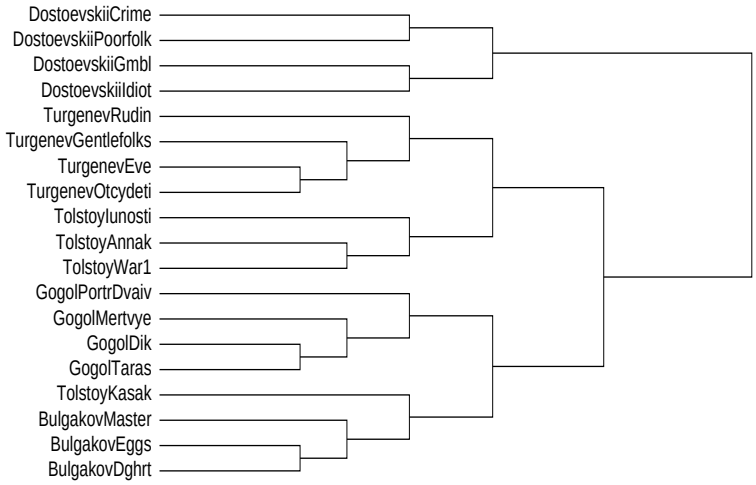

Figure 14: Clustering of Russian writers. Legend: I.S. Turgenev, 1818–1883 [Father and Sons, Rudin, On the Eve, A House of Gentlefolk]; F. Dostoyevsky 1821–1881 [Crime and Punishment, The Gambler, The Idiot; Poor Folk]; L.N. Tolstoy 1828–1910 [Anna Karenina, The Cossacks, Youth, War and Piece]; N.V. Gogol 1809–1852 [Dead Souls, Taras Bulba, The Mysterious Portrait, How the Two Ivans Quarrelled]; M. Bulgakov 1891–1940 [The Master and Margarita, The Fatefull Eggs, The Heart of a Dog]. $S(T) = 0.949$.

see whether the clustering is sensitive enough, and the authors distinctive enough, to result in clustering by author. In Figure 14 we see a perfect clustering. Considering the English translations of the same texts, in Figure 15, we see errors in the clustering. Inspection shows that the clustering is now partially based on the translator. It appears that the translator superimposes his characteristics on the texts, partially suppressing the characteristics of the original authors. In other experiments we separated authors by gender and by period.

8.4 Music

The amount of digitized music available on the internet has grown dramatically in recent years, both in the public domain and on commercial sites. Napster and its clones are prime examples. Websites offering musical content in some form or other (MP3, MIDI, ...) need a way to organize their wealth of material; they need to somehow classify their files according to musical genres and subgenres, putting similar pieces together. The purpose of such organization is to enable users to navigate to pieces of music they already know and like, but also to give them advice and recommendations ("If you like this, you might also like..."). Currently, such organization is mostly done manually by humans, but some recent research has been looking into the possibilities of automating music classification.

Initially, we downloaded 36 separate MIDI (Musical Instrument Digital Interface, a versatile digital music format available on the world-wide-web) files selected from a range of classical composers, as well as some popular music. The files were down-loaded from several different MIDI Databases on the world-wide web. The identifying information, composer, title, and so on, was stripped from the files (otherwise this may give a marginal advantage to identify composers to the compressor). Each of these files was run through a preprocessor to extract just MIDI Note-On and Note-Off events. These events were then converted to a player-piano style representation, with time quantized in 0.05 second intervals. All instrument indicators, MIDI control signals, and tempo variations were ignored. For each track in the MIDI file, we calculate two quantities: An *average volume* and a *modal note*. The average volume is calculated by averaging the volume (MIDI note velocity) of all notes in the track. The modal note is defined to be the note pitch that sounds most often in that track. If this is not unique, then the lowest such note is chosen. The modal note is used as a key-invariant reference point from which to represent all notes. It is denoted by 0, higher notes are denoted by positive numbers, and lower notes are denoted by negative numbers. A value of 1 indicates a half-step above the modal note, and a value of -2 indicates a whole-step below the modal note. The tracks are sorted according to decreasing average volume, and then output in succession. For each track, we iterate through each time sample in order, outputting a single signed 8-bit value for each currently sounding note. Two special values are reserved to represent the end of a time step and the end of a track. This file is then used as input to the compression stage for distance matrix calculation and subsequent tree search. To check whether any important feature of the music was lost during preprocessing, we played it back from the preprocessed files to verify whether

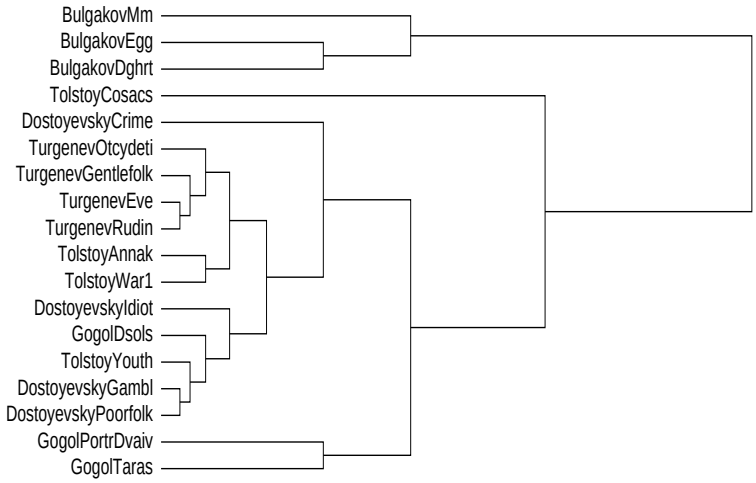

Figure 15: Clustering of Russian writers translated in English. The translator is given in brackets after the titles of the texts. Legend: I.S. Turgenev, 1818–1883 [Father and Sons (R. Hare), Rudin (Garnett, C. Black), On the Eve (Garnett, C. Black), A House of Gentlefolk (Garnett, C. Black)]; F. Dostoyevsky 1821–1881 [Crime and Punishment (Garnett, C. Black), The Gambler (C.J. Hogarth), The Idiot (E. Martin); Poor Folk (C.J. Hogarth)]; L.N. Tolstoy 1828–1910 [Anna Karenina (Garnett, C. Black), The Cossacks (L. and M. Aylmer), Youth (C.J. Hogarth), War and Piece (L. and M. Aylmer)]; N.V. Gogol 1809–1852 [Dead Souls (C.J. Hogarth), Taras Bulba ($\approx$ G. Tolstoy, 1860, B.C. Baskerville), The Mysterious Portrait + How the Two Ivans Quarrelled ($\approx$ I.F. Hapgood]; M. Bulgakov 1891–1940 [The Master and Margarita (R. Pevear, L. Volokhonsky), The Fatefull Eggs (K. Gook-Horujy), The Heart of a Dog (M. Glenny)]. $S(T) = 0.953$.

it sounded like the original. To the authors the pieces sounded almost unchanged. The compressor used to compute the NCD matrix of the genres tree, Figure 16, and that of 12-piece music set, Figure 17 is bzip2. For the full range of the music experiments see [10].

Before testing whether our program can see the distinctions between various classical composers, we first show that it can distinguish between three broader musical genres: classical music, rock, and jazz. This may be easier than making distinctions "within" classical music. All musical pieces we used are listed in the tables in the full paper (on the URL provided above). For the genre-experiment we used 12 classical pieces consisting of Bach, Chopin, and Debussy, 12 jazz pieces, and 12 rock pieces. The tree (Figure 16) that our program came up with has $S(T) = 0.858$. The discrimination between the 3 genres is reasonable but not perfect. Since $S(T) = 0.858$, a fairly low value, the resulting tree doesn't represent the NCD distance matrix very well. Presumably, the information in the NCD distance matrix cannot be represented by a dendrogram of high $S(T)$ score. This appears to be a common problem with large (> 25 or so) natural data sets. Another reason may be that the program terminatedi, while trapped in a local optimum. We repeated the experiment many times with almost the same results, so that doesn't appear to be the case. The 11-item subtree rooted at $n4$ contains 10 of the 12 jazz pieces, together with a piece of Bach's "Wohltemporierte Klavier (WTK)". The other two jazz pieces, Miles Davis' "So What," and John Coltrane's "Giant Steps" are placed elsewhere in the tree, perhaps according to some kinship that now escapes us (but may be identified by closer studying of the objects concerned). Of the 12 rock pieces, 10 are placed in the 12-item subtree rooted at $n29$, together with a piece of Bach's "WTK," and Coltrane's "Giant Steps," while Hendrix's "Voodoo Chile" and Rush "Yyz" is further away. Of the 12 classical pieces, 10 are in the 13-item subtrees rooted at the branch $n8, n13, n6, n7$, together with Hendrix's "Voodoo Chile," Rush's "Yyz," and Miles Davis' "So What." Surprisingly, 2 of the 4 Bach "WTK" pieces are placed elsewhere. Yet we perceive the 4 Bach pieces to be very close, both structurally and melodically (as they all come from the mono-thematic "Wohltemporierte Klavier"). But the program finds a reason that at this point is hidden from us. In fact, running this experiment with different compressors and termination conditions consistently displayed this anomaly. The small set encompasses the 4 movements from Debussy's "Suite Bergamasque," 4 movements of book 2 of Bach's "Wohltemperierte Klavier," and 4 preludes from Chopin's "Opus 28." As one can see in Figure 17, our program does a pretty good job at clustering these pieces. The $S(T)$ score is also high: 0.968. The 4 Debussy movements form one cluster, as do the 4 Bach pieces. The only imperfection in the tree, judged by what one would intuitively expect, is that Chopin's Prélude no. 15 lies a bit closer to Bach than to the other 3 Chopin pieces. This Prélude no 15, in fact, consistently forms an odd-one-out in our other experiments as well. This is an example of pure data mining, since there is some musical truth to this, as no. 15 is perceived as by far the most eccentric among

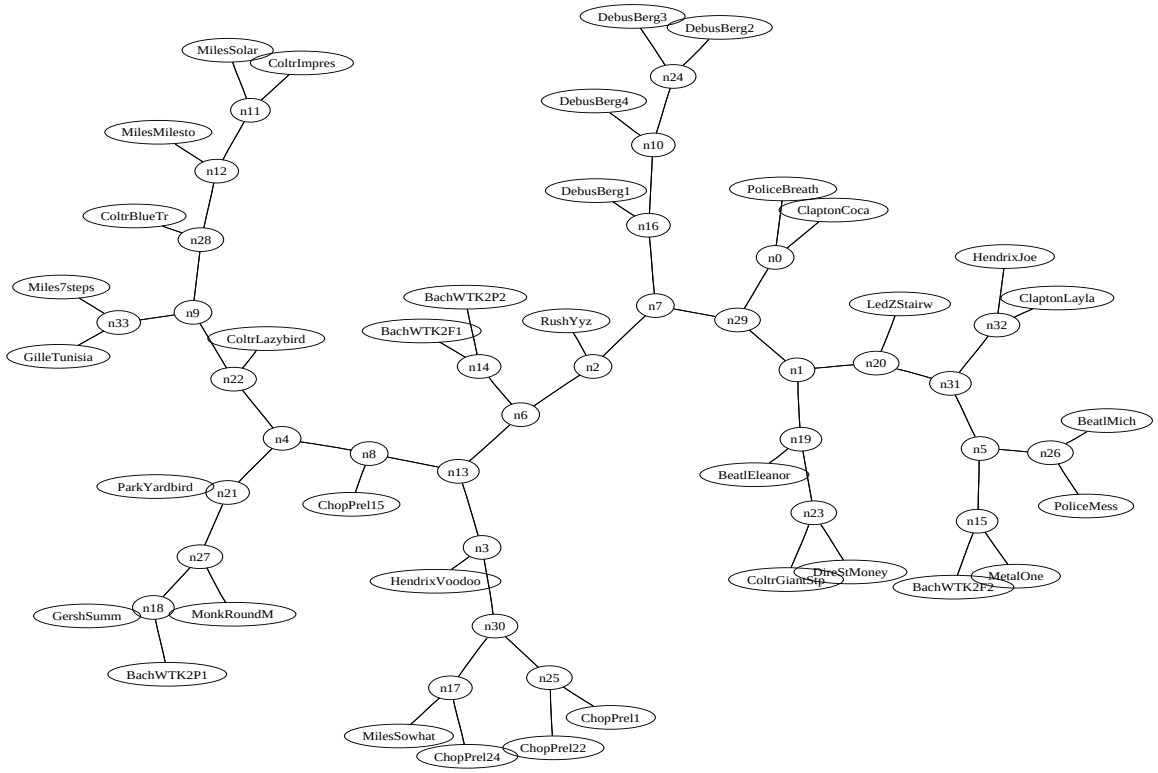

Figure 16: Output for the 36 pieces from 3 music-genres. Legend: 12 Jazz: John Coltrane [Blue Trane, Giant Steps, Lazy Bird, Impressions]; Miles Davis [Milestones, Seven Steps to Heaven, Solar, So What]; George Gershwin [Summertime]; Dizzy Gillespie [Night in Tunisia]; Thelonious Monk [Round Midnight]; Charlie Parker [Yardbird Suite]; 12 Rock & Pop: The Beatles [Eleanor Rigby, Michelle]; Eric Clapton [Cocaine, Layla]; Dire Straits [Money for Nothing]; Led Zeppelin [Stairway to Heaven]; Metallica [One]; Jimi Hendrix [Hey Joe, Voodoo Chile]; The Police [Every Breath You Take, Message in a Bottle] Rush [Yyz]; 12 Classic: see Legend Figure 17. $S(T) = 0.858$.

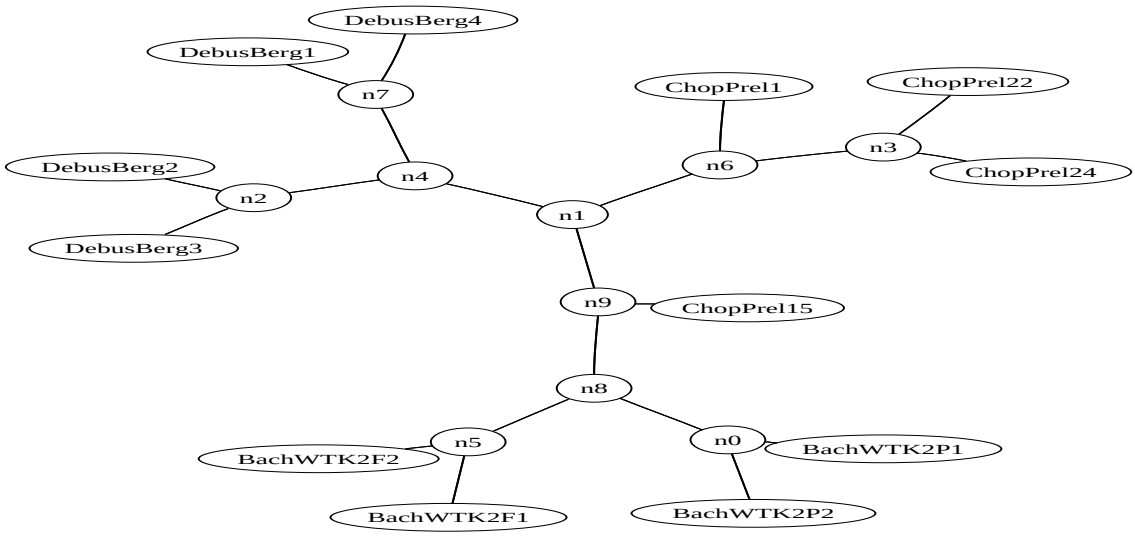

Figure 17: Output for the 12-piece set. Legend: J.S. Bach [Wohltemperierte Klavier II: Preludes and Fugues 1,2—BachWTK2{F,P}{1,2}]; Chopin [Préludes op. 28: 1, 15, 22, 24 —ChopPrel{1,15,22,24}]; Debussy [Suite Bergamasque, 4 movements—DebusBerg{1,2,3,4}]. $S(T) = 0.968$.

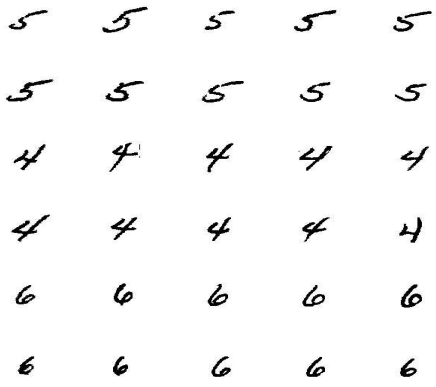

Figure 18: Images of handwritten digits used for OCR .

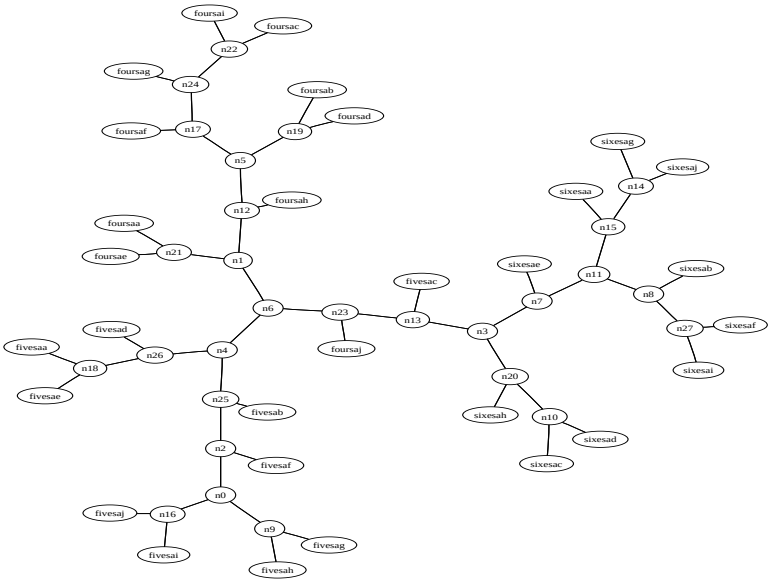

Figure 19: Clustering of the OCR images. $S(T) = 0.901$.

the 24 Préludes of Chopin's opus 28.

8.5 Optical Character Recognition

Can we also cluster two-dimensional images? Because our method appears focussed on strings this is not straightforward. It turns out that scanning a picture in raster row-major order retains enough regularity in both dimensions for the compressor to grasp. A simple task along these lines is to cluster handwritten characters. The handwritten characters in Figure 18 were downloaded from the NIST Special Data Base 19 (optical character recognition database) on the world-wide web. Each file in the data directory contains 1 digit image, either a four, five, or six. Each pixel is a single character; '#' for a black pixel, '.' for white. Newlines are added at the end of each line. Each character is 128x128 pixels. The NCD matrix was computed using the compressor PPMZ. The Figure 19 shows each character that is used. There are 10 of each digit "4," "5," "6," making a total of 30 items in this experiment. All but one of the 4's are put in the subtree rooted at $n1$, all but one of the 5's are put in the subtree rooted at $n4$, and all 6's are put in the subtree rooted at $n3$. The remaining 4 and 5 are in the branch $n23, n13$ joining $n6$ and $n3$. So 28 items out of 30 are clustered correctly, that is, 93%. In this experiment we used only 3 digits. Using the full set of decimal digits results in a lower clustering accuracy. However, we can use the NCD as a oblivious feature-extraction technique to convert generic objects into finite-dimensional vectors. We have used this technique to train a support vector machine (SVM) based OCR system to classify handwritten digits by extracting 80 distinct, ordered NCD features from each input image. In this initial stage of ongoing research, by our oblivious method of compression-based clustering to supply a kernel for an SVM classifier, we achieved a handwritten single decimal digit recognition accuracy of 85%. The current state-of-the-art for this problem, after half a century of interactive feature-driven classification research, in the upper ninety % level [34, 40]. All experiments are bench marked on the standard NIST Special Data Base 19 (optical character recognition database).

8.6 Astronomy

As a proof of principle we clustered data from unknown objects, for example objects that are far away. In [3] observations of the microquasar GRS 1915+105 made with the Rossi X-ray Timing Eplorer were analyzed. The interest in this microquasar stems from the fact that it was the first Galactic object to show a certain behavior (superluminal expansion in radio observations). Photonometric observation data from X-ray telescopes were divided into short time segments (usually in the order of one minute), and these segments have been classified into a bewildering array of fifteen different modes after considerable effort. Briefly, spectrum hardness ratios (roughly, "color") and photon count sequences were used to classify a given interval into categories of variability modes. From this analysis, the extremely complex variability of this source was reduced to transitions between three basic states, which, interpreted in astronomical terms, gives rise to an explanation of this peculiar source in standard black-hole theory. The data we used in this experiment made available to us by M. Klein Wolt (co-author of the above paper) and T. Maccarone, both researchers at the Astronomical Institute "Anton Pannekoek", University of Amsterdam. The observations are essentially time series, and our aim was experimenting with our method as a pilot to more extensive joint research. Here the task was to see whether the clustering would agree with the classification above. The NCD matrix was computed using the compressor PPMZ. The results are in Figure 20. We clustered 12 objects, consisting of three intervals from four different categories denoted as $\delta, \gamma, \phi, \theta$ in Table 1 of [3]. In Figure 20 we denote the categories by the corresponding Roman letters D, G, P, and T, respectively. The resulting tree groups these different modes together in a way that is consistent with the classification by experts for these observations. The oblivious compression clustering corresponds precisely with the laborious feature-driven classification in [3].

9 Conclusion

To interpret what the NCD is doing, and to explain its remarkable accuracy and robustness across application fields and compressors, the intuition is that the NCD minorizes all similarity metrics based on features that are captured by the reference compressor involved. Such features must be relatively *simple* in the sense that they are expressed by an aspect that the compressor analyzes (for example frequencies, matches, repeats). Certain sophisticated features may well be expressible as combinations of such simple features, and are therefore themselves simple features in this sense. The extensive experimenting above shows that even elusive features are captured.

A potential application of our non-feature (or rather, many-unknown-feature) approach is exploratory. Presented with data for which the features are as yet unknown, certain dominant features governing similarity are automatically

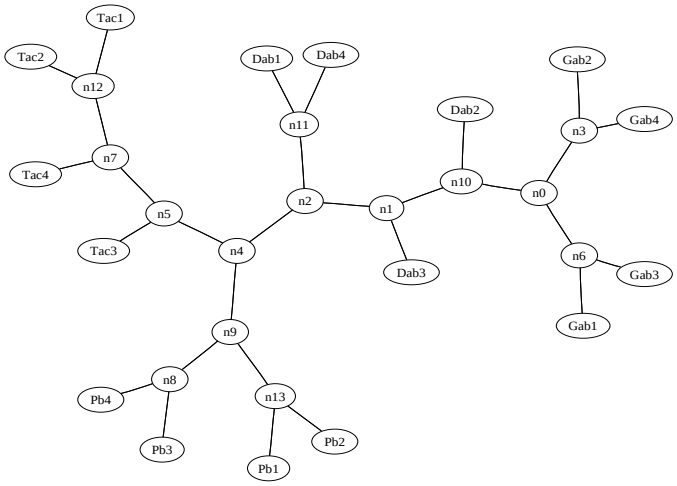

Figure 20: 16 observation intervals of GRS 1915+105 from four classes. The initial capital letter indicates the class corresponding to Greek lower case letters in [3]. The remaining letters and digits identify the particular observation interval in terms of finer features and identity. The T-cluster is top left, the P-cluster is bottom left, the G-cluster is to the right, and the D-cluster in the middle. This tree almost exactly represents the underlying NCD distance matrix: $S(T) = 0.994$.

discovered by the NCD . Examining the data underlying the clusters may yield this hitherto unknown dominant feature.

Our experiments indicate that the NCD has application in two new areas of support vector machine (SVM) based learning. Firstly, we find that the inverted NCD (1-NCD) is useful as a kernel for generic objects in SVM learning. Secondly, we can use the normal NCD as a feature-extraction technique to convert generic objects into finite-dimensional vectors, see the last paragraph of Section 8.5. In effect our similarity engine aims at the ideal of a perfect data mining process, discovering unknown features in which the data can be similar. This is the subject of current joint research in genomics of fungi, clinical molecular genetics, and radio-astronomy.

Acknowledgement

We thank Loredana Afanasiev, Graduate School of Logic, University of Amsterdam; Teun Boekhout, Eiko Kuramae, Vincent Robert, Fungal Biodiversity Center, Royal Netherlands Academy of Sciences; Marc Klein Wolt, Thomas Maccarone, Astronomical Institute “Anton Pannekoek”, University of Amsterdam; Evgeny Verbitskiy, Philips Research; Steven de Rooij, Ronald de Wolf, CWI; the referees and the editors, for suggestions, comments, help with experiments, and data; Jorma Rissanen and Boris Ryabko for discussions, John Langford for suggestions, Tzu-Kuo Huang for pointing out some typos and simplifications, and Teemu Roos and Henri Tirry for implementing a visualization of the clustering process.

References

[1] D. Benedetto, E. Caglioti, and V. Loreto. Language trees and zipping, *Physical Review Letters*, 88:4(2002) 048702.

[2] Ph. Ball. Algorithm makes tongue tree, *Nature*, 22 January, 2002.

[3] T. Belloni, M. Klein-Wolt, M. Méndez, M. van der Klis, J. van Paradijs, A model-independent analysis of the variability of GRS 1915+105, *Astronomy and Astrophysics*, 355(2000), 271–290.

[4] C.H. Bennett, P. Gács, M. Li, P.M.B. Vitányi, and W. Zurek. Information Distance, *IEEE Transactions on Information Theory*, 44:4(1998), 1407–1423.

[5] C.H. Bennett, M. Li, B. Ma, Chain letters and evolutionary histories, *Scientific American*, June 2003, 76–81.

[6] D. Bryant, V. Berry, P. Kearney, M. Li, T. Jiang, T. Wareham and H. Zhang. A practical algorithm for recovering the best supported edges of an evolutionary tree. *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms*, January 9–11, 2000, San Francisco, California, USA, 287–296, 2000.

[7] Y. Cao, A. Janke, P. J. Waddell, M. Westerman, O. Takenaka, S. Murata, N. Okada, S. Pääbo, M. Hasegawa, Conflict among individual mitochondrial proteins in resolving the phylogeny of Eutherian orders, *J. Mol. Evol.*, 47(1998), 307-322.

[8] X. Chen, B. Francia, M. Li, B. McKinnon, A. Seker, Shared information and program plagiarism detection, *IEEE Trans. Inform. Th.*, 50:7(2004), 1545–1551.

[9] R. Cilibrasi, The CompLearn Toolkit, 2003, http://complearn.sourceforge.net/ .

[10] R. Cilibrasi, P.M.B. Vitányi, R. de Wolf, Algorithmic clustering of music, *Computer Music Journal*, To appear. http://xxx.lanl.gov/abs/cs.SD/0303025

[11] G. Cormode, M. Paterson, S. Sahinalp, and U. Vishkin. Communication complexity of document exchange. In *Proc. 11th ACM–SIAM Symp. on Discrete Algorithms*, 2000, 197–206.

[12] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley & Sons, 1991.

[13] W. Chai and B. Vercoe. Folk music classification using hidden Markov models. *Proc. of International Conference on Artificial Intelligence*, 2001.

[14] M. Cooper and J. Foote. Automatic music summarization via similarity analysis, *Proc. IRCAM*, 2002.

[15] R. Dannenberg, B. Thom, and D. Watson. A machine learning approach to musical style recognition, *Proc. International Computer Music Conference*, pp. 344-347, 1997.

[16] R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification*, 2nd Edition, Wiley Interscience, 2001.

[17] M. Grimaldi, A. Kokaram, and P. Cunningham. Classifying music by genre using the wavelet packet transform and a round-robin ensemble. Technical report TCD-CS-2002-64, Trinity College Dublin, 2002. http://www.cs.tcd.ie/publications/tech-reports/reports.02/TCD-CS-2002-64.pdf

[18] A. Janke, O. Magnell, G. Wieczorek, M. Westerman, U. Arnason, Phylogenetic analysis of 18S rRNA and the mitochondrial genomes of wombat, Vombatus ursinus, and the spiny anteater, Tachyglossus acelaetus: increased support for the Marsupionta hypothesis, *J. Mol. Evol.*, 1:54(2002), 71–80.

[19] T. Jiang, P. Kearney, and M. Li. A Polynomial Time Approximation Scheme for Inferring Evolutionary Trees from Quartet Topologies and its Application. *SIAM J. Computing*, 30:6(2001), 1942–1961.

[20] E. Keogh, S. Lonardi, and C.A. Rtanamahatana, Toward parameter-free data mining, In: *Proc. 10th ACM SIGKDD Intn'l Conf. Knowledge Discovery and Data Mining*, Seattle, Washington, USA, August 22—25, 2004, 206–215.

[21] J.K. Killian, T.R. Buckley, N. Steward, B.L. Munday, R.L. Jirtle, Marsupials and Eutherians reunited: genetic evidence for the Theria hypothesis of mammalian evolution, *Mammalian Genome*, 12(2001), 513–517.

[22] M. Koppel, S. Argamon, A.R. Shimoni, Automatic catagorizing written texts by author gender, *Literary and Linguistic Computing*, To appear.

[23] A. Kraskov, H. Stögbauer, R.G. Adrsejak, P. Grassberger, Hierarchical clustering based on mutual information, 2003, http://arxiv.org/abs/q-bio/0311039

[24] J.B. Kruskal, Nonmetric multidimensional scaling: a numerical method, *Psychometrika*, 29(1964), 115–129.

[25] T.G. Ksiazek, et.al., A Novel Coronavirus Associated with Severe Acute Respiratory Syndrome, *New England J. Medicine*, Published at www.nejm.org April 10, 2003 (10.1056/NEJMoa030781).

[26] C.P. Kurtzman, J. Sugiyama, Ascomycetous yeasts and yeast-like taxa. In: *The mycota VII, Systemtics and evolution, part A*, pp. 179-200, Springer-Verlag, Berlin, 2001.

[27] C.P. Kurtzman, Phylogenetic circumscription of Saccharomyces, Kluyveromyces and other members of the Saccharomycetaceaea, and the proposal of the new genera Lachnacea, Nakaseomyces, Naumovia, Vanderwaltozyma and Zygotorulaspora, *FEMS Yeast Res.*, 4(2003), 233–245.

[28] P.S. Laplace, *A philosophical essay on probabilities*, 1819. English translation, Dover, 1951.

[29] M. Li, J.H. Badger, X. Chen, S. Kwong, P. Kearney, and H. Zhang. An information-based sequence distance and its application to whole mitochondrial genome phylogeny, *Bioinformatics*, 17:2(2001), 149–154.

[30] M. Li and P.M.B. Vitányi. Algorithmic Complexity, pp. 376–382 in: *International Encyclopedia of the Social & Behavioral Sciences*, N.J. Smelser and P.B. Baltes, Eds., Pergamon, Oxford, 2001/2002.

[31] M. Li, X. Chen, X. Li, B. Ma, P.M.B. Vitányi. The similarity metric, *IEEE Trans. Inform. Th.*, 50:12(2004), 3250- 3264.

[32] M. Li and P.M.B. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*, Springer-Verlag, New York, 2nd Edition, 1997.

[33] A. Londei, V. Loreto, M.O. Belardinelli, Music style and authorship categorization by informative compressors, Proc. 5th Triannual Conference of the European Society for the Cognitive Sciences of Music (ESCOM), September 8-13, 2003, Hannover, Germany, pp. 200-203.

[34] L.S. Oliveira, R. Sabourin, F. Bortolozzi, C.Y. Suen, Automatic recognition of handwritten numerical strings: A recognition and verification strategy, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24:11(2002), 1438–1454.

[35] United Nations General Assembly resolution 217 A (III) of 10 December 1948: Universal Declaration of Human Rights, http://www.un.org/Overview/rights.html

[36] A. Rokas, B.L. Williams, N. King, S.B. Carroll, Genome-scale approaches to resolving incongruence in molecular phylogenies, *Nature*, 425(2003), 798–804 (25 October 2003).

[37] D. Salomon, *Data Compression*, Springer-Verlag, New York, 1997.

[38] N. Saitou, M. Nei, The neighbor-joining method: a new method for reconstructing phylogenetic trees, *Mol. Biol. Evol.*, 4(1987), 406–425.

[39] P. Scott. Music classification using neural networks, 2001.
http://www.stanford.edu/class/ee373a/musicclassification.pdf

[40] Ø. D. Trier, A.K. Jain, T. Taxt, Feature extraction methods for character recognition—A survey, *Pattern Recognition*, 29:4(1996), 641–662.

[41] P.N. Yianilos, Normalized forms for two common metrics, NEC Research Institute, Report 91-082-9027-1, 1991, Revision 7/7/2002. http://www.pnylab.com/pny/

[42] A. C.-C. Yang, C.-K. Peng, H.-W. Yien, A.L. Goldberger, Information categorization approach to literary authorship disputes, *Physica A*, 329(2003), 473-483.

[43] G. Tzanetakis and P. Cook, Music genre classification of audio signals, *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.