Compression and Approximate Matching

L. ALLISON, D. POWELL AND T. I. DIX

School of Computer Science and Software Engineering, Monash University, Australia 3168
http://www.cs.monash.edu.au/~lloyd/tildeStrings

A population of sequences is called non-random if there is a statistical model and an associated compression algorithm that allows members of the population to be compressed, on average. Any available statistical model of a population should be incorporated into algorithms for alignment of the sequences and doing so changes the rank order of possible alignments in general. The model should also be used in deciding if a resulting approximate match between two sequences is significant or not. It is shown how to do this for two plausible interpretations involving pairs of sequences that might or might not be related. Efficient alignment algorithms are described for quite general statistical models of sequences. The new alignment algorithms are more sensitive to what might be termed 'features' of the sequences. A natural significance test is shown to be rarely fooled by apparent similarities between two sequences that are merely typical of all or most members of the population, even unrelated members.

Received April 4, 1998; revised January 28, 1999

1. INTRODUCTION

An alignment of two sequences shows how they could be related, i.e. how they can be matched, approximately. A particular model for relating sequences defines a cost or alternatively a score, to be optimized when finding an optimal alignment. Typical models focus on the similarities and differences between the sequences, trying to maximize the number of matches between characters, for example the longest common subsequence (LCS) [1], or to minimize the number of differences, for example edit distance [2, 3], or to do both in some combination [4]. There is generally a tacit assumption that the sequences themselves are random, in the sense of being incompressible, or that any non-randomness is limited to a simple skew in the frequency of use of individual characters; it is the aim of this paper to remove this assumption.

Figure 1 shows an example alignment of the sequences 'compression and approximate matching' and 'comprehension of appropriate meaning'. Spaces have been replaced by '/' to make them visible. The sequences have been padded out with a special null pseudo-character, denoted by '–', so that they have the same length. Matches are emphasized by a vertical bar, '|', between the matched characters. If the first sequence is considered to be the parent, although this is an arbitrary choice, then a column with a '–' in the top row represents an insertion and a column with a '–' in the bottom row represents a deletion. No column may contain two '–'s. A column with different (real) characters in the top and bottom rows represents a change, also known as a mismatch. A match is sometimes called a copy.

An alignment shows one way of editing one sequence into another using point, that is character, mutations. Each

```
compre--ssion/and/approximate/matching
||||||   |||||   ||||||   |||||    |||
comprehension/of-/appropriate/m-eaning
```

FIGURE 1. Example alignment.

operation can be given a cost or a score, and one can then search for an optimal alignment. Matches are good and are given low costs or high scores. Mutations are bad and are given high costs or low scores.

The costs (or scores) used in alignments can be viewed from an information theory point of view. There are two parts to such costs—those associated with the alignment itself [5] and those associated with the characters of the sequences. The two parts are often bundled together and not considered separately. Separating them allows one to consider and to model the process by which two sequences differ and also to consider and to model the population of sequences. This leads to more accurate models for approximate matching of sequences and to computer programs that give better results than otherwise.

Failing to correctly model the population of sequences can, for example, lead to a large number of false-positive close matches when comparing a new sequence against a collection. This has long been recognized as a problem in genetic databases, for example Fitch [6] discusses a correction based on randomizing sequences and retesting. Wootton [7] describes another method of masking-out regions of low 'compositional complexity' before looking for matches so as to reduce false positives. (Compositional complexity is there defined as the entropy of the multi-state distribution [8, 9] in sliding windows of some fixed length.)

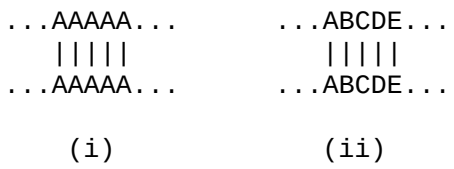

FIGURE 2. Low and high information content matches.

The 'xnu' program (Claverie and States [10]) also performs masking out. It can be used to preprocess protein search strings and masks out any tandem repeats with a period of four or less that have a specified degree of similarity. Masking-out is drastic because low complexity regions contain some, not zero, information and may themselves be of interest. The methods described here show how to attach the appropriate weight to low and high information sections of sequences for quite general notions of complexity. We also show that false-negatives are possible if the population of sequences is not correctly modelled.

We consider two general classes of model for matching pairs of sequences. In the first class, it is assumed that each sequence is a noisy observation of one hypothetical, real sequence. This corresponds to some laboratory situations, for example to the sequence assembly problem [11]. In the second class of model, there is no hypothetical real sequence; the objective is to align two sequences that are of equal standing. In both model classes, sequences are assumed to be non-random in the sense of being compressible; this is discussed further in the next section.

The non-randomness of sequences should be taken into account in their alignment. For example, assuming that runs of repeated characters are common, Figure 2 shows matches on low information content, i.e. compressible, subsequences 'AAAAA' and high information content, i.e. incompressible, subsequences 'ABCDE' that might occur in alignments.

It is intuitively obvious that partial match Figure 2i is good but that Figure 2ii is more significant. The subsequence 'ABCDE' is more surprising than 'AAAAA', for the assumed model, and the fact that it occurs in both sequences is strong evidence for those sections being related whereas 'AAAAA' is more likely to have occurred twice merely by chance. Consequently partial match Figure 2ii should give a greater benefit to an alignment than Figure 2i. If an alignment could only contain match Figure 2i or 2ii but not both, then it would be better for it to contain Figure 2ii. However, if runs of repeated characters were rare and alphabetically ordered runs were common, the roles would be reversed!

In the following, it is shown how to make precise the intuition from the above example. Sequence alignment algorithms that generalize common alignment algorithms by incorporating various models of sequence compression are described. The examples given use the DNA alphabet {ACGT}, but the methods are applicable to sequences over other alphabets.

2. COMPRESSION AND NON-RANDOMNESS

A population of sequences, F, is said to be 'non-random' if there is a statistical model M and an associated compression algorithm $m()$, such that (i) $m^{-1}(m(S)) = S$ and (ii) $|m(S)| < |S|$ on average, for sequences S chosen from F. $|x|$ denotes the length of a sequence x measured in bits. The definition of non-random may seem strange to some. On the one hand, the existence of such an $m()$ obviously shows the population to be non-random. Conversely, a population being non-random must mean that there is some systematic bias in its composition although this might be subtle. Any such bias will allow an algorithm to make better than chance predictions of the sequences and lead to a compression algorithm. We shall sometimes abuse notation and write about $m()$ as if it were the model.

The ability to compress sequences comes from repetition, pattern and structure within them. As a familiar example, 'u' almost invariably follows 'q' in English text ('Qantas' being an exception) which allows the u to be encoded at very little cost. It is now realized that data compression essentially consists of two parts—prediction and encoding. Shannon's theory of communication [12] shows that an item I of probability $P(I)$ is given a code word of length $-\log_2(P(I))$ bits in an optimal code. A predictor makes predictions about the next item and an encoder allocates code words on that basis. The decoder uses an identical predictor. Predictors are based on statistical models of the data and modelling is now seen to be the hardest part of data compression: the best model leads to the greatest compression. An arithmetic encoder [13] is capable of approaching the theoretical limits of encoding arbitrarily closely, even down to fractions of a bit. In this work we use predictors which yield probabilities of characters given the preceding context, i.e. $P(S[i] = x|S[1..i-1])$ where x ranges over the alphabet.

There is often prior knowledge about a population of sequences. For example, if DNA sequences were random, that is if each character (each base {ACGT}) was equally likely to occur in a position and if each position's value was independent of its neighbours' values, it would be impossible to do better than to allocate each character a two-bit code. However, low-order Markov models compress DNA to about 1.9 bits per character and more sophisticated models do rather better [14, 15, 16, 17]. Some sections of DNA are highly compressible. Many short sequences, such as TATA boxes, are known to occur in DNA. Runs of 'A's known as poly-A, $(AT)^+$ and $(CG)^+$ sequences also occur more often and at greater length than 'by chance'. As another example, the Alu's [18] are a family of sequences, each about 300 characters long, which occur hundreds of thousands of times in human DNA. A typical Alu sequence is about 87% similar to the consensus sequence.

If some pattern or structure amongst a population of sequences is known and if it is statistically significant then it can be used to compress the sequences. For example, if trying to compress DNA, one could have a special codeword to indicate the presence of an Alu sequence, which would

be followed by an encoding of the differences from the consensus Alu. The allocation of such an Alu code word would add very slightly to the coding of non-Alu stretches of DNA, so that one would lose out overall if Alu's did not occur sufficiently often and with sufficient fidelity to justify this action. In this way compression gives a significance test on any hypothetical pattern or structure that is claimed.

A model for a population of sequences might be based on prior knowledge or, failing that, one might choose a general, parameterized model and fit it to a particular data set. For example, one might assume a Markov model of some order k and estimate its parameters from the data.

3. ALIGNMENTS OF RANDOM SEQUENCES

Point-mutation models are commonly used to describe the mutation of sequences and the relationship of one sequence to another. The basic operations are to copy (alias match), change (alias mismatch), insert or delete a character. The generic dynamic programming algorithm (DPA) of Figure 3 can be used with a variety of point-mutation models and costs to find an optimal alignment of two sequences, S1 and S2, of lengths $\sim n$, in $O(n^2)$ time.

The DPA can be made to calculate the longest common subsequence (LCS), the edit distance and other functions of two sequences by making suitable choices for z, $c()$, $f()$ and $g()$. The popular linear and piecewise linear gap-costs [19] can also be included by storing extra state information in each entry, $M[i, j]$, and by making suitable choices of $c()$, $f()$ and $g()$.

For given probabilities, P(match), P(mismatch), P(insert) and P(delete), which must sum to 1.0, the following instantiation of the dynamic programming algorithm finds a most probable alignment of two given sequences:

```
z = 1                -- NB
g( ) = max( )
f( ) = *
c(x,x)   = P(match) * P(x)
c(x,y)   = P(mismatch) * P(x,y | x!=y)
c(x,"_") = P(delete) * P(x)
c("_",x) = P(insert) * P(x)
```

The actual alignment can be found by a trace-back of the choices made by $g()$. If P(match) etc. are not known in advance, an iterative expectation maximization (EM) approach [20, 21] can be used to estimate them.

Note that two different alignments are two different or exclusive hypotheses of how the two sequences are related, so the alignments' probabilities may legitimately be added. Furthermore, if the sequences are related under the point-mutation model then it must be by some alignment. So, summing all alignment probabilities gives the probability that the two sequences arose in some related but unspecified way; a specific alignment is a nuisance parameter if one just wants to know if the sequences are related, but not how [22]. The resulting DPA still runs in $O(n^2)$ time:

```
g( ) = +          -- z, f() & c() otherwise as above
P(S1 & S2 | related) = SUM[all alignments L] P(S1 & S2 & L)
```

The probabilities computed in variants of the DPA are very small for sequences of realistic lengths and would cause arithmetic underflow, so it is better to compute with their negative logs:

```
z = 0
either
  g( ) = min( )        -- find an optimal alignment
or
  g( ) = logplus( )    -- sum probability of all alignments
           where logplus(-log2(P1),-log2(P2)) = -log2(P1+P2)
f( ) = +
c(x,x)   = -log2(P(match) * P(x))
c(x,y)   = -log2(P(mismatch) * P(x,y | x!=y))
c(x,"-") = -log2(P(delete) * P(x))
c("-",x) = -log2(P(insert) * P(x))
```

This also corresponds to a coding or information theory interpretation and it is natural to imagine transmitting the sequences to a receiver and to speak of the 'message length' (after Shannon's communication theory) of an alignment [22], say. Note that all logs are taken to base two, giving 'bits' as the unit of measurement, in what follows.

The converse of the hypothesis that the two sequences are related forms a natural null-hypothesis that the sequences arose quite independently and its information content is simply

```
|S1| + |S2|
```

provided that $S1$ and $S2$ are random. On the other hand, if $S1$ and $S2$ are compressible by algorithm $m()$, the true information content of $S1$ and $S2$ under the null-hypothesis is

```
|m(S1)|+|m(s2)|
```

No hypothesis with a message length greater than that of the null-hypothesis is 'acceptable'.

4. GIVING A COST TO ALIGNMENTS OF COMPRESSIBLE SEQUENCES

Next consider the problem of calculating a natural cost for a given alignment of two sequences drawn from a population of non-random sequences. This allows two alignments to be compared so that we can say which one is better. The search problem, i.e. finding an optimal alignment, is also considered. It will be seen that almost any model of non-random sequences can be incorporated into assigning costs to alignments although not all models lead to efficient search algorithms.

Recall that a population of sequences, F, is non-random (or compressible or of low information content) if there is some model M with a compression algorithm $m()$ such that $|m(S)| < |S|$ on average for sequences drawn from F. We should use $m()$ in costing alignments for two reasons. Firstly, $m()$ is rightly used in the null-hypothesis to compress the individual sequences and will give it an unfair advantage over alignments unless they use $m()$ as well. Secondly, $m()$ should influence which alignments are more or less optimal as suggested in the introduction. There seem to be two starting points for trying to incorporate $m()$ into alignments, each of which leads to a class of models. The first class

```
M[0, 0] = z

for each i in 1 .. S1.length
   M[i,0] = f( M[i-1, 0  ], c(S1[i], "_" ) )   -- Boundary

for each j in 1 .. S2.length
   M[0,j] = f( M[0,   j-1], c("_",  S2[j]) )   -- conditions

for each i in 1 .. S1.length and j in 1 .. S2.length
   M[i,j] = g(f(M[i-1, j-1], c(S1[i], S2[j])), -- (mis)match
              f(M[i-1, j  ], c(S1[i], "_"  )), -- delete S1[i]
              f(M[i,   j-1], c("_",   S2[j]))) -- insert S2[j]
```

FIGURE 3. Generic DPA $M[i, j]$ = cost or score of $S1[1..i]\&S2[1..j]$.

has been introduced elsewhere by Powell *et al.* [23] and is sketched here for completeness; it also inspired the second class which is more general and new.

4.1. Model class 1: two observations of one real sequence

In this situation it is supposed that there is one real but unknown sequence, R, and that two observations of it, $S1$ and $S2$, are made by experiment. The experiments are not perfect and introduce experimental error so that the observed sequences are not identical to each other. One problem is to infer the real sequence, another is to find an optimal alignment of $S1$ and $S2$, and a third is to estimate the probability that $S1$ and $S2$ are in fact related in this way. Something close to this situation occurs within the sequence assembly problem [11].

Each of the given sequences (observations) may be the result of experimental error on an unknown real sequence. Being a real sequence, the latter would be compressible by $m()$. The observed sequences would be compressible under a related model although perhaps not to the same extent as R. However, if R were known, $S1$ and $S2$ could be encoded as a list of differences from R, for a given model of experimental errors. The information in $S1$ and $S2$ for a hypothetical R is

$$|m(R)| - \log_2(P(S1|R)) - \log_2(P(S2|R))$$

R is dealt with by the model for sequences, M, with its compression algorithm, $m()$, and $S1$ and $S2$ by the model of experimental error for which one of the variations on the edit distance serves well.

For a given model of sequences, a model of experimental error and an alignment also specifying R, the information content of this complete hypothesis can be readily calculated. Note that if only an alignment is required, R becomes a nuisance parameter and it is necessary to sum over all possible R's to avoid bias. It is possible to achieve this in $O(n)$ time, given an alignment, for finite-state models of sequences with one small approximation: if we limit R to having at most one character per column of the alignment, it is possible to sum the probabilities of $S1$ and $S2$ for all such R. The omission is that there could be an arbitrary number of characters in R that were missed in both observations $S1$ and $S2$—such an occurrence will have a very low probability, but a non-zero one. The converse, that both $S1$ and $S2$ contain erroneous observations of non-existent characters, is easily allowed. With the above reservation, a scan along the alignment can calculate $P(\text{columns}[1..i]|R[i] = \text{ch})$ conditional upon each possible character 'ch'. In model class 2, see later, sequence R is done away with altogether.

The search problems are (i) to find an optimal alignment together with R and (ii) to find an optimal alignment summing over all possible R's consistent with it. The solution is to treat R as a hidden variable by adding extra states to the entries of the DPA matrix $M[i, j]$ which represent costs conditional on a particular character value for the current position in R.

4.1.1. Implementation and tests

The DPA is modified for this first model class [23] by including extra state information in each entry $M[i, j]$. R-state, $I1$-state and $I2$-state refer to 'use the real sequence R', 'insert in $S1$' and 'insert in $S2$' respectively; see Figure 4. A transition to the R-state indicates that a character from R has been used; it may have been copied, changed or deleted in $S1$ and similarly in $S2$. A transition to $I1$ indicates that a character has been inserted into $S1$ relative to R, and similarly for $I2$. An insertion into sequence $S1$ does not affect sequences R or $S2$. To avoid any double-counting, we insist that any insertions into $S1$ occur before any adjacent insertions into $S2$, i.e. there is no direct transition from state $I2$ to $I1$.

The arcs represent possible transitions to the states of cell $M[i, j]$ from its north, west and north-west neighbours. If an optimal real sequence, R, is sought then the algorithm is made to choose the most probable incoming arc to each state. The probability of $S1$ and $S2$, given that they are related in this way through any R, can be found by summing the probabilities of paths into each state. An optimal alignment, over all R, can be found by a trace-back which chooses the maximum cell-to-cell probability 'flow'.

For the case of first-order Markov models of real

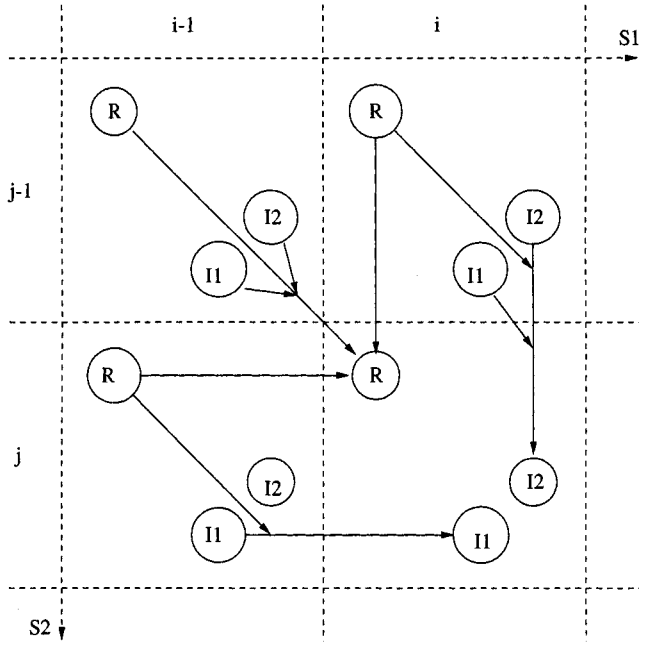

FIGURE 4. DPA for model class 1; arrivals in $M[i, j]$. Key: R, char from R; I1, ins′ char in $S1$; I2: ins′ char in $S2$.

sequences, the R-state contains information conditional upon each possible character value for the corresponding position of the R sequence, for example {ACGT} for DNA. Thus all character-to-character transitions in sequence R can be given their appropriate code-lengths. The $I1$- and $I2$-states do not involve a character from R. They carry forward information conditional on the last character of the R sequence for the resumption of R-state transitions. In practice the state information in $M[i, j]$ can be combined and reduced somewhat [23].

Tests were done using artificial sequences generated from a first-order Markov model, *MMg*. This choice was made purely on the basis of simplicity and we do not argue that *MMg* is a true model of any real population. 100 pairs of sequences of length 100 were generated from *MMg*. The sequences in a pair are completely unrelated except in coming from *MMg*. Of course they do share something, perhaps a kind of convergent evolution, but they are unrelated in the sense that they are not siblings and one was not derived from the other. Each pair of sequences was aligned using the algorithm described above under two methods: (i) using a uniform model for R, (ii) using *MMg* as the model for R. For each method, the difference in message lengths of the null-hypothesis and an optimal alignment, i.e. their $-$log odds-ratio, was calculated and the mean and the standard deviation of this quantity over 100 trials were recorded. A positive (versus negative) mean implies that the elements of the pairs are inferred to be related (versus unrelated) on average, under the method. The number of positive and negative inferences were also recorded. For each method, the null-hypothesis and the alignment used the same population model. Table 1 summarizes the results. We see that use of the incorrect, uniform model leads to a great

TABLE 1. Unrelated sequences, alignment with two models for R.

```
MMg, generating model:

           A C G T  odds for S[i+1]|S[i]
          +-------
        A |1 1 1 9
        C |9 1 1 9  i.e. sequences are
   S[i] G |9 1 1 9       AT-rich
        T |9 1 1 1

Analysis:
                    -log odds
method          null:alignment   inference
  (i) uniform    13 +/- 13,       11-, 89+
 (ii) MMg       -25 +/- 8,       100-,  0+
```

many false positives, but method (ii) correctly infers that the pairs are not related.

It is possible in principle to use other models for the R sequence, provided that they have finite memory, by appropriate elaboration of the state information in $M[i, j]$ while the algorithm's complexity remains $O(n^2)$ but with a worse constant. Second-order models are probably the upper limit in practice. Model class 2, below, is much more flexible in this regard. Linear gap-costs etc. for indels [5, 19] could also be included in model class 1 by elaboration of the $I1$- and $I2$-states.

4.2. Model class 2: averaging two sequences

In this situation there are two real sequences, $S1$ and $S2$, and we wish to know if and how they are directly related. They may have a common ancestor but this cannot be used in the way that R was used in model class 1 because it would merely give us three real sequences to compare and not simplify the problem.

The probability of two related sequences can be expressed in various ways:

```
P(S1&S2 | related) = P(S1) . P(S2 | S1, related)
                   = P(S2) . P(S1 | S2, related)
                   = SUM[all alignments L] P(S1 & S2 & L)
```

If the sequences are compressible, $S1$ by model $M1$ and algorithm $m1()$, $S2$ by model $M2$ and algorithm $m2()$, then $|m1(S1)| = -\log_2(P(S1))$ and $|m2(S2)| = -\log_2(P(S2))$, assuming that $m1()$ and $m2()$ give optimal compression under $M1$ and $M2$, respectively. Note that if $S1$ and $S2$ really are related then good choices for $M1$ and $M2$ cannot be very different but they might be instances of a parameterized model with slightly different parameter values for example. In order to code $S1$ and $S2$ one might code $S1$, compressing it with $m1()$, and then code $S2$ as the differences from $S1$, but this does not take advantage of $M2$. Alternatively one might code $S2$, compressing it with $m2()$, and then code $S1$, but now failing to take advantage of $M1$. These two schemes could even give

different message lengths which is unsatisfactory because we want an encoding in $-\log_2(P(S1\&S2|\text{related}))$ bits. We really want to blend $S1$ and $S2$ in some way. There is good reason to prefer a symmetric method which was realized for random sequences by coding them through their alignment. We may nevertheless sometimes write as though $S1$ is the 'parent' and $S2$ the 'child', although this is quite arbitrary and the roles can be reversed.

Informally, an alignment can be thought of as representing a fuzzy sequence. For example, the alignment of 'compression and approximate matching' with 'comprehension of appropriate meaning' in Figure 1 can be taken to represent the set of possible sequences beginning {compression..., comprension..., comprehssion..., etc.} These alternatives could be taken as possible values for R in model class 1 but removing any such explicit R leads to a more flexible method. The alternatives will be compressible to varying degrees. We try for a way of using their compressibility, without committing to any particular one, to transmit an alignment of $S1$ and $S2$. Also remember that if an $O(n^2)$ optimal-alignment algorithm is to be realized, it is desirable that the amount of work done for each character pair, $(S1[i], S2[j])$, be bounded by a constant and that there be no backtracking to revise the work at a later stage.

The following examples illustrate the various cases that the DPA examines in its central step:

```
           i           i           i           i
S1: .....ACG     ....ACG     .....ACG     ....AC-
         ||*         ||*         ||*         ||*
S2: .....ACG     ....ACT     .....AC-     ....ACT
           j           j           j           j

       match    mismatch    delete      insert
```

Omitting the characters of the two sequences, an alignment can be coded as a sequence over {match, mismatch, insert, delete}. These give the structure of the alignment while ignoring the sequences' character values. To encode $S1$ and $S2$ the characters must also be included. The case of insertions gives us a clue of how to do this. If the characters of $S2$ were independently generated then we could do no better than give a fixed code word (cost) to each character value regardless of the context in $S2$. However, it is assumed here that $S2$ is compressible and that the probability distribution for $S2[j]$ depends on $S2[1..j-1]$ so the probability of $S2[j]$ in an insertion can be estimated as

```
P(S2[j]|S2[1..j-1])
```

Deletions are similar to insertions and the probability of $S1[i]$ in a deletion can be estimated as

```
P(S1[i]|S1[1..i-1])
```

i.e. the probability and hence the code word length or cost for the character inserted in $S2$ (respectively deleted from $S1$) comes from the model $M2$ for $S2$ ($M1$ for $S1$). Note that these values depend only on $S1[1..i-1]$ and $S2[1..j-1]$.

In the case of a match, the character value $x = S1[i] = S2[j]$ need only be transmitted once; this is where the savings in good alignments come from, in transmitting both $S1[i]$ and $S2[j]$ for little more than the cost of one character. Now there are two sources of information about the character value: $M1$ and $M2$. A prediction of the value x can come from $M1$ or $M2$ or some combination of them. The following average is the simplest estimate of the character's probability that uses both sequence models in a sensible and symmetric way:

```
(P(x|S1[1..i-1])+P(x|S2[1..j-1]))/2
```

A mismatch also involves $S1[i]$ and $S2[j]$ but now they are known to differ. The characters' joint probability can be estimated by multiplying $P(S1[i]|S1[1..i-1])$ and $P(S2[j]|S2[1..j-1])$ after renormalizing the latter to account for $S2[j]$ differing from $S1[i]$. A different estimate comes from a mirror calculation, that interchanges the roles of $S1$ and $S2$. The average of these two alternatives is the simplest estimate of the characters' joint probability that uses both sequence models in a sensible and symmetric way:

```
(P(S1[i]|S1[1..i-1]) * P(S2[j]|S2[j]!=S1[i] & S2[1..j-1]) +
 P(S2[j]|S2[1..j-1]) * P(S1[i]|S1[i]!=S2[j] & S1[1..i-1]) ) / 2

= (P(S1[i]|S1[1..i-1]) * P(S2[j]|S2[1..j-1])) /
  (1/(1-P(S2[j]|S1[1..i-1])) + 1/(1-P(S1[i]|S2[1..j-1]))) / 2
```

When negative logs of the above probabilities are taken and added to those for insert, delete, match and mismatch, we have suitable code word lengths (costs) for use in the DPA. All that is required is the probability of each possible character value in each position along $S1$ conditional on the preceding values, similarly for $S2$, and these can either be derived from $M1$ and $M2$ during the DPA or may already be available if $S1$ and $S2$ have previously been compressed to assess the null-hypothesis. This can be done to all common variations of the DPA, for example for the most probable alignment, for the sum of all alignment probabilities, with linear or piecewise linear gap-costs.

4.2.1. Implementation and tests

The DPA was modified to accept the predictor components of compression algorithms as extra parameters—one for each of the input sequences $S1$ and $S2$. An iterative EM approach was used to estimate the probabilities of match, mismatch and indel from the data; it was assumed that inserts and deletes satisfy $P(\text{insert}) = P(\text{delete}) = P(\text{indel})/2$ although this is trivial to relax. Various models for sequences were implemented including 'uniform', i.e. two bits per character, and both non-adaptive and adaptive Markov models of order k, for various values of k. A non-adaptive model contains fixed parameter values which must either be 'common knowledge' or must be included as part of the length of any message that relies on the model. An adaptive model can use the sequence $S[1..i-1]$, for example, to estimate parameter values up to that point, when encoding $S[i]$.

Tests were done using the same 100 pairs of unrelated sequences generated from the first-order Markov model, *MMg*, of Section 4.1.1. Each pair of sequences was aligned with the modified DPA for model class 2 under three

TABLE 2. Unrelated sequences from *MMg*, alignment with three models.

method	-log odds null:alignment	inference
(i) uniform/uniform	13 +/- 13,	15-, 85+
(ii) MMg/MMg	-44 +/- 7,	100-, 0+
(iii) MM1/MM2	-30 +/- 7,	100-, 0+

methods: (i) using a uniform model for both $S1$ and $S2$; (ii) using *MMg* for both $S1$ and $S2$; (iii) using $MM1$ for $S1$ and $MM2$ for $S2$. Method (i) assumed the sequences to be incompressible. $MM1$ and $MM2$ were adaptive first-order Markov models. Their use in method (iii) amounted to knowing the general form of the model for the sequences but not the parameter values of the particular model instances, a situation common in the real world. (Adaptive models of this kind cannot be used with the algorithm for model class 1 because they adapt differently to alternative alignments and hypothetical sequences R.) For each method, the difference in message lengths of the null-hypothesis and an optimal alignment, i.e. their $-$log odds-ratio, was calculated and the mean and the standard deviation of this quantity over the 100 trials were recorded together with the numbers of positive and negative inferences under the method. The null-hypothesis and the optimal alignment used the same population models. Table 2 summarizes the results.

Method (i) implies that most pairs of completely unrelated sequences generated from *MMg* are related, with just 15 exceptions. It is possible to make method (i) firmly convinced either way merely by adjusting the parameters of model *MMg*. This has serious implications for matching a new sequence against a library of sequences or for ranking pairwise match scores in the sequence assembly problem for example. Method (ii) correctly believes that the pairs of sequences are unrelated, with the mean being six standard deviations away from the turn-over point. Method (iii) believes the pairs of sequences are unrelated, but with less certainty than method (ii); this is due to method (iii) having to estimate the parameters of the models $MM1$ and $MM2$ for each pair of sequences.

Note that many variations in the details of the general models and the alignment algorithm are possible. For example, (i) the models for sequences $S1$ and $S2$ can be required to be identical, (ii) non-adaptive or adaptive models [9] can be used, and so on. The cost of encoding parameter values must be included if models of differing complexities are to be compared fairly. (In the tests, the null-hypothesis and the optimal alignment used the same sequence models and any model costs cancel out in their $-$log odds-ratio in this case.)

The tests above show the possibility of alignment returning false positives unless the statistical properties of the population of sequences are taken into account. False negatives are also possible.

In a further series of tests, pairs of related sequences were generated and aligned using the three methods. A parent string was generated and mutated with certain probabilities of mutation to give two child sequences $S1$ and $S2$ which were then aligned. The parent string was generated according to some model, here either the uniform model or *MMg*. The relative probabilities of changes, insertions and deletions were kept at 2:1:1 while the overall probability of mutation was varied. A change necessarily changed a character to a different value. Inserted and changed characters were chosen from the probability distribution implied by the parent's generating model using the context at that point in the sequence. For changes, the original character was removed from the distribution which was then renormalized before a replacement character value was sampled. The child sequences were therefore from a very similar but not necessarily identical population to their parent unless the latter came from the uniform population. In the case that the parent came from *MMg*, the population of the children was learned as follows: 100 parents of length 1000 were generated from *MMg* and mutated by the appropriate amount, a first-order Markov model MMg' then being fitted to the child sequences. MMg' is similar to *MMg* but a little more uniform.

In these tests, a parent sequence of length 100 was generated from either the uniform population or from *MMg*. The parent was mutated by a certain amount to give $S1$ and $S2$. $S1$ and $S2$ were aligned by the modified DPA under three methods: (i) using a uniform model for both $S1$ and $S2$, (ii) using MMg' for both $S1$ and $S2$, (iii) using $MM1$ for $S1$ and $MM2$ for $S2$, where $MM1$ and $MM2$ were adaptive first-order Markov models. This was repeated 100 times for each combination of parent model and mutation level. The mean and standard deviation of the $-$log odds-ratio of the null-hypothesis and an optimal alignment were calculated, and also the number of positive (inferred related) and negative (not related) cases. The results are summarized in Table 3.

At first sight the results are disappointing. The pairs of sequences are siblings, truly related to each other by descent, so one might hope for 100 positive results when aligning sequences knowing the true model. In fact it appears that using the wrong model gives greater accuracy at inferring relatedness—the uniform model for MMg' data and MMg' for uniform data, with the order-one adaptive models $MM1/MM2$ falling in between—but this is an illusion. It must be remembered that the parent's children, $S1$ and $S2$, are separated from each other by two mutation stages. Thus a mutation level of 20%, parent to children, actually means that $S1$ and $S2$ differ by something approaching 40% mutation and this is very high for an alphabet of size four. It turns out that by 25% mutation, parent to children, the alignments obtained for uniform data are frequently no better than those found for unrelated sequences. For example, Deken [24] gives lower and upper bounds on the ratio of the length of an LCS of two unrelated strings to the length of the strings of 0.55 and 0.72 for uniform data and an alphabet of size four. In other words, by about 25% mutation from parent to children, it is often impossible to tell reliably

TABLE 3. Alignment with three models, two kinds of related strings.

		|---- True Model of related strings ----|			
DPA model	mutation rate	Uniform mutated log-odds	-ve	+ve	MMg' = MMg mutated log-odds	-ve	+ve
Uniform/ Uniform	15%	46+/-18,	0-,	100+	49+/-18,	1-,	99+
	20%	19+/-18,	14-,	86+	22+/-15,	10-,	90+
	25%	-2+/-14,	55-,	45+	3+/-16,	43-,	57+
MMg'/MMg'	15%	93+/-21,	0-,	100+	15+/-14,	13-,	87+
	20%	60+/-20,	0-,	100+	-7+/-13,	68-,	32+
	25%	35+/-16,	0-,	100+	-16+/-12,	92-,	8+
MM1/MM2	15%	58+/-18,	0-,	100+	26+/-16,	6-,	94+
	20%	31+/-17,	2-,	98+	5+/-13,	36-,	64+
	25%	10+/-14,	25-,	75+	-8+/-12,	72-,	28+

Key: -log odds Null:Alignment +/- S.D., -ve cases, +ve cases
Mutation: change:insert:delete probability ratios 2:1:1

that two sibling sequences from a uniform population are related, on the basis of an optimal alignment: the use of the true model is giving correct results.

A similar but stronger effect occurs when the parent comes from *MMg* and hence the children come from *MMg*′. *MMg* and *MMg*′ give AT-rich sequences and there is the further correlation between neighbouring characters, so the effective alphabet size is less than four (in the sense that sequences can be compressed to less than two bits per character). Deken's bounds for the LCS ratio are 0.76 to 0.86 for a uniform binary alphabet and 0.61 to 0.78 for a uniform ternary alphabet. Alignment knowing the true model, here *MMg*′, is again doing the right thing and it becomes impossible to reliably identify siblings in *MMg*′, on the basis of an optimal alignment, when they differ from their parent by between 15% and 20% mutation.

Table 3 therefore really shows that alignment using the wrong model often gives false positives, i.e. it claims sequences are related even when their optimal alignment is seen to be 'unacceptable' if the population's true characteristics are taken into account. (Note that a method which always stated that two strings were related, regardless of the evidence, would of course return 100 positive cases on all of the test data sets.) However, there is another side to this. The sets of *unique* sequences in the populations from the uniform model, from *MMg*′ and indeed from most other models are the same; it is only the sequences' prior probabilities that differ from population to population. One can occasionally get sequences typical of *MMg*′ under the uniform model and *vice versa*. Sequences typical of *MMg*′ are unusual under the uniform model (and *vice versa*); they have rare features and atypical characteristics. We can consider the uniform data to be a set of rare or unusual examples under *MMg*′ and we can consider the *MMg*′ data to be a set of rare or unusual examples under the uniform model. If two sequences sharing some rare features are given, one is more likely to be justified in calling them related than otherwise. So we also see that alignment using the wrong model is likely to give false negatives, particularly for sequences that are distantly related and unusual. Such sequences can however be shown to be related given knowledge of what is typical.

Note that the algorithm for model class 1 (Section 4.1) gave similar results to those in Table 3 for those cases where it could be applied, i.e. using the fixed uniform and *MMg* models for sequence R.

4.3. Discussion

It was pointed out earlier that an alignment and the null-hypothesis are both just hypotheses, that the difference in message lengths is their posterior $-\log$ odds-ratio and that this gives a significance test for alignments. This can also be related to the common practice of 'shuffling' for correcting alignment costs (or scores): two sequences, $S1$ and $S2$, are aligned and get a certain cost C. The sequences are then shuffled, giving $S1'$ and $S2'$ respectively, which are aligned to get a different cost C'. $S1$ and $S2$ are not considered to be related, and any alignment of $S1$ with $S2$ is discounted, unless C is 'significantly' better than C' [6]. $S1'$ is an 'average' sequence from the same zero-order Markov model as $S1$, similarly for $S2'$ and $S2$. Aligning $S1'$ with $S2'$ is like aligning two average strings from the same zero-order populations of sequences as $S1$ and $S2$, assuming that $S1$ and $S2$ do indeed come from zero-order populations. It is the case that for unrelated strings, an optimal alignment has a message length greater than the null-hypothesis with high probability [22]. Summing the probabilities of all alignments leads to a shorter message

length, but one that is still greater than that of the null-hypothesis. (We use the message length terminology here because it is difficult to combine scores over all alignments except in a coding or probabilistic framework.) So for zero-order populations of sequences, the message length (cost) of an optimal alignment of $S1'$ and $S2'$ is somewhat greater than that of the null-hypothesis in our framework but when allowing a suitable confidence interval around C' the effect of the correction is similar—for zero-order populations.

The present work uses $S1$ and $S2$ encoded under the null-hypothesis to judge the significance of any alignment etc. that claims to relate $S1$ to $S2$. This is efficient to compute if there are efficient compression algorithms for $S1$ and $S2$. The model for a population of sequences can be almost any model at all provided only that it has an associated compression algorithm, for example it can be based on order-k Markov models, on finite-state automata, on mixtures of these, etc. In contrast it is hard to see how to shuffle $S1$ and $S2$ while maintaining their statistical properties under an arbitrary model, although Fitch describes how to do this for first-order models and Altschul and Erickson [25] examine the problem of maintaining the frequencies of pairs and triples of characters. One could also fit an arbitrary model to $S1$ and to $S2$, generate a pair of random strings from these models, align the random strings to get a cost, C'', and repeat many times but this is inefficient.

A crucial limitation of shuffling is that it takes place outside the main alignment process. It cannot change the rank ordering of the alignments of two given sequences but only changes the criterion for acceptability, i.e. it might simply indicate that a bad alignment is bad while missing a good alignment. It was argued in the introduction, and is incorporated in the new algorithms, that the local information content should be able to change the rank ordering of alignments during the search and, in particular, change which is optimal.

Note that this paper is concerned with conventional order-preserving alignments. However, the 'approximate repeats model' of DNA sequences can be used to obtain a non-order-preserving alignment of two compressible sequences by using it to analyse their concatenation [17].

5. CONCLUSIONS

The non-randomness, or equivalently the compressibility, of sequences should be taken into account when finding an optimal alignment or when summing over all alignments to calculate the probability of their being related. Doing so weights high information content and low information content subsequences appropriately in selecting between alternative alignments and in choosing an optimal one. This can be seen to place more emphasis on matching features of sequences, although it is not a matter that some subsequence is or is not a feature—it is rather a matter of degree based on the information content. It also allows a fair comparison with the null-hypothesis, i.e. that the sequences are unrelated.

The idea of using compressibility in alignment has been made precise and two interpretations have been given, one of which corresponds to the sequence assembly problem and the other to the general alignment problem. An $O(n^2)$ alignment algorithm has been described for the first interpretation under finite-state models of non-random sequences in conjunction with finite-state models of mutation or relation. An $O(n^2)$ alignment algorithm has been described for the second interpretation provided either that there are sequence compression algorithms with complexity no worse than $O(n^2)$ or that the probability distributions over characters, $P(S[i] = x|S[1..i-1])$, are otherwise available on a position-by-position basis along each sequence.

Tests show that optimal alignment can give both false-positives, i.e. infer that unrelated sequences are related, and also false-negatives unless the properties of the population of sequences are taken into account during alignment. An obvious consequence is that the rank ordering of approximate matches of a sequence against a sequence database will probably be incorrect unless the properties of the sequence populations are considered.

Finally, this paper completely begs the question of what is a good statistical model of a population of sequences to use with the new alignment algorithms. It can do nothing else because the answer *must* be 'it just depends on the application area'. A model should be based on any prior knowledge that is available about the source of the sequences. There is recent work on the compression of DNA [15, 16, 17] for example and our strong preference is to use compression as the criterion by which to judge competing models. A general model should be used when there is little prior knowledge but the number of parameters of the model should be small compared to the amount of data.

ACKNOWLEDGEMENTS

The authors would like to thank Rohan Baxter, David Dowe, Peter Tischer, Chris Wallace and other members of the Central Inductive Agency for discussions on many varied topics that helped to inspire this work.

REFERENCES

[1] Hirschberg, D. S. (1975) A linear space algorithm for computing maximal common subsequences. *Comm. Assoc. Comp. Mach.*, **18**, 341–343.

[2] Levenshtein, V. I. (1965) Binary codes capable of correcting deletions, insertions and reversals. *Dokl. Akad. Nauk SSSR*, **163**, 845–848 (Engl. transl. (1966) *Sov. Phys.–Dokl.*, **10**, 707–710).

[3] Sellers, P. H. (1974) An algorithm for the distance between two finite sequences. *J. Combinatorial Theory*, **16**, 253–258.

[4] Needleman, S. B. and Wunsch, C. D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, **48**, 443–453.

[5] Allison, L. (1993) Normalization of affine gap costs used in optimal sequence alignment. *J. Theor. Biol.*, **161**, 263–269.

[6] Fitch, F. M. (1983) Random sequences. *J. Mol. Biol.*, **163**, 171–176.

[7] Wootton, J. C. (1997) Simple sequences of protein and DNA. In Bishop, M. J. and Rawlings, C. J. (eds), *DNA and Protein Sequences Analysis*, pp. 169–183. IRL Press, Eynsham, UK.

[8] Wallace, C. S. and Boulton, D. M. (1968) An information measure for classification. *Comp. J.*, **11**, 185–194.

[9] Boulton, D. M. and Wallace, C. S. (1969) The information content of a multistate distribution. *J. Theor. Biol.*, **23**, 269–278.

[10] Claverie, J.-M. and States, D. J. (1993) Information enhancement methods for large scale sequence analysis. *Comput. Chem.*, **17**, 191–201.

[11] Karp, R. M. (1993) Mapping the genome: some combinatorial problems arising in molecular biology. In *25th ACM Symp. Theory of Comp.*, pp. 278–285. ACM Press, New York.

[12] Shannon, C. E. (1948) A mathematical theory of communication. *Bell Syst. Technical J.*, **27**, 379–423, 623–656.

[13] Langdon, G. G. (1984) An introduction to arithmetic coding. *IBM J. Res. Development*, **28**, 135–149.

[14] Grumbach, S. and Tahi, F. (1994) A new challenge for compression algorithms: genetic sequences. *Inf. Proc. Management*, **30**, 875–886.

[15] Rivals, E. and Dauchet, M. (1997) Fast discerning repeats in DNA sequences with a compression algorithm. In *Proc. Genome Informatics Workshop*, Tokyo, pp. 215–226.

[16] Loewenstern, D. and Yianilos, P. N. (1997) Significantly lower entropy estimates for natural DNA sequences. In *Data Compression Conf., DCC '97*, pp. 151–160. IEEE Press, Los Alamitos, CA.

[17] Allison, L., Edgoose, T. and Dix, T. I. (1998) Compression of strings with approximate repeats. In *Proc. Intelligent Systems in Molecular Biology, ISMB98*, pp. 8–16. AAAI Press, Menlo Park, CA.

[18] Bains, W. (1986) The multiple origins of the human Alu sequences. *J. Mol. Evol.*, **23**, 189–199.

[19] Gotoh, O. (1982) An improved algorithm for matching biological sequences. *J. Molec. Biol.*, **162**, 705–708.

[20] Baum, L. E. and Eagon, J. E. (1967) An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model of ecology. *Bull. AMS*, **73**, 360–363.

[21] Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970) A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Stat.*, **41**, 164–171.

[22] Allison, L., Wallace, C. S. and Yee, C. N. (1992) Finite-state models in the alignment of macromolecules. *J. Molec. Evol.*, **35**, 77–89.

[23] Powell, D., Allison, L., Dix, T. I. and Dowe, D. L. (1998) Alignment of low information sequences. In *Proc. 4th Australasian Comp. Sci. Theory Symp., CATS '98*, Perth, pp. 215–229. Springer.

[24] Deken, J. (1983) Probabilistic behaviour of longest common subsequence length. In Sankoff, D. and Kruskal, J. B. (eds), *Time Warps, String Edits and Macromolecules*, pp. 359–362. Addison Wesley, Reading, MA.

[25] Altschul, S. F. and Erickson, B. W. (1985) Significance of nucleotide sequence alignment: a method for random sequence permutation that preserves dinucleotide and codon usage. *Mol. Biol. Evol.*, **2**, 526–538.