Banishing Bias from Consensus Sequences

Amir Ben-Dor[1], Giuseppe Lancia[2], Jennifer Perone[3], and R. Ravi[2]

[1] Dept. of Computer Science, Technion, Haifa 32000, Israel.
[2] GSIA, Carnegie Mellon University, Pittsburgh, PA 15213.
[3] New York University School of Medicine, New York.

Abstract. With the exploding size of genome databases, it is becoming increasingly important to devise search procedures that extract relevant information from them. One such procedure is particularly effective in finding new, distant members of a given family of related sequences: start with a multiple alignment of the given members of the family and use an integral or fractional consensus sequence derived from the alignment to further probe the database. However, the multiple alignment constructed to begin with may be biased due to skew in the sample of sequences used to construct it.

We suggest strategies to overcome the problem of bias in building consensus sequences. When the intention is to build a fractional consensus sequence (often termed a profile), we propose assigning weights to the sequences such that the resulting fractional sequence has roughly the same similarity score against each of the sequences in the family. We call such fractional consensus sequences *balanced profiles*. On the other hand, when only regular sequences can be used in the search, we propose that the consensus sequence have minimum maximum distance from any sequence in the family to avoid bias. Such sequences are NP-hard to compute exactly, so we present an approximation algorithm with very good performance ratio based on randomized rounding of an integer programming formulation of the problem. We also mention applications of the rounding method to selection of probes for disease detection and to construction of consensus maps.

1 Introduction

Efforts in genome projects have led to wide availability of genetic information in the form of nucleic acid and protein sequences. This is reflected by the exponentially increasing sizes of several sequence databases such as SwissProt [BB92].

Proteins are comprised of sequences of amino acids, or residues, which determine their structure and thereby function. Many proteins exist in many different organisms, or in several different forms in the same organism. The sets of these proteins are called families. These families exhibit structural, and therefore presumably sequential, similarities. The careful study of one protein in a family can provide information concerning the function, or predicted function of other proteins in that family. Likewise, the classification of a relatively unstudied protein into a well-defined family can offer insight toward its structure and function. In order to study families of proteins, the technique of multiple alignment is used.

Multiple alignments allow the simultaneous comparison of several sequences. Using blocks of conserved regions identified from multiple alignment data, a database can be probed for sequences with similarity with all of the sequences in the alignment. In order to do so, it is first necessary to derive from the multiple alignment a single sequence called the *consensus* which best represents all the aligned sequences, that can be used to search the database. Alternatively, a *profile* can be derived: this is a numerical representation of the multiple alignment [GME87], which, for each position and each residue, scores the likelihood that the given residue will appear at the indicated position in the protein alignment. Intuitively, we can think of the profile as a "fractional" consensus in which at any position, some fraction of each residue is present instead of just one.

Sequence collections are seldom a fair representation of the diversity of sequences consistent with a given protein structure conserved in a family. An example is the set of all currently available globin sequences, of which more than half are vertebrate α- and β-globins, while the remaining subfamilies are represented by much fewer sequences. A reason for this bias is that experimental sequence collections are not accurate representations of the diversity associated with the structure of a given protein in nature. This is partly by necessity since there are a select few organisms that are suitable for scientific research. Hence, biases in sequence databases tend to exist toward common experimental model organisms which are intensively sequenced.

A problem resulting from such biased database collections is that multiple alignments and consensus sequences that are built from such collections tend to be biased as well. For example, consider a multiple alignment constructed from a group of closely related proteins and one distant family member. The close proteins will dominate the consensus sequence and preclude retrieval of sequences which may bear more resemblance to the outlyer. Thus in the globin example, a profile or consensus built from a multiple alignment of all currently known globins would effectively recognize vertebrate globin sequences, while invertebrate globins would be poorly recognized.

The more intrinsic problem here is that the consensus built from a multiple alignment of a skewed sample from a family may not reflect the sequence homology of the family. This is what renders it ineffective in identifying distant members of the family that may be present in the database. In this paper, we describe two approaches to banish bias from consensus sequences for the two cases of constructing fractional and integral consensus sequences.

1.1 Weights for Unbiased Profile Construction

The traditional approach to correcting bias in constructing profiles from a multiple alignment is to weight the different sequences in the alignment differently in constructing the profile. A plethora of weighting schemes have been proposed in the literature [ACL89, GSC94, HH94, THG94, SA90, LXB94, EMD95, KM95]. The basic idea is to emphasize under-represented sequences by giving them high weights, while de-emphasizing over-represented sequences by giving them low weights. It is an open problem to determine a system of weights that results in the profile that can be used to search the database most effectively for biologically relevant signals. In Section 2 we discuss some of the existing weighting schemes.

We propose a new method for sequence weighting whose goal is to yield a profile that has roughly the same similarity score to each of the sequences in the alignment. We call a profile of this type a *balanced profile* and the problem of determining the corresponding weights will be called the Balanced Profile Weight Assignment Problem. For this problem we outline a simple iterative algorithm which converges to the desired weights. Preliminary experiments with an implementation of this algorithm indicate that this method may be more effective than an unweighted profile construction, especially when the alignment is composed of several divergent sequences. We elaborate on this in Section 3.

1.2 Unbiased Consensus Sequences

A similar fairness problem arises also when building the (integral) consensus sequence. One way to define a consensus sequence is to require that it minimizes its total distance from the sequences of the alignment (sum-of-pairs criterion), but this objective is biased toward overrepresented sequences. To overcome this bias, we define the consensus as the sequence whose maximal distance from any of the sequences in the alignment is minimum. Under this definition, the problem of determining the consensus turns out to be NP-hard. In this paper we model this consensus problem as an integer programming problem and give an approximation algorithm based on randomized rounding applied to its fractional relaxation.

1.3 Two applications of randomized rounding

Consider a problem arising in the design of probes for disease detection. The probes work by hybridizing with complementary strands of sufficient similarity. To design such probes to be specific for a particular strain of bacteria, we would like the probe sequence to be as close as possible to the genetic sequence from this strain while staying as far away as possible (in Hamming distance) from the sequences of all the other strains. We propose approximation algorithms for finding near-optimal probes by applying the rounding mentioned earlier.

Another application is to the construction of a consensus map from a variety of physical maps, all of which identify the location of the same set of markers linearly along the same fragment of DNA. The construction of a consensus map that is unbiased against any skew in the input data can be formulated as an integer program. Applying randomized rounding gives good approximate solutions to this problem as well.

The rest of the paper is organized as follows. In section 2 we describe some of the existing schemes for weighting sequences in an alignment and address the Balanced Profile Weight Assignment Problem. Section 3 reports computational results of database search with our method as opposed to the unweighted one. Section 4 describes the randomized rounding approximation algorithm for the consensus problem. Finally in section 5 we outline extensions of the randomized rounding technique to designing probes for disease detection and to physical map construction.

2 Profiles and Weighting Schemes

Let $\Sigma = \{\sigma_1, \ldots, \sigma_{|\Sigma|}\}$ be a finite alphabet (in particular, we can take Σ to be the set of 20 amino acids). We will consider elements of Σ^n, called *sequences.* Let $d : \Sigma \times \Sigma \longrightarrow \mathcal{R}$ be a distance function (e.g., for amino-acids d = PAM-250). We generalize d to sequences in Σ^n in the natural way, i.e., $d(x, y) = \sum_{i=1}^{n} d(x_i, y_i)$. A *weighted multiple alignment* is a vector of k sequences, i.e. a $k \times n$ matrix over $\Sigma \cup \{-\}$ (where $-$ denotes the blank character), together with a set $w_1, \ldots, w_k$ of weights for the sequences.

Given a multiple alignment, a *profile* is an $n \times |\Sigma|$ real–valued matrix P. Each entry $P[i, j]$ of P scores the likelihood with which σ_j is the i–th symbol of the sequences in the alignment (blanks are traditionally excluded from the profile). A profile can be constructed for a group of k aligned sequences each of equal length n using Profilemake, Genetic Computer Group's profile building tool [GCG94]. Profiles can also reflect values for any gaps which may appear in the alignment; however, gaps will not be discussed here (see [GME87] for further information). A database can then be searched with a profile by using the GCG's procedure Profilesearch [GCG94], which implements an alignment algorithm by Smith and Waterman, based on dynamic programming [SW81].

When using Profilemake, the scores at position r, character σ, are defined as follows:

$$P[r, \sigma] = \sum_{j=1}^{|\Sigma|} W_j \, s(\sigma_j, \sigma) \tag{1}$$

where s is the Dayhoff similarity score [DBH83]. The weight W_j depends on the number of occurrences of each type of residue at each alignment position and the sum of the weighted number of sequences. Profilemake can use either linear weighting:

$$W_j = \frac{\sum_{i=1}^{k} w_i \delta_{i,j}}{\sum_{i=1}^{k} w_i}$$

or logarithmic weighting:

$$W_j = \frac{\ln\left[1 - \sum_{i=1}^{k} w_i \delta_{i,j} \Big/ \left(1 + \sum_{i=1}^{k} w_i\right)\right]}{\ln\left[1 \Big/ \left(1 + \sum_{i=1}^{k} w_i\right)\right]}$$

where $\delta_{i,j}$ is equal to 1 if sequence i has residue σ_j at the current position, and is equal to 0 otherwise. As expected with either of these methods the weight will be zero if the given amino acid does not occur in the alignment position, and it will be one if it is the only amino acid which appears at that position.

Weights assigned to each sequence affect these values. In the unweighted system, all weights w_i are equal, so the contribution to W_j is the same for all sequences. In this case, the W_j value calculated for a residue σ_j is simply the fraction of input sequences that contain residue σ_j at the current position. When different weights are assigned to sequences, each sequences contribution to W_j is scaled by its weight. Sequences can be accentuated or de-emphasized to reflect closely related sequences by varying these weights. Intuitively, weights on distant sequences should be larger than the weights of the closely related sequences. Strategies to determine appropriate weights are discussed next.

While there is a consensus about the necessity of weighting sequences when searching with multiple alignments, there is considerable debate concerning what weighting method should be used. Many of the differences in these weighting systems are based on differing opinions on the problem as well as defining correct behavior. Some of the existing weighting techniques are the following (see also [VS93]):

- *Pairwise Distance:* A sequence weight is set to be equal to the sum of the distances from this sequence to all other sequences in the alignment. The idea is that a far-off sequence which is under-represented in the population, will have higher weight than a single sequence in a cluster of closely related sequences, thus correcting the bias. If D is the square matrix of pairwise distances between the sequences, we have $w = D\mathbf{1}$ for this method, where $\mathbf{1}$ is a vector with 1 in each component.
- *Voronoi Weighting:* This method, introduced by Sibbald and Argos [SA90], relies on constructing a Voronoi diagram from the sequences, based on pairwise distances [SA90]. A hypothetical population of sequences is built using the information from the real sequence alignment. A Voronoi diagram for this population is constructed using the input sequences as Voronoi centers. The weight for each input sequence is proportional to its Voronoi volume, i.e. the volume of the Voronoi polygon occupied by the sequence. Instead of generating all sequences, estimates of this volume can be obtained by random sampling.
- *Weighting by Phylogeny:* Altschul, Carroll, and Lipman describe a method for assigning weights based on an implied evolutionary relationship among the given sequence set [ACL89]. Using a tree constructed with all sequences in an alignment, weights can be determined by inverting a matrix of variances and covariances between pairs of sequences that is inferred from the tree.
- *Balanced Profiles:* The notion of balanced profiles appears in the work of Vingron and Sibbald [VS93], where they also draw a parallel between this scheme and that of Altschul et al. mentioned above [ACL89]. Even though the sequence weights are defined in exactly the same way as we do here, the way in which the weights are used in their method to compute the profile

is different from ours. As mentioned earlier, in our method, the profile at each position is computed using a combination of the weights as well as the underlying distance function (see Equation 1), while in Vingron and Sibbald the profile is defined simply as the weighted sum of the sequences in the alignment (i.e. $P[i, \sigma]$ is the sum of the weights for the sequences having residue σ in column i). With their definition, the weights yield a balanced profile if $Dw = \lambda \mathbf{1}$ so that, if D is invertible, w can be found by solving $w = \lambda D^{-1}\mathbf{1}$. This objective is pursued in Vingron and Argos [VA89] where the weights of sequences far from the (unweighted) profile are increased thus moving the weighted profile away from nearby sequences and toward distant ones.

– *Maximum Discrimination:* This probabilistic method introduced by Eddy, Mitchison and Durbin [EMD95], uses Hidden Markov Model to model the protein family (based on the given multiple alignment). The objective is to find a HMM that maximizes the probability that all of the sequences participating in the multiple alignment will be produced by the model (as opposed to being produced by a random model). The following gradient descent training method is proposed: Find the sequence (or sequences) with the lowest score, and change the model probabilities so as to better recognize those sequences (this method is called the Maxmin algorithm in [EMD95]). We remark that in our algorithm, a very similar training method is used; However, no probabilistic modeling of the multiple alignment is employed and hence our algorithm is much simpler and faster.

2.1 The Balanced Profile Weight Assignment Problem

The main utility of a sequence weighting scheme in building profiles is its potential to correct for bias in the input sequences. Motivated by this intended application of weighting schemes, we define a *balanced profile* to be one with almost identical similarity to all sequences participating in its construction. In other words, all sequences are chosen with the same intensity by this profile. The problem of weighting the sequences to build a profile then becomes one of setting the weights so that the resulting profile is balanced.

A profile can be thought of as a sequence of weighted residuals. A natural way to define the similarity score of a profile versus a target sequence is, by linearity, to compute the weighted sum, for all the positions in the sequence, of the similarities of residues in the profile and that in the target. When this scoring function is used, computing the weights which yield a balanced profile would simply amount to solving a system of linear equations in the variables $\{w_i\}$ [4]. However, this method can not be applied to the scoring function used by PROFILESEARCH, since it is not additive over the columns (e.g. the score for a run of k gaps is not the same as k times the score of one gap).

We propose an algorithm which is independent from the details of the scoring function adopted in PROFILESEARCH, or in any other scoring procedure to be

[4] This remark also appears in the article by Vingron and Sibbald[VS93]

used. Our algorithm is iterative and starts by giving equal weights to all the sequences. At each iteration the profile is computed using the current weights. If the profile is not balanced, the weights are updated and a new iteration is executed. Preliminary computational experiments with the scoring function used by Profilesearch have shown that the algorithm convergences on the average very fast (under 20 iterations) to the final weights. A formal statement of the procedure follows.

```
Repeat
  1. Build a profile from the multiple alignment (initially, all weights are equal).
  2. Score this profile against the sequences in the alignment.
  3. If the profile is not balanced
     then change the weight of either the sequences with the highest or lowest
     score.
until the profile is balanced.
```

In order to choose which weight to change (Step 3), the following procedure is followed. Define $score_{mid} = (score_{max} + score_{min})/2$. Determine how many scores lie on either side of $score_{mid}$. If there are fewer scores below $score_{mid}$, raise the weight of all sequences with the lowest score by δ, and if there are fewer weights above $score_{mid}$, then lower the scores of the sequences of the highest score by δ, where $\delta = (score_{max} - score_{min})/score_{max}$. The choice of δ reflects the dependence on the step size on the current spread of scores, moving rapidly when the spread is high and more carefully otherwise.

In order to prevent negative weights, if the weight of the sequences with the scores to be lowered is below a threshold, simply increase the lowest scoring sequence's weight instead. The iterations are stopped when all scores are within some specified error bound of each other, i.e. roughly similar. This method of weight changing assumes that increasing the weight of one sequence while holding everything else constant, will increase the amount by which the profile reflects that sequence compared to the others. This is true for both the linear and logarithmic weighting schemes used by Profilemake.

3 Search Results with Balanced Profiles

Balanced profiles were constructed for protein blocks and searched against SwissProt. The results of these searches are compared to searches with profiles of the same blocks constructed using equal weights. The data for these experiments were provided by Henikoff and Henikoff, which they used to calibrate their weight setting scheme against an equal-weights model [HH94]. Known protein families were extracted from Prosite, and subsets of these families were used to construct blocks with Protomat [Bai92]. Weights were assigned to sequences with the balanced profile scheme, and profiles were constructed from these blocks.

In order to compare the balanced profile search results to the ones with equal-weights, the Profilesearch results are compared to the sequences known to be in the protein family of the block subset. Thirty-seven blocks (corresponding to as many families) were arbitrarily chosen from the complete data set of Henikoff and Henikoff. Thirty seven pairs of profiles were constructed from these blocks using the balanced weights and equal weights for each case, and these were used to search the database for members of the corresponding blocks. The overview of the computational results is presented in Table 1.

Number of Sequences	Equal Weights	Balanced Profile	Ties
0-50	6	4	27
51-100	2	4	31
101-200	4	3	30
201-300	3	1	33
301-400	0	1	36
401-500	0	1	36
501-1000	4	3	30
1001-5000	3	2	32
over 5000	0	1	36
overall	4	4	29

Table 1. Overview of results - the number of wins for each scheme and the ties are shown for each of the ranges. E.g., among the 37 tests, Equal weight profiles recovered more sequences from the family 6 times than the balanced profile and the reverse happened 4 times, while in the remaining 27 cases, they were tied in the recovery rate in this range. The final row tabulates overall wins considered over all the ranges for the 37 tests. Note that every row sums to the total number of tests - 37.

The small sample size does not allow conjecture toward significant conclusions, however it appears that the balanced profile will perform no worse on average than the equal-weights profile. Simply by inspection, it appears that the balanced profile performed better when the blocks which composed the multiple alignment were more divergent. This is to be expected since this is exactly the type of data that exploits the benefits of using balanced profiles. As seen in Table 1, the overwhelming number of results are ties. This is likely a result of the nature of the experimental dataset. The majority of blocks used in these experiments were composed of sequences with a high degree of similarity. When data sets are constructed from very similar sequences, the weights generated by the balanced profile method should not significantly impact the database search. Those blocks which resulted in ties often had more residues which were completely conserved over all sequences than those for which the balanced profile won. These preliminary experiments have helped to provide some insight into potential situations where balanced profiles would be a helpful search tool. Further experiments with datasets with a wide variety of skew will enable the specific usefulness of balanced profiles to be further defined.

4 Approximation Algorithms for the Consensus Problem

The *sphere* in Σ^n of *radius* r with center $a \in \Sigma^n$ is the set of all sequences $v \in \Sigma^n$ such that $d(v, a) \leq r$, and is denoted by $S(a, r)$. Given a sequence x and a set $V \subseteq \Sigma^n$, the *radius* of V with respect to x, denoted by $R(V, x)$ is the the smallest integer r such that $V \subseteq S(x, r)$. We define the *radius* of V, denoted by $R(V)$, as $R(V) = \min_{x \in \Sigma^n} R(V, x)$.

The consensus problem is then the following: given a set of sequences $V \subseteq \Sigma^n$, representing the rows of some multiple alignment, find a sequence $c \in \Sigma^n$ (the consensus) such that $R(V, c) = R(V)$ (in other words, find the sequence c which minimizes $\max_{a \in V} d(c, a)$).

It has been shown by Frances and Litman ([FL94]) that determining the consensus is NP-complete (using a reduction from 3-SAT), in the special case where the alphabet is binary and the distance measure is the Hamming distance. A slight modification of their reduction generalizes the hardness result to arbitrary finite alphabet Σ. Assuming the distance function d takes only rational values, and using the fact that scaling d does not change the problem, allows us to generalize the hardness results to arbitrary distance functions d. Because of the computational complexity of the problem, exact methods for finding the consensus sequence may require too much computing time and heuristic procedures should be sought instead. In particular, we are interested in fast performance-guarantee algorithms.

Our version of the consensus problem arises as a natural alternate objective for tree alignment in the special case when the tree is a star with the given sequences at the leaves and the internal node has to be computed so as to minimize the bottleneck cost of the tree – namely, minimize the maximum distance from the internal sequence to the other input sequences. This is termed bottleneck tree alignments by Ravi and Kececioglu [RK95]. Thus our method for finding unbiased consensus sequences also finds near-optimal bottleneck tree alignments for the star. However, our method is applicable only for the case when the distance function does not allow gaps, and is therefore not applicable to the general version of the tree problem allowing arbitrary edit distances.

A trivial 2-approximation algorithm for the consensus problem consists simply in picking any of the given sequences as the consensus [RK95]. In fact, assuming that the distance between sequences satisfies the triangle inequality, we have the following.

Claim 1. *Let V be a set of sequences; then any sequence $v \in V$ gives a 2-approximation of the radius of V.*

Proof. Let $v^\star$ be an optimal solution, and let r denote the minimal radius of a sphere around $v^\star$ that contains V, that is $d(v^\star, x) \leq r$ for every $x \in V$. Let v be an arbitrary sequence in V. We need to show that for every word $w \in V$, the distance between w and v is at most $2r$. Using the inequalities $d(v^\star, v) \leq r$ and

$d(v^\star, w) \leq r$ together with the triangle inequality $d(v, w) \leq d(v, v^\star) + d(v^\star, w)$, completes the proof.

□

4.1 Near-optimal approximation using randomized rounding

In this section we use the method of randomized rounding [RT87, Rag88, MR95] to achieve a near-optimal solution to the consensus problem. The method can be roughly described as follows: First we formulate our problem as an integer-programming problem, using zero-one variables. Then, we relax the integrality constrains, so that the variables are allowed to have fractional values. An optimal solution to this linear programming problem can be found in polynomial time [Kar84]. To obtain a solution to the original, integer program, we "round" the solution of the relaxed problem, using the fractional values in each column as probabilities. We then show that with high probability, the value of the rounded solution is close to the value of the non-integral (optimal) solution.

The consensus problem can be cast as a zero-one linear program as follows. Let c be the consensus sequence to be determined. For every symbol $\sigma \in \Sigma$, and every column i ($1 \leq i \leq n$), we use a zero-one variable $x_{i,\sigma}$ to indicate whether $c_i = \sigma$. Note that we do not allow the blank character to occur as part of the consensus sequence. Our integer program can be expressed as follows.

$$\begin{array}{rll} Minimize & r & \\ s.t. & \sum_\sigma x_{i,\sigma} = 1 \qquad \forall i \in \{1, \dots n\} & (2) \\ & \sum_{i,\sigma} x_{i,\sigma} d(\sigma, v_i) \leq r \; \forall v \in V & (3) \\ where & x_{i,\sigma} \in \{0, 1\} & (4) \end{array}$$

The constraint (2) ensures that a unique symbol is chosen for each position of c. The constraint (3) specifies that the total distance between any member of V and c is at most r (i.e, $R(V, c) \leq r$). The objective function seeks a solution of minimum radius, r, with the zero-one constraint imposed. Let r_0 denote the value of the objective function in the optimum solution to the program above. Since this problem is NP-Hard, we do not hope to compute r_0 efficiently. Instead, we solve its *linear programming relaxation.*

We replace the integrality constraint (4) with $x_{i,\sigma} \geq 0$. In other words, we allow $x_{i,\sigma}$ to assume real values between 0 and 1 (the constraint $x_{i,\sigma} \leq 1$ is implicit in the constraint (2)). Let $\hat{r}$ be the value of the objective function for this problem and $\hat{x}_{i,\sigma}$ its solution. Since the linear program is a relaxation of the integer program, it is clear that $r_0 \geq \hat{r}$. The $\hat{x}_{i,\sigma}$'s may be fractional values, and therefore may not constitute a feasible solution to the integer program. We must therefore "round" these fractional values to 0's and 1's to obtain a feasible solution x.

Note that the fractional solution $\hat{x}_{i,\sigma}$ still satisfies the constraints of the original linear program, in particular, for each i, the values $\{\hat{x}_{i,\sigma}\}_{\sigma \in \Sigma}$ are all non-negative, and satisfy $\sum_{\sigma\in\Sigma}\hat{x}_{i,\sigma}=1$. Therefore, they define a probability distribution over Σ. The rounding process goes as follows. Independently for each i, choose a symbol for c_i according to the probability distribution defined by $\{\hat{x}_{i,\sigma}\}_{\sigma\in\Sigma}$. That is, $Pr(c_i=\sigma)=\hat{x}_{i,\sigma}$.

Consider some fixed sequence $v\in V$; The expected value of the distance between c and v satisfies the following.

$$\begin{aligned}
E[d(c,v)] &= E[\sum_{i=1}^{n} d(c_i,v_i)] \\
&= \sum_{i=1}^{n} E[d(c_i,v_i)] \quad \text{(by linearity of expectation)} \\
&= \sum_{i=1}^{n}\sum_{\sigma\in\Sigma} Pr(c_i=\sigma)d(\sigma,v_i) \\
&= \sum_{i=1}^{n}\sum_{\sigma\in\Sigma} \hat{x}_{i,\sigma}d(\sigma,v_i) \\
&\le \hat{r} \\
&\le r_0
\end{aligned}$$

Following [RT87], we use Hoeffding's bound [Hoe63] (See, e.g., [MR95]).

Lemma 2. *Let $X_1, X_2, \ldots, X_n$ be n independent random variables, each ranging over the real interval $[a,b]$. Let S be a random variable denoting the sum of the X_i's, and δ any nonnegative real number. Then*

$$Pr(S \ge (1+\delta)E[S]) \le e^{-\frac{E[S]\delta^2}{3(b-a)^2}}$$

Notice that each random variable $d(c_i,v_i)$ range over $[0,D]$, where D is an upper bound on the distance function, e.g. $D=\max_{\alpha,\beta\in\Sigma}\{d(\alpha,\beta)\}$. Let $\epsilon>0$ be a positive constant. For

$$\delta = D\sqrt{\frac{3}{E[S]}\log\frac{|V|}{\epsilon}},$$

we get $Pr(d(c,v)\ge r_0(1+\delta))\le\frac{\epsilon}{|V|}$. Summing over every member of V, we have $Pr(R(V,c)\ge r_0(1+\delta))\le\epsilon$. We get the following theorem.

Theorem 3. *Let $V\subseteq\Sigma^n$ be a set of sequences, and let c_{opt} denote the optimal consensus. Let r_0 denote $R(V,c_{opt})$. Let c be the rounded solution to the relaxed integer program above, and let r denote $R(V,c)$. Then*

$$Pr\left(r > r_0 + D\sqrt{3r_0\log\frac{|V|}{\epsilon}}\right) < \epsilon$$

where $\epsilon>0$ is a constant.

Notice that this probabilistic algorithm can be de-randomized using standard techniques of conditional probabilities, along with pessimistic estimators as in [Rag88].

Remark. Using a different version of the Hoeffding bound: $Pr(S - E[S] > \delta) \leq e^{-\frac{2\delta^2}{(b-a)^2 n}}$, we can derive the following result.

$$Pr\left(r > r_0 + D\sqrt{\frac{n}{2}\log\frac{|V|}{\epsilon}} \right) < \epsilon$$

where $\epsilon > 0$ is a constant. This version is better when $r_0 >> D\sqrt{\frac{n}{2}\log\frac{|V|}{\epsilon}}$.

5 Extensions

5.1 Selecting Probes for Bacterial Infections

Assume a patient has a bacterial infection. Let S be the set consisting of short nucleotides sequences which are specific DNA (either chromosomal or ribosomal) of several possible bacteria that might be the cause of the infection. Let $T \subseteq S$ be a set of sequences of the target bacteria species that actually cause the infection.

DNA diagnosis for the bacterial infection ([DKK88, MM90]) is a technique using the complementary nature of DNA nucleotides to decide whether the cause of the infection is from the set T. The idea is to choose a DNA sequence, called the *DNA probe*, such that its complement is "close" to the sequences in T, and "far away" from the sequences in $S \setminus T$. In practice, even if the complementary sequence of a probe has a few mismatches with a specific substring of the target sequences, the probe forms duplexes with some of the target sequences. Thus for any given probe, the accuracy of the diagnosis using this probe is a function of two parameters:

- The maximum number of mismatches between the complementary sequence of the probe and any nucleotide sequence from the target bacteria. The smaller this number is, the smaller is the probability of a false negative.
- The minimum number of mismatches between the complementary sequence of the probe and any nucleotide sequences of a bacteria not in the target species. The larger this number is, the smaller is the probability of a false positive.

We say that a sequence t is a *k–separator* with respect to $\langle T, S \setminus T \rangle$ if

$$\min_{v \in S \setminus T} d(t,v) - \max_{v \in T} d(t,v) = k.$$

The Probe–Selection Problem can now be stated as follows: given two sets of sequences, T and $S \setminus T$, find a sequence t with maximum separation.

Note that this problem generalizes the consensus problem (which occurs when $T = S$) and is therefore NP-complete. By casting the problem as a 0-1 program and using randomized rounding as in the preceding section, we obtain the following result.

Theorem 4. *Let $T \subseteq S$ be two sets of sequences, and let t_{opt} denote the optimal $\langle T, S \setminus T\rangle$-separator. Let k_{opt} denote $\min_{v \in S\setminus T} d(t_{opt}, v) - \max_{v \in T} d(t_{opt}, v)$. Let t be the rounded solution to the relaxed integer program, and let k denote $\min_{v \in S\setminus T} d(t, v) - \max_{v \in T} d(t, v)$. Then*

$$Pr\left(k < k_{opt} - D\sqrt{4k_{opt}\log\frac{m}{\epsilon}}\right) < \epsilon$$

where $m = \max\{|T|, |S \setminus T|\}$, and $\epsilon > 0$ is a constant.

5.2 Building Consensus Maps

Assume a human chromosome (modeled by the real interval [0,1]) is known to contain n specific markers. Various mapping techniques are used to order those markers, and even suggest chromosomal locations of them.

Assume we are given a set $\mathcal{M}$ of physical maps of the chromosome. Each map consists of the locations of n markers. That is, each map can be represented by a vector $v \in [0,1]^n$. The distance between two maps, $v, u \in [0,1]^n$ is defined as:

$$d(v,u) = \sum_{i=1,\ldots,n} |v_i - u_i|$$

The goal is to find a map which is a good representative of all maps.

A trivial solution is to choose each marker location in the consensus map as the average location of that marker in the input maps. This solution, however, introduce bias. Alternatively, we can use our bottleneck criterion and apply randomized rounding method to choose a consensus map with almost no bias.

Notice that we can not simply introduce a real variable $y_i \in [0,1]$, for each marker location as the resulting program will not be linear. In order to write an linear program we need to approximate the infinite alphabet $\Sigma = [0,1]$ with a polynomial size alphabet $\Sigma' = \{\frac{1}{m+1}, \frac{2}{m+1}, \ldots, \frac{m}{m+1}\}$. We restrict each marker location in the consensus map to choose a position from Σ'. Naturally, we lose precision, but we can choose m large enough so the inaccuracy introduced will be not more than the inaccuracy introduced by the rounding phase.

We introduce mn zero-one variables $x_{i,j}$, where $x_{i,j} = 1$ if and only if the i-th marker location in the consensus map c is $\frac{i}{m+1}$. The resulting integer linear program is therefore:

$$\begin{array}{rll} Minimize & r & \\ s.t. & \sum_j x_{i,j} = 1 & \forall i \in \{1, \ldots n\} \\ & \sum_{i,j} x_{i,j} d(\frac{j}{m+1}, v_i) \leq r \ \forall v \in \mathcal{M} & \\ where & x_{i,j} \in \{0,1\} & \end{array}$$

To get the consensus map, we relax the integrality constraints, and round the linear solution, as in the previous section.

Notice that we can bound the distance between the Σ-optimal map to the Σ'-optimal map by $\frac{n}{m}$ for $m = n\sqrt{n\,|\mathcal{M}|}$, and using the fact that the distance function d is bounded by 1 (i.e. $D = 1$), we have the following.

Theorem 5. *Let $\mathcal{M}$ be a set of maps, and let c_{opt} denote the optimal consensus map. Let r_0 denote $R(\mathcal{M}, c_{opt})$. Let c be the rounded solution to the relaxed integer program above, and let r denote $R(\mathcal{M}, c)$. Then*

$$Pr\left(r > r_0 + 3\sqrt{\frac{n}{2}\log\frac{2\,|\mathcal{M}|}{\epsilon}}\right) < \epsilon$$

where $\epsilon > 0$ is a constant.

6 Open Questions

In order to check the performance of the balanced profile method more accurately, the most divergent protein families should be selected for searching. This way, we would be testing the cases that the weighting systems are developed for, namely sequence sets which may result in bias. A fair comparison with weighting schemes other than equal weights must be done to determine its relative efficacy.

While our method for consensus sequences applies to compute bottleneck tree alignments for a star, they do not extend directly to arbitrary tree topologies. As a first step, an extension of our method to the general version of the star bottleneck problem allowing edit distances should be investigated. Then, it would be interesting to see if our technique can be further extended to the following problem: given a leaf-labeled tree, find ancestral sequence labels at the internal nodes so that the maximum cost of any edge (edit-distance or even Hamming distance between the endpoints) in the tree is minimized. A logarithmic approximation for this problem even with edit-distances is already known [RK95].

References

[ACL89] Stephen F. Altschul, Raymond J. Carroll, and David J. Lipman. Weights for Data Related by a Tree. *Journal of Molecular Biology*, 207, 647–653, 1989.

[Bai92] A. Bairoch. PROSITE: A Dictionary of Sites and patterns in Proteins. *Nucleic Acids Research*, 20, 2019–2022, 1992.

[BB92] A. Bairoch, and B. Boeckmann. The SWISSPROT Protein Sequence Data Bank. *Nucleic Acids Research*, 20, 2019–2022, 1992.

[DBH83] M.O. Dayhoff, W.C. Barker and L.T. Hunt. Establishing homologies in protein sequences. *Methods Enzymol.*, 91:524–545, 1983.

[DKK88] R. Dular, R. Kajioka, and S. Kasatiya. Comparison of gene-probe commercial kit and culture technique for the diagnosis of mycoplasma pneumoniae infection. *J. of Clinical Microbiology*, 26(5):1068–1069, May 1988.

[EMD95] S.R. Eddy, G. Mitchison, and R. Durbin. Maximum discrimination hidden Markov models of sequence consensus. *J. of Computational Biology*, 2:9-23. 1995.

[FL94] M. Frances and A. Litman. On covering problems of codes. Technical Report 827, Technion, Israel, July 1994.

[GCG94] *Program Manual for the Wisconsin Package*, Version 8, September 1994, Genetics Computer Group, 575 Science Drive, Madison, Wisconsin, USA 53711.

[GME87] M. Gribskov, A. D. McLachlan, and D. Eisenberg. Profile Analysis: Detection of Distantly Related Proteins. *Proceedings of the National Academy of Science, U.S.A.*, 84, 4355–4358, 1987.

[GSC94] M. Gerstein, E. Sonnhammer, and C. Chothia. Volume Changes in protein evolution. *J. Mol. Biol.*, 235:1067-1078, 1994.

[HH94] Steven Henikoff and Jorja G. Henikoff. Position-based Sequence Weights. *J. Mol. Biol.*, 243, 574–578, 1994.

[Hoe63] W. Hoeffding. Probability inequalities for sums of bound random variables. *J. Amer. Statist. Assoc.*, 58:13–30, 1963.

[ISNH94] M. Ito, K. Shimizu, M. Nakanishi, and A. Hashimoto. Polynomial-time algorithms for computing characteristic strings. *Proc. CPM 94*, LNCS 807:274–288, 1994.

[Kar84] N. Karmarkar. A new polynomial time algorithm for linear programming, *Combinatorica*, 4:373-395, 1984.

[KM95] A. Krogh, and G. Mitchison. Maximum entropy weighting of aligned sequences of protein or DNA, in *Proc. Third Int. Conf. on Intelligent System for Mol. Biol.*, (C. Rawlings, D. Clark, R. Altman, L. Hunter, T. Lengauer, S. Wodak, eds.) pp. 215-221, AAAI Press, Menlo Park, CA, 1995.

[LXB94] R. Luthy, I. Xenarios, and P. Bicher. Improving the sensitivity of the sequence profile method, *Protein Science*, 3:139–146, 1994.

[MM90] A.J.L. Macario and E.C.De. Macario. *Gene Probes for Bacteria.* Academic Press, 1990.

[MR95] R. Motwani and P. Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.

[Rag88] P. Raghavan. A probabilistic construction of deterministic algorithms: Approximating packing integer programs. *Journal of Computer and System Sciences*, 37:130–143, 1988.

[RK95] R. Ravi and J. D. Kececioglu. Approximation algorithms for multiple sequence alignment under a fixed evolutionary tree, *Proc. CPM 95*, LNCS 937:330–339, 1995.

[RT87] P. Raghavan and C.D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs, *Combinatorica*, 7:365–374, 1987.

[SA90] Peter R. Sibbald and Patrick Argos. Weighting Aligned Protein or Nucleic Acid Sequences to Correct for Unequal Representation. *Journal of Molecular Biology*, 216, 813–818, 1990.

[SW81] T.F. Smith and M.S. Waterman. Comparison of Biosequences. *Adv. Appl. Math.*, 482–489, 1981.

[THG94] J.D. Thompson, D.G. Higgins and T.J. Gibson. Improved sensitivity of profile searches through the use of sequence weights and gap excision, *Comput. Applic. Biosci.*, 10:19–29, 1994.

[VA89] M. Vingron and P. Argos. A fast and sensitive multiple sequence alignment algorithm. *Comput. Appl. Biosci.*, 5:115–121, 1989.

[VS93] M. Vingron and P.R. Sibbald. Weighting in sequence space: A comparison of methods in terms of generalized sequences. *Proc. Natl. Acad. Sci. USA*, 90:8777-8781, 1993.

This article was processed using the LaTeX macro package with LLNCS style