18.417 Introduction to Computational Molecular Biology	
Lecture 20: November 18, 2004	Scribe: Eugenia Lyashenko
Lecturer: Ross Lippert	Editor: Eugenia Lyashenko

Probabilistic Methods: Random Projections

Introduction.

Assume we have n sequences of bases (symbols) of length t.

Definition 1 *An (ℓ, d)-**motif** is a substring of length ℓ occurring with no more then d mismatches in each of the n strings.*

The planted (ℓ, d)-motif problem is as follows:

- Input: n sequences of length t with at least one (ℓ, d)-motif.
- Output: (ℓ, d)-motif.

The planted (ℓ, d)-motif problem was introduced by Pavel Pevzner in some computational biology talk in 2001. **Random Projections** is a randomized approach to a motif finding problem. It is guaranteed to find a right answer with high probability, but it is very particular to the motif finding problem.

If we want to find an ℓ-long pattern implanted in DNA sequences without any mutations, then the motif finding problem is reduced to a simple count of the number of exact occurrences of the ℓ-mers and finding the most frequent ones. If, however, the pattern is implanted with mutations, the problem is much more complicated.

Assume that we are given that a random ℓ-mer with up to d changes is present in a sequence, but we do not know if the motif is present. What if there are many of such substrings, and they do not constitute a motif?

The intuition behind why the random approach should return a right answer with high probability is following: considering the size of the input, the motifs are selected only if they are actually implanted in there.

Indeed, consider an example of a planted $(15, 4)$-motif problem on 20 words of 600 letters. We give a probability argument that a $(15, 4)$-motif will not occur, unless inserted.

We start with a general setup. Consider n sequences of length t. At each position one of 4 letters can be present. Each character can be viewed as an independent Bernoulli trial with probability of success $1/4$ (the character matches the one at the certain position in the ℓ-mer), and correspondingly probability of failure $3/4$.

We hit the motif if in the given ℓ-mer at most d letters mismatch. Then we can roughly calculate probability of a hit P_{hit}:

$$P_{hit} = \sum_{i=0}^{d} \binom{\ell}{i} \left(\frac{1}{4}\right)^{(\ell-i)} \left(\frac{3}{4}\right)^{i}.$$

This is a fake probability since it does not take into account overlaps of ℓ-mers. However, it gives an idea of what is the probability that given pattern S occurs with at most d mismatches at a given position of a random sequence.

Since in each sequence of length t there are at most $(t - \ell + 1)$ possible ℓ-mers, the probability that at least on of them will have up to d mismatches with the pattern P is

$$P_1 = 1 - (1 - P_{hit})^{(t-l+1)}.$$

Then the probability that mutated pattern P is present in all n sequences is

$$P_n = P_1^n.$$

Thus the expected number of (ℓ, d) motifs is

$$E(\ell, d) \approx 4^{\ell}(1 - (1 - P_{hit})^{t-l+1})^n.$$

Plugging in 9 for ℓ and 2 for d, we see that $E(9, 2) \approx 1.6$. Hence $(9, 2)$-motif occurs fairly often. Similarly, $E(15, 5) \approx 2.8$ and $E(13, 4) \approx 5.2$.

We call (ℓ, d)-motifs *twilight zone motifs* if they lie at the boundary of motifs which we do not expect to find unless inserted. Our probability analysis shows that a $(15, 4)$-motif is in twilight zone.

Approaches to motif finding problem

If we wanted to solve motif finding problem by brute force, we would need to sort through all 4^{ℓ} possible motifs. Clearly, this approach is impractical and too compu-

tationally intensive. The several approaches developed in the past are:

Inductive. Counting over-represented k-mer for $k < \ell$. This approach reduces a problem to motifs of smaller size.

Gibbs sampling. Gibbs sampling is described in [1][p. 412].

WINNOWER. Constructs a graph whose vertices correspond to the ℓ-mers present in the input sequences, with an edge connecting two vertices if and only if the corresponding ℓ-mers differ in at most $2d$ positions and do not both come from the same input sequence. WINNOWER then looks for a clique of size t in this graph.

Random projections algorithm

Later a better idea was introduced: random projections. Suppose we have a planted (ℓ, d)-motif problem on t sequences of n letters. We randomly select k positions out of ℓ positions. This selection is called (ℓ, k) **gapped pattern**.

For each ℓ-mer in the data we look at its k-subset which is defined by the gapped pattern. This subset is called **projection**. We use the obtained k-subset as a hash value for the ℓ-mer.

Figure 20.1: Hashing ℓ-mers.

Given a gapped pattern, count all k-length hashes of all ℓ-mers. We cannot avoid collisions, since many ℓ-mers can produce same hash. We group ℓ-mers that hash to the same k-mer in a set called **bucket**. Random sequences should have the same bucket size for any hashed pattern. Hopefully, some buckets will be significantly over-represented, which will mean that they represent motifs.

Lets see what parameter k should we consider. First of all, for a hash to have a practical meaning, we need $k \leq \ell - d$. Otherwise, we risk including mutated bases in our hash function.

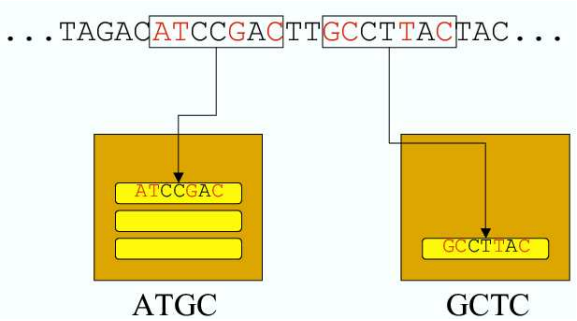

Figure 20.2: Buckets.

On the other hand, we want to exclude random sequences. For this we bound k from below. Given a random sequence, calculate expected number of counts in its bucket:

$$E[counts] = \frac{n \cdot t}{4^k}.$$

Hence if $k > \log_4 n \cdot t$, then $E[counts] < 1$ for a random sequence.

Let P be a pattern which represents an (ℓ, d)-motif. We define a **planted bucket** to be a bucket to which P hashes.

The probability that instance of P in a single sequence hashes to $h(P)$ is

$$Prob[P \to h(P)] = \frac{\binom{l-d}{k}}{\binom{l}{k}} = P_{hit}.$$

For $k = 7$ and for $(\ell, d) = (15, 4)$, $P_{hit} \approx 0.06$ and we have 20 chances to hit it.

The expected number of buckets hit by random sequences is

$$E[|h(P)|] = \frac{n \cdot t}{4^k} + n \cdot P_{hit},$$

where $n \cdot P_{hit}$ is a probability that expected sequence has got matched with a bucket. This formula tells us that we will see an over-representation in some bucket.

For $(\ell, d) = (15, 4)$ and $k = 7$, $E[|h(P)|] = 0.73 + 1.02 = 1.75$.

We define a threshold S on bucket counts. Buckets with size above S will be suspects for a planted bucket.

The probability that the size of planted bucket will fail to exceed the threshold is:

$$Prob[|h(P)| < S] = \sum_{i=0}^{s-1} \binom{n}{i} (P_{hit})^i (1 - P_{hit})^{(n-i)} = P_{fail}.$$

Above value can be as large as 0.1, which clearly does not help to find the planted bucket.

The solution is to repeat the projection m times. The probability of failing to flag the planted bucket $h(P)$ after m trials is P_{fail}^m.

If we agree to not be able to find motif 1% of time, take

$$m \geq \frac{\log(0.01)}{\log(P_{fail})}.$$

Then 99% of time, the planted bucket gets over the threshold S at least once in m trials.

Having identified the planted bucket we need to figure out what is the motif. The algorithm for doing this is roughly as follows:

1. Recover the locations which hash to the planted bucket.
2. Construct a profile from these substrings.
3. Use an iteration motif finder from this profile.
4. If the consensus is an (ℓ, d)-motif, then report.

Conclusion

Random projections algorithm works better than other mentioned algorithms because previous work was largely based on k-mer counts. In other algorithms, every frequent k-mer is used either in some weird graph problem or every ℓ-mer in the sequence is used as an iterative start point as done in WINNOWER, for example. But consecutive patterns just do not get into the twilight zone.

However, situation with real data is a bit different. In real data we have non-uniform letter frequencies, non-independence and mismatch positions much more correlated, since "random data" has non-random profile.

On real data we do not have uniformity and there is no guarantee that motif indeed occurs in every sequence. Nevertheless the random projection algorithm demonstrated to reproduce some known motif results.

References

[1] N.C. Jones, P.A. Pevzner, *Introduction to Bionformatics Algorithms*, A Bradford Book, The MIT Press, Cambridge, 2004.

[2] J. Buhler, M. Tompa, *Finding Motifs Using Random Projections*, Annual Conference on Research in Computational Molecular Biology, Proceedings RECOMB'01, Montreal, 2001.