L4: Blast: Alignment Scores etc.

[Diagram of Blast webpage with various options such as Nucleotide, Protein, Genomes, Translated, Special, Meta, etc.]

Disclaimer
Privacy statement
Accessibility
Valid XHTML 1.0, CSS.
Why is Blast Fast?
Silly Question

- Prove or Disprove:
  - There are two people in New York City with exactly the same number of hairs.
Large database search

Database size n=10M, Query size m=300.

$O(nm) = 3 \times 10^9$ computations
Observations

- Much of the database is random from the query’s perspective.
- Consider a random DNA string of length $n$.
  - $\Pr[A] = \Pr[C] = \Pr[G] = \Pr[T] = 0.25$
- What is the probability that an exact match to the query can be found?
Basic probability

- Probability that there is a match starting at a fixed position \(i = 0.25^m\)
- What is the probability that some position \(i\) has a match.
- Dependencies confound probability estimates.
Basic Probability: Expectation

- Q: Toss a coin: each time it comes up heads, you get a dollar
  - What is the money you expect to get after n tosses?
  - Let $X_i$ be the amount earned in the i-th toss

\[ E(X_i) = 1 \cdot p + 0 \cdot (1 - p) = p \]

- Total money you expect to earn

\[ E\left(\sum_i X_i\right) = \sum_i E(X_i) = np \]
Expected number of matches

- Expected number of matches can still be computed.

\[ \text{Let } X_i = 1 \text{ if there is a match starting at position } i, \text{ } X_i = 0 \text{ otherwise} \]

\[ \Pr(\text{Match at Position } i) = p_i = 0.25^m \]

\[ E(X_i) = p_i = 0.25^m \]

- Expected number of matches =

\[ E(\bigcap_i X_i) = \prod_i E(X_i) = n \left(\frac{1}{4}\right)^m \]
Expected number of exact Matches is small!

- Expected number of matches = $n*0.25^m$
  - If $n=10^7$, $m=10$,
    - Then, expected number of matches = 9.537
  - If $n=10^7$, $m=11$
    - expected number of hits = 2.38
  - $n=10^7$, $m=12$,
    - Expected number of hits = 0.5 < 1

- Bottom Line: An exact match to a substring of the query is unlikely just by chance.
Observation 2

- What is the pigeonhole principle?
  - Suppose we are looking for a database string with greater than 90% identity to the query (length 100).
  - Partition the query into size 10 substrings. At least one much match the database string exactly.
Why is this important?

- Suppose we are looking for sequences that are 80% identical to the query sequence of length 100.
- Assume that the mismatches are randomly distributed.
- What is the probability that there is no stretch of 10 bp, where the query and the subject match exactly?

\[
\left( \frac{8}{10} \right)^{10}^{90} = 0.000036
\]

- Rough calculations show that it is very low. Exact match of a short query substring to a truly similar subject is very high.
  - The above equation does not take dependencies into account
  - Reality is better because the matches are not randomly distributed
Just the Facts

- Consider the set of all substrings of the query string of fixed length $W$.
  - Prob. of exact match to a random database string is very low.
  - Prob. of exact match to a true homolog is very high.
  - Keyword Search (exact matches) is MUCH faster than sequence alignment.
- Consider all $(m-W)$ query words of size $W$ (Default = 11)
- Scan the database for exact match to all such words
- For all regions that hit, extend using a dynamic programming alignment.
- Can be many orders of magnitude faster than SW over the entire string
Why is BLAST fast?

• Assume that keyword searching does not consume any time and that alignment computation the expensive step.
• Query m=1000, random Db n=10^7, no TP
• SW = O(nm) = 1000*10^7 = 10^{10} computations
• BLAST, W=11
  • E(#11-mer hits)= 1000* (1/4)^11 * 10^7=2384
  • Number of computations = 2384*100*10=2.384*10^6
  • Ratio=10^{10}/(2.384*10^6)=4200
• Further speed improvements are possible
Keyword Matching

- How fast can we match keywords?
- Hash table/Db index? What is the size of the hash table, for m=11
- Suffix trees? What is the size of the suffix trees?
- Trie based search. We will do this in class.
Related notes

- **How to choose the alignment region?**
  - Extend greedily until the score falls below a certain threshold

- **What about protein sequences?**
  - Default word size = 3, and mismatches are allowed.

- **Like sequences, BLAST has been evolving continuously**
  - Banded alignment
  - Seed selection
  - Scanning for exact matches, keyword search versus database indexing
P-value computation

- How significant is a score? What happens to significance when you change the score function?
- A simple empirical method:
  - Compute a distribution of scores against a random database.
  - Use an estimate of the area under the curve to get the probability.
  - OR, fit the distribution to one of the standard distributions.
Z-scores for alignment

- Initial assumption was that the scores followed a normal distribution.
- Z-score computation:
  - For any alignment, score $S$, shuffle one of the sequences many times, and recompute alignment. Get mean and standard deviation.

$$Z_S = \frac{S - \mu}{\sigma}$$

- Look up a table to get a P-value.
Blast E-value

- Initial (and natural) assumption was that scores followed a Normal distribution
- 1990, Karlin and Altschul showed that ungapped local alignment scores follow an exponential distribution
- Practical consequence:
  - Longer tail.
  - Previously significant hits now not so significant
Exponential distribution

- Random Database, \( \Pr(1) = p \)
- What is the expected number of hits to a sequence of \( k \) 1's

\[
(n \cdot \binom{k}{n} p^k \cdot e^{k \ln p} = ne
\]

- Instead, consider a random binary Matrix. Expected # of diagonals of \( k \) 1s

\[
\frac{n \cdot \binom{k}{n} (m \cdot \binom{k}{m} p^k \cdot nme^{k \ln p} = nme
\]
• As you increase k, the number decreases exponentially.
• The number of diagonals of k runs can be approximated by a Poisson process

\[ Pr[u] = \frac{u^u e^{-L}}{u!} \]
\[ Pr[u > 0] = 1 - e^{-L} \]

• In ungapped alignments, we replace the coin tosses by column scores, but the behaviour does not change (Karlin & Altschul).
• As the score increases, the number of alignments that achieve the score decreases exponentially
Choose a score such that the expected score between a pair of residues < 0

Expected number of alignments with a particular score

\[ E = Kmne^{\frac{S}{ln K}} = mn2 \]

\[ Pr(S \geq x) = 1 - e^{Kmne^{\frac{S}{ln K}}} \]

For small values, E-value and P-value are the same
Conclusion

- Blast is fast because
  - Keyword search is fast (To be shown).
  - Keywords form good filters.
  - The Poisson approximation can be used to show that the keywords form a good filter, as well as to compute P-values for scores.
**Blast Variants**

0. What Are Blastn/Blastp/Blastx/Tblastn/TBlastx
   § What is mega-blast?
   § What is discontiguous mega-blast?
   § Phi-Blast/Psi-Blast?
   § BLAT?
   § PatternHunter?

1. Longer seeds.
2. Seeds with don’t care values
3. Later
4. Database pre-processing
5. Seeds with don’t care values