Using Multiple Alignments to Improve Seeded Local Alignment Algorithms

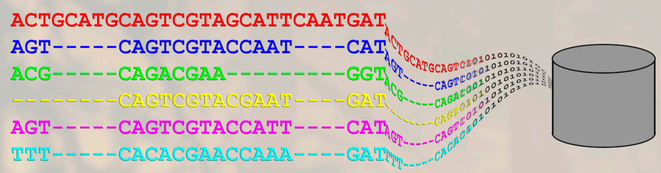
Jason Flannick Serafim Batzoglou

Department of Computer Science, Stanford University, Stanford, CA 94301

Background and Motivation

With the rapid increase in the amount of genomic data available, tools for searching large sequence databases have become very important. Seeded algorithms in the vein of BLAST are the most popular such tools, and the development of spaced seeds in recent years has helped to give these heuristics a mathematical foundation.

From a different angle, algorithms are now capable of multiply aligning entire genomes. This opens a new door: the possibility of multiple alignment databases becoming prevalent. Past work on searching multiple alignments has focused on improved scoring mechanisms.

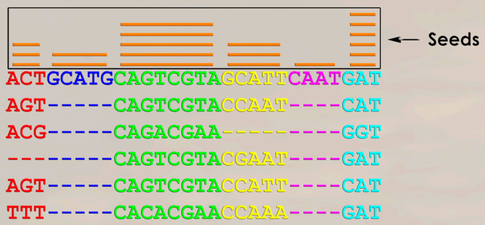


In this work we develop tools for indexing these multiple alignment databases and investigate their performance.

Main Idea

One can build an index from a database using multiple spaced seeds. Adding additional seeds increases search time but achieves higher levels of sensitivity. We assume here we have a set of "candidate seeds".

Sequence database search algorithms must index the same set of seeds at every position. With a multiple alignment database, we can be more flexible. Multiple alignments implicitly specify a set of regions, each of which is conserved to a different extent.



By indexing fewer seeds in poorly conserved regions, we can index more seeds in well-conserved regions, obtaining higher levels of sensitivity at the same total computational cost.

Algorithm

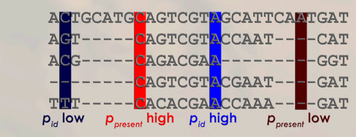
Our indexing algorithm consists of three parts: converting the multiple alignment into a more suitable mathematical format, partitioning the alignment into a set of regions, and determining the set of spaced seeds to index at each position.

Decoding

We obtain for each position in the alignment two values:

$p_{present}$: the probability that the position has a homolog in the query

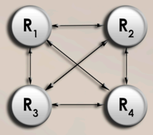
p_{id}: the probability that the homologous position in the query matches the consensus character in the multiple alignment



We can compute these values using dynamic programming, using a phylogenetic tree rooted at the query.

Partitioning

Region boundaries in a multiple alignment reflect changes in conservation level. We group regions into a small set of *region classes*; regions in the same region class have roughly the same conservation level. A region class is defined by two parameters: the average $p_{present}$ for all positions in the region class and the average p_{id} for all positions in the region class.



We use a Hidden Markov Model, where each region class is a state that generates $p_{present}$ and p_{id} values that are close to its average values with high probability.

Seed Assignment

We begin by building a table, with one row for each region class and one column for each potential number of seeds that can be indexed. Each entry in the table specifies a potential number of seeds to index at every position in a region class; the *weight* of an entry is the total number of seeds that would be indexed, and the *value* of an entry is the expected number of homologous regions we would cover with a seed match.

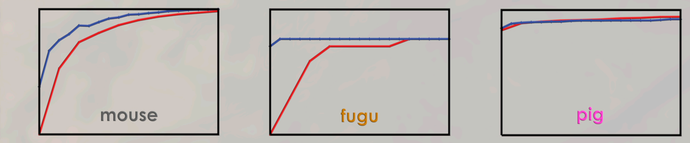
We can reduce this to a variant of the classic Knapsack problem, where the weight of the knapsack is our maximum allowed computational cost.

Experimental Results

We evaluated our method by testing how many homologous regions it aligned between a query and a database. Our multiple alignment consisted of the CFTR regions of baboon, cat, chicken, chimp, cow, dog, human, and rat. Our queries were mouse, fugu, and pig.

How does our method perform?

We compared our method, Typhon, to the STANDARD method that indexes the same seeds at every position. The plots below show sensitivity as a function of the size of the index.



How do multiple alignment databases perform?

We compared Typhon and STANDARD to the STANDARD algorithm for searching the sequence database containing the query's closest relative.

Sensitivity improvements are most dramatic for species that are distant from all species in the multiple alignment.

Conclusions

Our results argue in favor of using multiple alignment databases to increase search sensitivity. We obtain improvements due to two reasons:

1. Multiple alignment databases allow new algorithmic developments.
2. Multiple alignment databases allow more sensitive searches than sequence databases, regardless of the search algorithm used.

References

Flannick J, Batzoglou S. Using multiple alignments to improve seeded local alignment algorithms. Nucleic Acids Res 33(14): 4563–4577, 2005