SynBioBLAST

Ethan Ransom

Abstract

My proposal is to build a version of NCBI's BLAST tool that searches against the sequences contained in SynBioHub. Users would be able to input unannotated sequences (perhaps as part of a migration from other repositories) and the tool would find matches in the SynBioHub collections. Implementation of this tool will require retrieving the contents of SynBioHub and using them to construct a BLAST database. Once the database is constructed, new additions to SynBioHub need to be detected and incrementally added to the database. Using this database, an implementation of BLAST (either the reference implementation or one written over the course of this project) will service user queries.

Introduction

The BLAST algorithm was published in 1990 as a significant improvement upon existing search alignment algorithms available at the time. BLAST gains its speed by sacrificing a dynamic programming approach for a heuristic-based one. This approach is less accurate but is up to 50 times faster. BLAST's speed vastly improved the user experience of querying extremely large databases, and has been extremely successful--the original paper has been cited 50,000 times, making it the most cited paper of the 1990s.

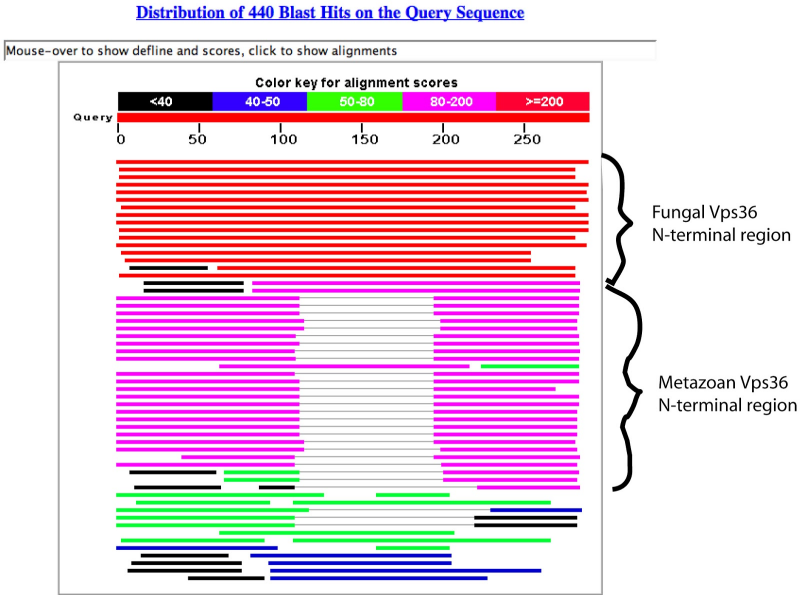

Figure 1: Results of performing a BLAST query.

J. Mol. Biol. 1990 Oct 5;215(3):403–10 —the original description and characterization of the BLAST Algorithm.

Basic Local Alignment Search Tool

Stephen F. Altschul[1], Warren Gish[1], Webb Miller[2]
Eugene W. Myers[3] and David J. Lipman[1]

[1]*National Center for Biotechnology Information*
National Library of Medicine, National Institutes of Health
Bethesda, MD 20894, U.S.A.

[2]*Department of Computer Science*
The Pennsylvania State University, University Park, PA 16802, U.S.A.

[3]*Department of Computer Science*
University of Arizona, Tucson, AZ 85721, U.S.A.

(Received 26 February 1990; accepted 15 May 1990)

A new approach to rapid sequence comparison, basic local alignment search tool (BLAST), directly approximates alignments that optimize a measure of local similarity, the maximal segment pair (MSP) score. Recent mathematical results on the stochastic properties of MSP scores allow an analysis of the performance of this method as well as the statistical significance of alignments it generates. The basic algorithm is simple and robust; it can be implemented in a number of ways and applied in a variety of contexts including straightforward DNA and protein sequence database searches, motif searches, gene identification searches, and in the analysis of multiple regions of similarity in long DNA sequences. In addition to its flexibility and tractability to mathematical analysis, BLAST is an order of magnitude faster than existing sequence comparison tools of comparable sensitivity.

1. Introduction

The discovery of sequence homology to a known protein or family of proteins often provides the first clues about the function of a newly sequenced gene. As the DNA and amino acid sequence databases continue to grow in size they become increasingly useful in the analysis of newly sequenced genes and proteins because of the greater chance of finding such homologies. There are a number of software tools for searching sequence databases but all use some measure of similarity between sequences to distinguish biologically significant relationships from chance similarities. Perhaps the best studied measures are those used in conjunction with variations of the dynamic programming algorithm (Needleman & Wunsch, 1970; Sellers, 1974; Sankoff & Kruskal, 1983; Waterman, 1984). These methods assign scores to insertions, deletions and replacements, and compute an alignment of two sequences that corresponds to the least costly set of such mutations. Such an alignment may be thought of as minimizing the evolutionary distance or maximizing the similarity between the two sequences compared. In either case, the cost of this alignment is a measure of similarity; the algorithm guarantees it is optimal, based on the given scores. Because of their computational requirements, dynamic programming algorithms are impractical for searching large databases without the use of a supercomputer (Gotoh & Tagashira, 1986) or other special purpose hardware (Coulson *et al.*, 1987).

Rapid heuristic algorithms that attempt to approximate the above methods have been developed (Waterman, 1984), allowing large databases to be searched on commonly available computers. In many heuristic methods the measure of similarity is not explicitly defined as a minimal cost set of mutations, but instead is implicit in the algorithm itself. For example, the FASTP program (Lipman & Pearson, 1985; Pearson & Lipman, 1988) first finds locally similar regions between two sequences based on identities but not gaps, and then rescores these regions using a measure of similarity between residues, such as a PAM matrix (Dayhoff *et al.*, 1978) which allows conservative replacements as well as identities to increment the similarity score. Despite their rather indirect approximation of minimal evolution measures, heuristic tools such as FASTP have been quite popular and have identified many distant but biologically significant relationships.

Figure 2: The original BLAST paper--the most cited paper of the 1990s.

Implementation

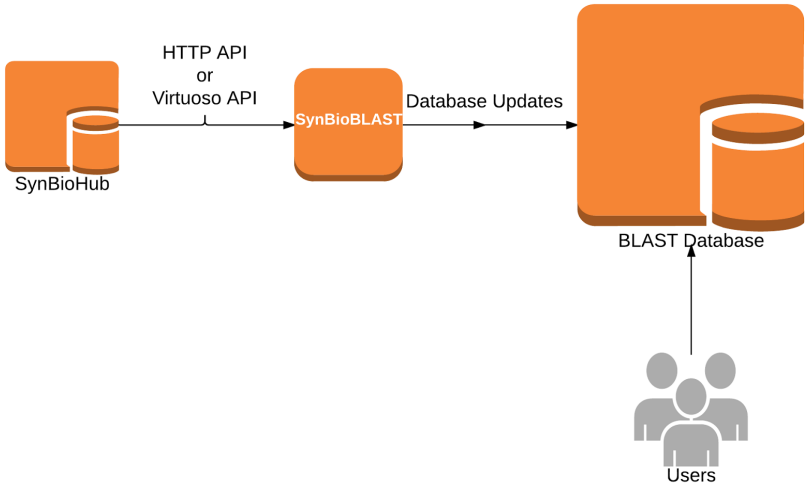

Figure 3: A brief overview of the SynBioBLAST dataflow.

Scraping SynBioHub

My implementation will need to integrate with SynBioHub either at the database level or the HTTP API level. Both have advantages and disadvantages, but choosing an approach is not possible without further investigation into the HTTP API of SynBioHub and the API of virtuoso, the backend database of SynBioHub.

Building a BLAST database

The data retrieved by the scraper will be used to construct a BLAST database, which is a special indexed format used by the BLAST algorithm. Any new additions to SynBioHub will need to be incrementally added to the database. (See the BLAST Handbook for reference, in particular the Dataflow section ->BLAST Databases: Some Details to Keep in Mind)

Running Queries

Using this database, user queries can be serviced. The user interface for launching queries could be part of a standalone service, or integrated into SynBioHub. The NCBI BLAST is sharded over many servers, each running a query against a chunk of the database. Depending on the efficiency of the algorithm and the number of sequences included in the database, SynBioBlast may need to do something similar. In addition, BLAST is very CPU-intensive, necessitating that the actual search be run on a separate machine from the SynBioHub application instances.

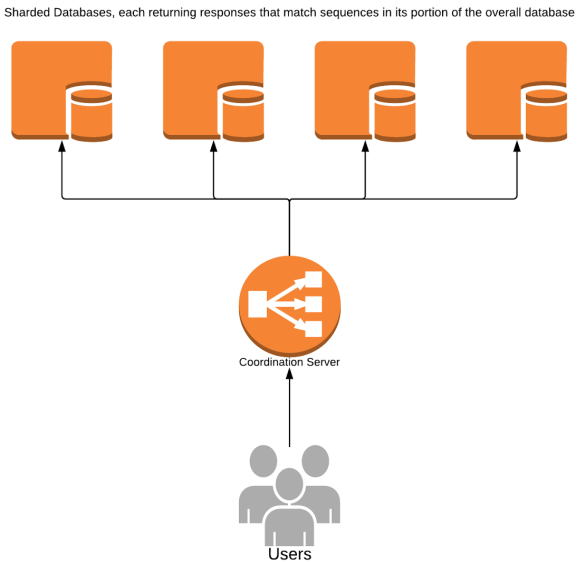

Figure 4: A sharded architecture, which may be necessary if the database is too large to be processed by a single machine.

Reference Implementation vs. Custom Implementation

Research for this project proposal has shown that while a large amount of documentation for the BLAST algorithm exists, the algorithm is of sufficient complexity that writing it again from scratch may be too time consuming. Fortunately, NCBI publishes both the code of the algorithm and code for building the BLAST databases. More investigation will be needed to determine whether the reference implementation will be suitable. My preference is to make use of the reference implementation, as it will be of much higher quality and integrating it will almost certainly consume less time that writing a custom version would.

References

1. Original BLAST paper: https://blastalgorithm.com/
2. Smith-Waterman Algorithm: https://en.wikipedia.org/wiki/Smith%E2%80%93Waterman_algorithm
3. BLAST: https://en.wikipedia.org/wiki/BLAST
4. BLAST Handbook: https://www.ncbi.nlm.nih.gov/books/NBK153387/
5. BLAST source code: ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/