Alarfaj et al., J Comput Sci Syst Biol 2017, 10:5
DOI: 10.4172/jcsb.1000260

Journal of
Computer Science & Systems Biology

Research Article Open Access

Blast-i2b2: Blast for Biological Sequence Comparison in i2b2 Platform

Alaa Alarfaj[1*], Ghada Badr[2,3], Mohy Uddin[4]

[1]*King Abdullah International Medical Research Center, King Saud bin Abdul-Aziz University for Health Sciences, King Abdul-Aziz Medical City, Ministry of National Guard Health Affairs, Riyadh, Kingdom of Saudi Arabia*

[2]*School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Canada*

[3]*IRI - The City of Scientific Research and Technological Applications, Alex, Egypt*

[4]*King Abdullah International Medical Research Center, King Saud bin Abdul-Aziz University for Health Sciences, Publication office, King Abdul-Aziz Medical City, Ministry of National Guard Health Affairs, Riyadh, Kingdom of Saudi Arabia*

Abstract

Finding similarities between biological sequences is an important process to analyze these sequences. The similarity between biological sequences can be determined by using sequence alignment methods. BLAST (Basic Local Alignment Search Tool) is a search tool that is designed to perform the sequence alignment process. The current BLAST platform compares a biological sequence with sequences in a biological sequences database, lists the similar results, and gives the user the ability to download these results to a local machine without storing them in a form, such as data repository or data warehouse which makes reusability and further updates of the results difficult. In this paper, we introduce *BLAST-i2b2*, which implements the BLAST in the i2b2 (Informatics for Integrating Biology and the Bedside) platform, stores the BLAST results, and updates the stored result whenever there are changes in the target biological sequence database. A data warehouse is provided that stores *BLAST-i2b2* search results efficiently, and can be used for further analysis and faster updates. In addition, this paper presents the i2b2 platform, the update BLAST algorithm, and shows the testing of *BLAST-i2b2* performance on three databases of different sizes. The results show that the *BLAST-i2b2* update process reduces the BLAST run time by 84% for the small size database, 93% for the medium size database, and 97% for the large size database. The sequence analysis part is not within the scope of this paper and will be taken into account in the future work.

Keywords: BLAST; i2b2; Sequence alignment; Data warehouse; Sequence comparison

Introduction

In the field of Bioinformatics, the comparison between biological sequences is the most widely used strategy to determine the functionality of newly sequenced genes, extract new members belonging to a specific gene family, and find evolutionary relationships between sequences. Sequence alignment is one of the main techniques for finding similarities between biological sequences. It aligns a given biological sequence with a set of biological sequences that are stored in a given database. BLAST (Basic Local Alignment Search Tool) [1] is a search tool that implements the sequence alignment method and is available at the NCBI (National Center for Biotechnology Information) website[1]. Biological sequence databases contain huge number of DNA, RNA, or protein sequences and are periodically updated. The updates of these databases may contain some newly added sequences, some deleted sequences, or some modified sequences; and usually the updates are minor. The processing time for completing the BLAST job increases with the increase in the target database size. The current BLAST algorithm compares the query sequence with each sequence in the target database and only returns the similar sequences without storing any results in a main center. If that database is updated, the user must run BLAST again to get updated results. In this case, the user has to resubmit the query and the BLAST will run again against the whole updated target database. This shows the need for a data warehouse that can store each submitted query by the user and store its results. Hence, if the target database is updated, the user can request to get an updated result of the previously submitted query and even keep previous versions if needed. In this case, BLAST can run against only the updated sequences, resulting in considerably reducing the run time. In additions, storing the BLAST results enables further analysis and tracking of the similarity changes over time when databases are updated. As the target databases keep updating, the data warehouse should also be capable of dealing with these updates along with the time required for searching the updated database.

The i2b2 (Informatics for Integrating Biology & the Bedside)[2] is an informatics platform that allows the integration of clinical and genomics data in a single repository. It provides researchers with software tools for collecting and managing biomedical research data from a variety of resources. The i2b2 platform's design is scalable and can be extended by developing new components or "*plug-ins*" to suit the needs of the researchers. It also packaged with built-in tools for data query, analysis, and visualization [2]. This paper presents the *BLAST-i2b2* platform - an implementation of BLAST functionality using i2b2, and enhances the BLAST results storage and update features using the capabilities of i2b2 data warehouse. One of the important features of our proposed approach is that we save the BLAST results in a well-designed i2b2 warehouse. In *BLAST-i2b2*, the BLAST compares the query sequence with each sequence in the target database to find the

1 https://blast.ncbi.nlm.nih.gov/Blast.cgi

2 https://www.i2b2.org

***Corresponding author:** Alaa Alarfaj, King Abdullah International Medical Research Center, King Saud bin Abdul-Aziz University for Health Sciences, King Abdul-Aziz Medical City, Ministry of National Guard Health Affairs, Riyadh, Kingdom of Saudi Arabia, Tel: +966114294348; E-mail: alarfajal@ngha.med.sa

Received November 01, 2017; **Accepted** November 05, 2017; **Published** November 10, 2017

Citation: Alarfaj A, Badr G, Uddin M (2017) BLAST-i2b2: BLAST for Biological Sequence Comparison in i2b2 Platform. J Comput Sci Syst Biol 10: 108-120. doi:10.4172/jcsb.1000260

similar sequences, show the BLAST results, and allows the user to save the query parameters and BLAST results in the data warehouse. In addition, it can update the saved results in the data warehouse whenever there are updates in the target biological database, based on user request.

In this scenario, the user can view the BLAST history and select one of the submitted queries to check if there is an update on the target database. If that target database is modified, BLAST compares the stored query sequence with only the modified sequences in the database instead of comparing it with the whole sequences in that database, making BLAST much faster. Since the results are stored in the data warehouse, this approach will allow the researchers to reuse those results for further analysis and research purposes using the built in data exploration and visualization capabilities of i2b2 platform in the future. The next section provides an overview of basic sequence alignment techniques, the BLAST algorithm, NCBI database, and i2b2 platform. Section 3 presents the proposed *BLAST-i2b2* along with save and update algorithms. It also contains the data warehouse design to illustrate the data storage mechanism. Section 4 describes the experimental setup of *BLAST-i2b2* and provides the results in order to test the performance of *BLAST-i2b2* and shows the evaluation of the proposed approach. Finally the conclusion section provides a summary and future work.

Background and Motivations

Sequence alignment

The similarities between biological sequences can be found by using Sequence Alignment method [3]. An alignment is simply done by pairing letters from two sequences: one is defined as a query sequence that is given by the user and the other is the database sequence that resides in the biological sequence database. The result is in the form of a similarity sequence as illustrated in Figure 1. Each letter in the similarity sequence (or alignment) is either a match or a gap. If a letter in the query sequence is identical to the corresponding letter in the database sequence, a match will be inserted in the similarity sequence; and if they are not identical, a gap will be inserted. In order to produce the similarity between two sequences, a similarity score is calculated using a scoring matrix. This matrix is usually called substitution matrix, which assigns predetermined scores to both matched and unmatched sequence pairs. The final similarity score of the two aligned segments is the sum of all scores of each segment pair. PAM [4] and BLOSUM [5] are the most commonly used scoring matrices [6]. Different similarity searching algorithms have been proposed in order to find similarities between sequences, such as Needleman-Wunsch algorithm [7], Smith-Waterman algorithm [8], and BLAST algorithm [1]. In this paper, we focus on the BLAST algorithm.

BLAST

BLAST algorithm was developed by Altschul, Gish, Miller, Myers, and Lipman in 1990 [1]. It was produced to compare a given DNA, RNA, or Protein query sequence against a list of database sequences to identify the most similar database sequences above certain value.

ATGGCGAAATCA	The query sequence
ATGGCG–A– –CA	The similarity sequence
ATGGCGTACC CA	The database sequence

Figure 1: Sequence alignment method.

BLAST use PAM and BLOSUM matrices to calculate the similarity scores [6].

BLAST algorithm: BLAST algorithm [6] consists of four main processes: pre-processing, seeding, extension, and evaluation as illustrated in Figure 2. The main goal of BLAST is to find all HSPs (High Segment Pairs) that have a score equals or more than a given value S [6]. The Inputs to the BLAST algorithm are: the query sequence (the sequence that the user wants to compare), the word size W (the length of the subsequence that initiates the alignment), the threshold T (the number of different alignments expected to be matched by chance. If T=1, it means that 1 alignment match can be found by chance), and cut-off S (a number that reports high-scoring segment pairs).

Pre-process step: Using the query sequence, BLAST generates a word list and a neighbors list. In order to generate the word list, BLAST breaks the query sequence into words of size W. An example is shown in Figure 3a, where W is set to 3. The query sequence and the generated word list are shown. For each word in the word list, a new list of W-length neighbors, which have similarities equal to or greater than T, is generated. Figure 3a shows an example of HEA neighbors' that has a similarity equal to or greater than T=13 using BLOSUM62 matrix.

Seeding step: For each word in the word list, the algorithm scans the matches (seeds) between word's neighbors and target database sequences. Figure 3b shows the seeds of HEA's neighbors.

Extension step: For each match (hit) found in the database, the query sequence is aligned to the database sequence at the seed position. Then, the alignment is extended in both directions until it reaches the highest score. Figure 3c shows the extension process between the alignment of the query sequence and seq1 in the target database.

Evaluation step: Obtained alignments are evaluated to identify if they are High-scoring Segment Pairs (HSPs). To evaluate these alignments, the algorithm compares alignments' scores (significance) with a given cut-off value S. The highest segment score, which is equal to or greater than S, is selected. Figure 3d shows the evaluation process between HEA's seeds with significance score more than or equal to s=40.

BLAST graphical user interface: NCBI provides many variations of BLAST programs[3] that allow the user to search sequence similarities

3 https://blast.ncbi.nlm.nih.gov/Blast.cgi

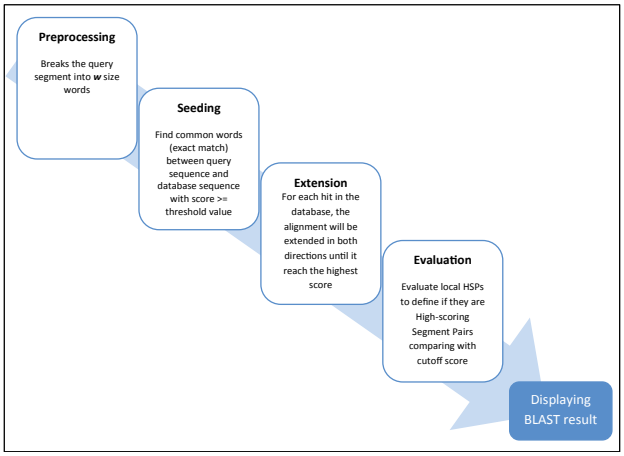


Figure 2: The BLAST Algorithm.

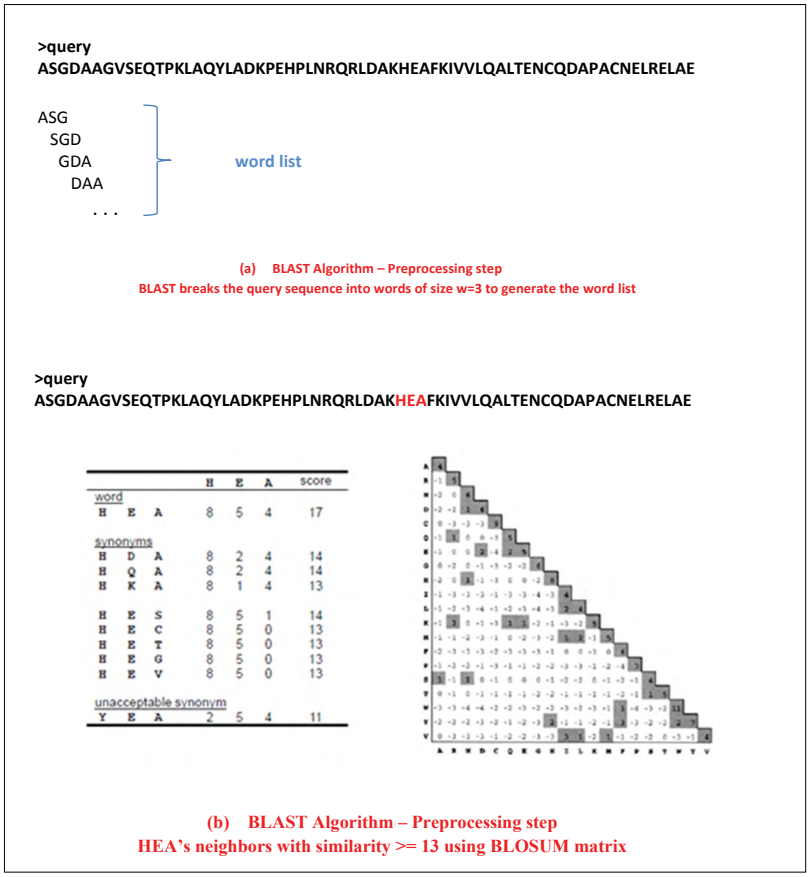


Figure 3a: BLAST Algorithm - Pre-processing step.

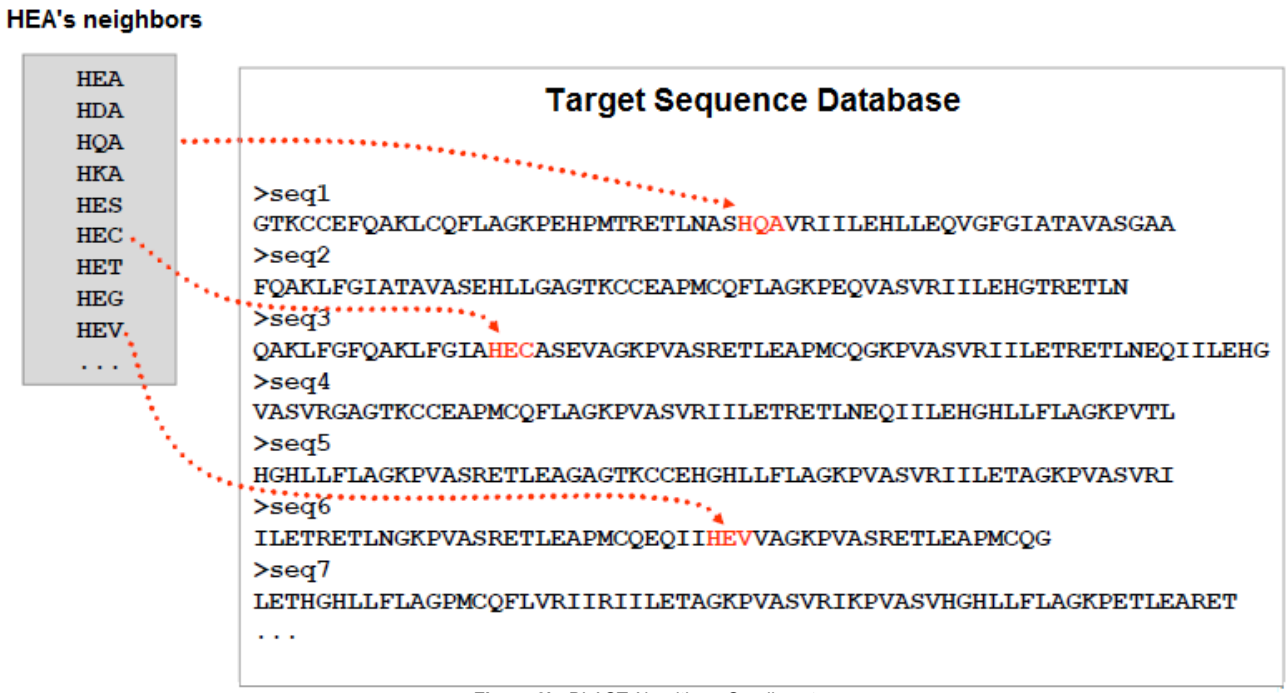


Figure 3b: BLAST Algorithm - Seeding step.

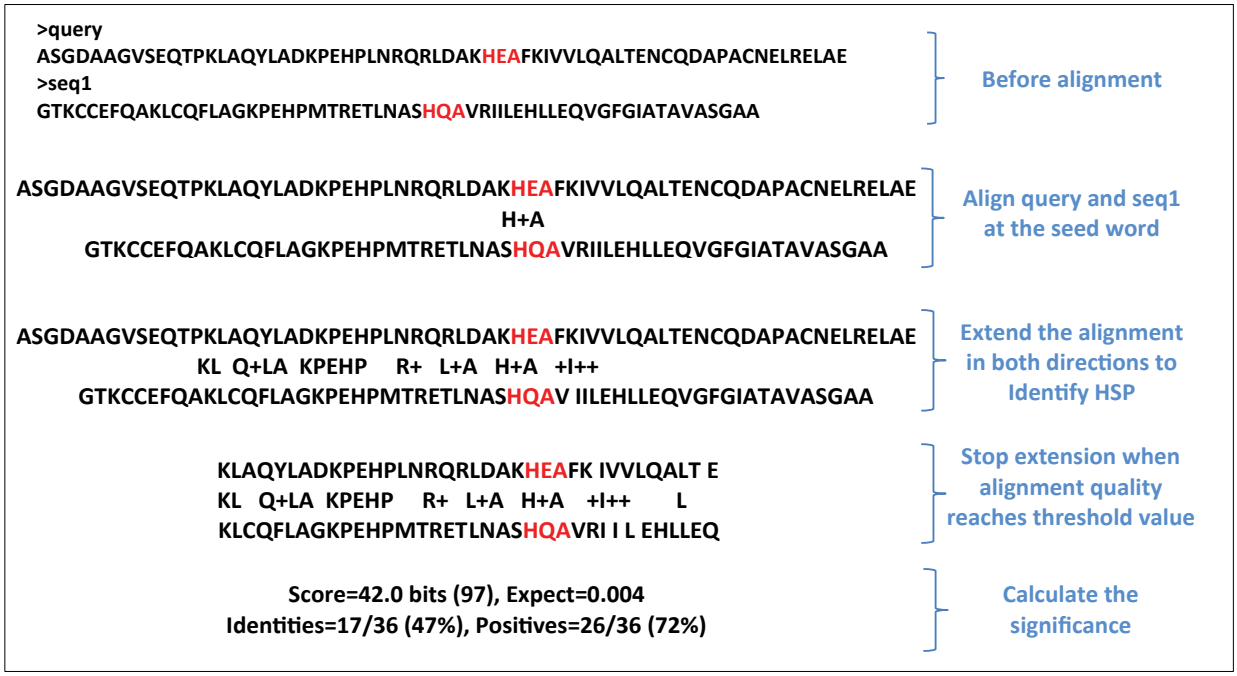


Figure 3c: BLAST Algorithm - Extension step.

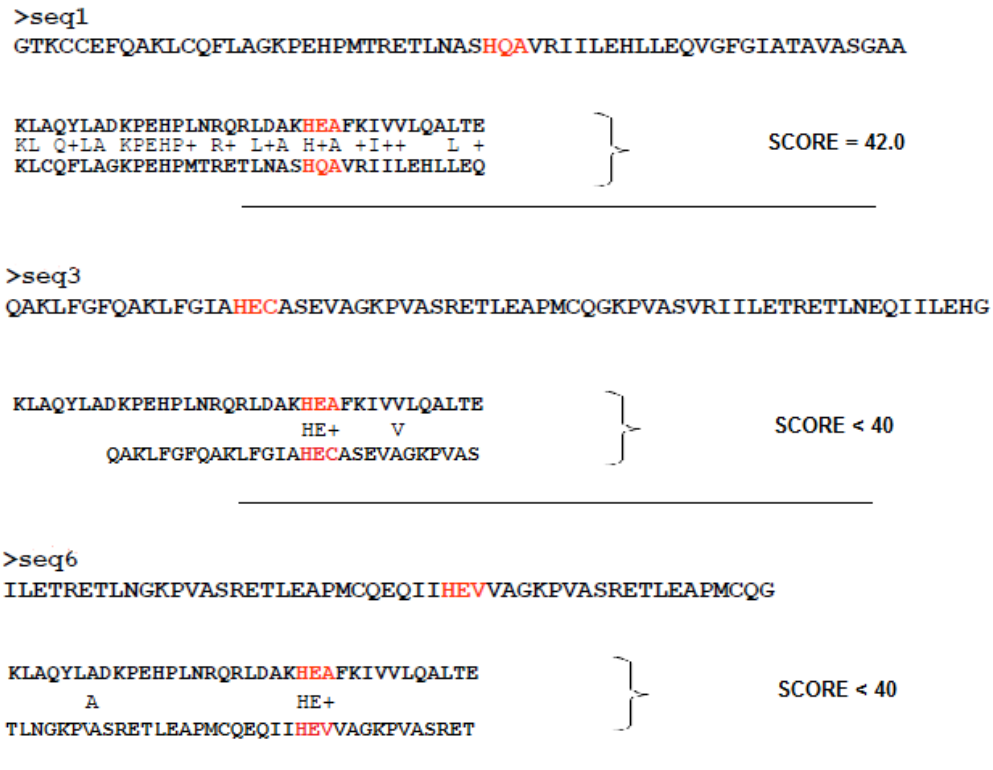


Figure 3d: BLAST Algorithm - Evaluation step.

such as BLASTN, BLASTP, BLASTX, TBLASTN, and TBLASTX. BLASTN compares a given DNA sequence to a DNA database. BLASTP compares a given protein sequence to a protein database. BLASTX compares a given translated DNA sequence to a protein database. TBLASTN compares a given protein sequence to a translated DNA database. TBLASTX compares a given translated DNA sequence to a translated DNA database. BLASTN program interface is shown in Figure 4. The other BLAST programs' interfaces are similar to BLASTN. BLASTN interface has three main sections:

Enter query sequence: This section allows the user to enter a query sequence or upload it as FASTA file. A query sequence maybe entered in three different ways: enter the accession number [9] of the query sequence, enter the gi [9] of the query sequence, or enter the FASTA format [10] of the query sequence. The other choice is upload the sequence file. The file may contain a single sequence or a list of sequences. The data in the file may be either a list of accession numbers, gi numbers, or sequences in FASTA format.

Choose search set: This section allows the user to select the target sequence database from a drop-down menu. The following databases used by BLASTN program are: Human genomic plus transcript (Human G+T), Mouse genomic plus transcript (Mouse G+T), Nucleotide collection (nr/nt), Reference RNA sequence (refseq_rna), RefSeq Genome Database (refseq_genomes), RefSeq Representative genomes (refseq_representative_genomes), Human RefSeqGene sequences (RefSeq_Gene), Expressed sequence tags (est), Genome survey sequence (gss), High throughput genomic sequences (HTGS), Patent sequences (pat), Protein Data Bank (pdb), Human ALU repeat elements (alu_repeats), Sequence tagged sites (dbsts), NCBI Genomes (chromosome), Whole-genome shotgun contigs (wgs), Transcriptome Shotgun Assembly (TSA), 16S ribosomal RNA sequences (Bacteria and Archaea), Sequence Read Archive (SRA), Reference genomic sequences (refseq_genomic)[4].

Program selection: This section allows the user to select the comparison method either to compare a query sequence with highly

4 https://blast.ncbi.nlm.nih.gov/Blast.cgi?PROGRAM=blastn&PAGE_TYPE=BlastSearch&LINK_LOC=blasthome

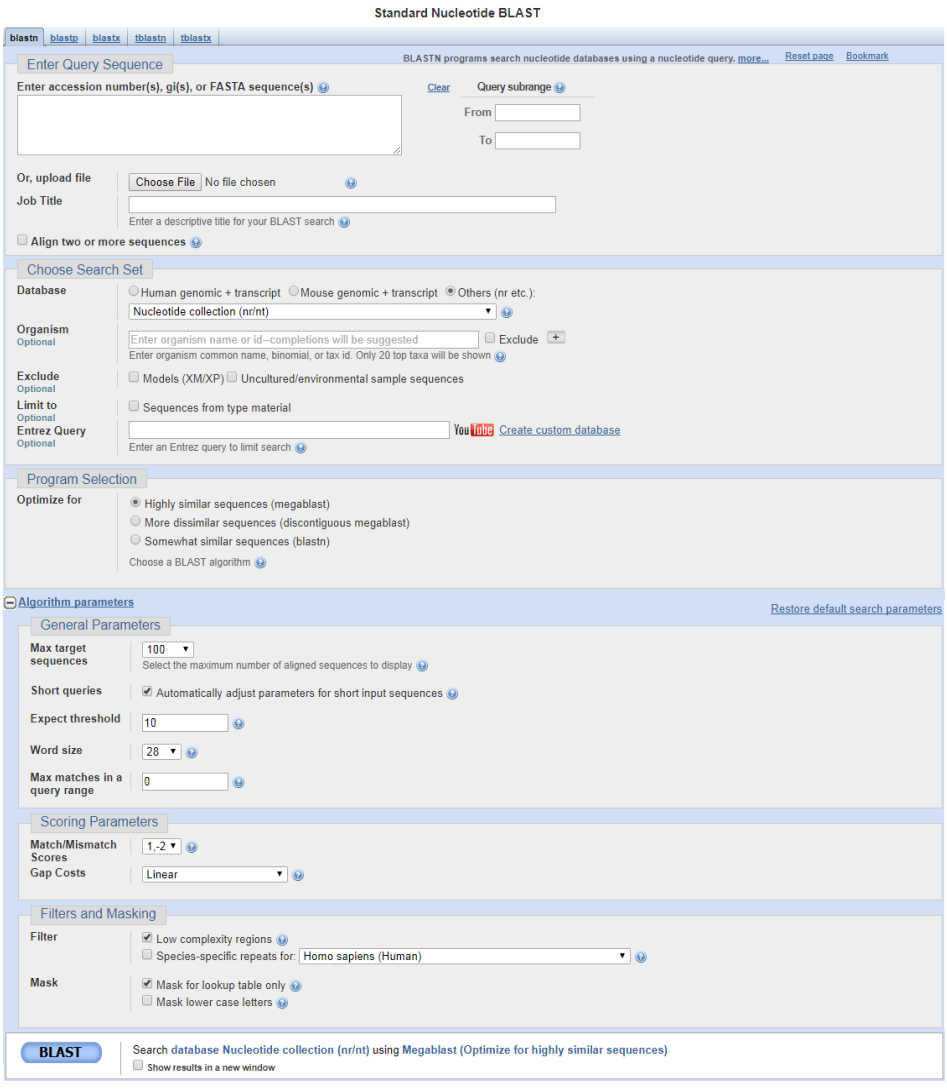


Figure 4: BLASTN GUI which search nucleotide databases using a nucleotide query.

similar sequences (megablast) [11], or with more dissimilar sequences (discontiguous megablast), or with somewhat similar sequences (blastn).

Algorithm parameters: This section is optional and allows the user to modify the default values of the following parameters: maximum number of aligned sequences to display, expected threshold value, word size value, match/mismatch scores, and gap costs. For each parameter shown in Figure 4, there is a small question mark icon the user can click on to find out more description about that parameter. The output of BLASTN is all alignments (hits) listed in increasing order of E-value (expected threshold) which measures the hit quality (smaller numbers mean better hits). It is displayed in HTML format. The result page contains three parts: a graphical format with hits founded as illustrated in Figure 5a, a table with hits sequence identifiers and scoring data as illustrated in Figure 5b, and alignments for the query sequence and hits as illustrated in Figure 5c. The output can be downloaded to the local machine as text file, GenBank, CSV, XML, or ASN file.

i2b2

As a part of NIH (National Institutes of Health) plan, the mission of NCBC (National Center for Biomedical Computing)[5] is to develop and implement innovative software programs that are needed in biomedical research. One of the projects sponsored by NCBC is - i2b2, which is an open source scalable informatics framework that provides clinical investigators the necessary software tools that they need to collect, manage, and combine medical data and research data. The platform design is scalable and its architecture can be extended using additional plug-ins functionalities. Developers are able to create custom plug-ins based on the needs of clinical and research investigators. The platform enables the integration of patients' clinical and genomics data from heterogeneous resources. It is packaged with a query tool that can be used to query patient information and can provide the users with de-identified data of a group of patients meeting certain inclusion or exclusion criteria. Some major applications supported by i2b2 include cohort management [12], which consists of finding a group of patients sharing common criteria for analysis, Population-based studies [13], which join the data of multiple numbers of centers to include large number of participants of a wider range of population groups. This platform is widely used by various academic medical centers and hospitals around the world [14].

i2b2 Users: The i2b2 platform is designed for the following users:

- Clinical investigators: who need to collect and manage project-related clinical research data.
- Bioinformatics scientists: who need to customize the flow of biomedical data and interactions.
- Bio-computational software developers: who are responsible for developing new software services that can be integrated into the i2b2 platform [2].

i2b2 Architecture: The i2b2 architecture is based on Service Oriented Architecture (SOA) and contains a set of interconnected cells / software modules as a hive that has common messaging protocol and interact using web services and XML message. Each cell can be developed independently for a specific goal and can be integrated to the hive to enhance the functionality of i2b2.

5 http://www.ncbcs.org/

Figure 5a: Graphical overview of BLASTN result.

Alignments | Download | GenBank | Graphics | Distance tree of results

Description	Max score	Total score	Query cover	E value	Ident	Accession
Homo sapiens mRNA for prepro cortistatin like peptide, complete cds	680	680	100%	0.0	100%	AB000263.1
Homo sapiens cortistatin, mRNA (cDNA clone IMAGE:5194613), partial cds	669	669	100%	0.0	99%	BC040034.1
Homo sapiens full length insert cDNA clone ZD80A01	669	669	100%	0.0	99%	AF086433.1
Homo sapiens preprocortistatin (Cort) mRNA, complete cds	664	664	100%	0.0	99%	AF013252.1
Homo sapiens cortistatin (CORT), mRNA	658	658	100%	0.0	99%	NM_001302.4
Homo sapiens cortistatin, mRNA (cDNA clone MGC:149757 IMAGE:40118117), complete cds	641	641	95%	3e-180	99%	BC119725.1
Homo sapiens cortistatin, mRNA (cDNA clone MGC:149756 IMAGE:40118115), complete cds	636	636	95%	1e-178	99%	BC119724.1
PREDICTED: Gorilla gorilla gorilla cortistatin (CORT), mRNA	630	630	100%	6e-177	98%	XM_004024635.2
PREDICTED: Pan paniscus cortistatin-like (LOC103786711), mRNA	630	630	100%	6e-177	98%	XM_008975625.2
Pan troglodytes cortistatin (CORT), mRNA	630	630	100%	6e-177	98%	NM_001204893.1
Homo sapiens DNA44175 CORT (UNQ307) mRNA, partial cds	590	590	86%	1e-164	100%	AY358561.1
Synthetic construct DNA, clone: pFN21AE1641, Homo sapiens CORT gene for cortistatin, without stop codon, in Flexi system	586	586	86%	1e-163	99%	AB590589.1
Synthetic construct Homo sapiens clone ccsbBroadEn_00347 CORT gene, encodes complete protein	582	582	87%	2e-162	99%	KJ890953.1

Figure 5b: Tabular overview of BLASTN result including hits identifiers and scoring data.

Download | GenBank | Graphics ▼ Next ▲ Previous Descriptions

Homo sapiens cortistatin, mRNA (cDNA clone IMAGE:5194613), partial cds

Sequence ID: BC040034.1 Length: 701 Number of Matches: 1

Range 1: 49 to 416 GenBank Graphics ▼ Next Match ▲ Previous Mat

Score	Expect	Identities	Gaps	Strand
669 bits(362)	0.0	366/368(99%)	0/368(0%)	Plus/Plus

```
Query  1    ACAAGATGCCATTGTCCCCCGGCCTCCTGCTGCTGCTGCTCTCCGGGGCCACGGCCACCG  60
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  49   ACAAGATGCCATTGTCCCCCGGCCTCCTGCTGCTGCTGCTCTCCGGGGCCACGGCCACCG  108

Query  61   CTGCCCTGCCCCTGGAGGGTGGCCCCACCGGCCGAGACAGCGAGCATATGCAGGAAGCGG  120
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  109  CTGCCCTGCCCCTGGAGGGTGGCCCCACCGGCCGAGACAGCGAGCATATGCAGGAAGCGG  168

Query  121  CAGGAATAAGGAAAAGCAGCCTCCTGACTTTCCTCGCTTGGTGGTTTGAGTGGACCTCCC  180
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  169  CAGGAATAAGGAAAAGCAGCCTCCTGACTTTCCTCGCTTGGTGGTTTGAGTGGACCTCCC  228

Query  181  AGGCCAGTGCCGGGCCCCTCATAGGAGAGGAAGCTCGGGAGGTGGCCAGGCGGCAGGAAG  240
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  229  AGGCCAGTGCCGGGCCCCTCATAGGAGAGGAAGCTCGGGAGGTGGCCAGGCGGCAGGAAG  288

Query  241  GCGCAccccccAGCAATCCGCGCGCCGGGACAGAATGCCCTGCAGGAACTTCTTCTGGA  300
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  289  GCGCACCCCCCCAGCAATCCGCGCGCCGGGACAGAATGCCCTGCAGGAACTTCTTCTGGA  348

Query  301  AGACCTTCTCCTCCTGCAAATAAAACCTCACCCATGAATGCTCACGCAAGTTTAATTACA  360
            |||||||||||||||||||||||||||||||||||||||||||||| |||| |||||||
Sbjct  349  AGACCTTCTCCTCCTGCAAATAAAACCTCACCCATGAATGCTCACGCAAGTGTAATGACA  408

Query  361  GACCTGAA  368
            ||||||||
Sbjct  409  GACCTGAA  416
```

Related Information

Gene - associated gene details

GEO Profiles - microarray expression data

Map Viewer - aligned genomic context

Figure 5c: Alignment between the query sequence and the hit sequence.

There are two types of cells in the hive: core cells and plug-in cells, as shown in Figure 6. Core cells constitute the back-end infrastructure, i.e., server side components, and establish the basic services of the hive such as managing the hive setup, security, users, projects, files, and data. Plug-in cells are client side components, which consist of an application suite of data querying and mining tools. The platform design is scalable and can be extended to provide independent software services. Each plug-in can be developed independently by different investigators to achieve specific analytic goals. Then, they can be integrated into the hive to enhance the functionality of i2b2.

i2b2 Data Model: The i2b2 platform was created to host many projects and each project has its own data warehouse, i.e., Clinical Research Chart (CRC) cell. The data warehouse was designed using the star schema [15] as illustrated in Figure 7. The i2b2 star schema consists of one fact table and four dimension tables. The fact table **OBSERVATION_FACT** presents the quantitative data that is the result of some query. The dimension tables are a detailed description of the fact table. **PATIENT_DIMENSION** includes patient demographics fields such as birth date, gender, and age. **VISIT_DIMENSION** represents sessions (also called visit, event, or encounter) where observations were made. **CONCEPT_DIMENSION** contains diagnoses, procedures, medications and lab tests. **PROVIDER_DIMENSION** represents a doctor or provider at an institution.

i2b2 Client Applications: The client side applications available to users for query and analysis include:

- The Workbench[6], which is a fat-client application, which can be

6 https://www.i2b2.org/software/

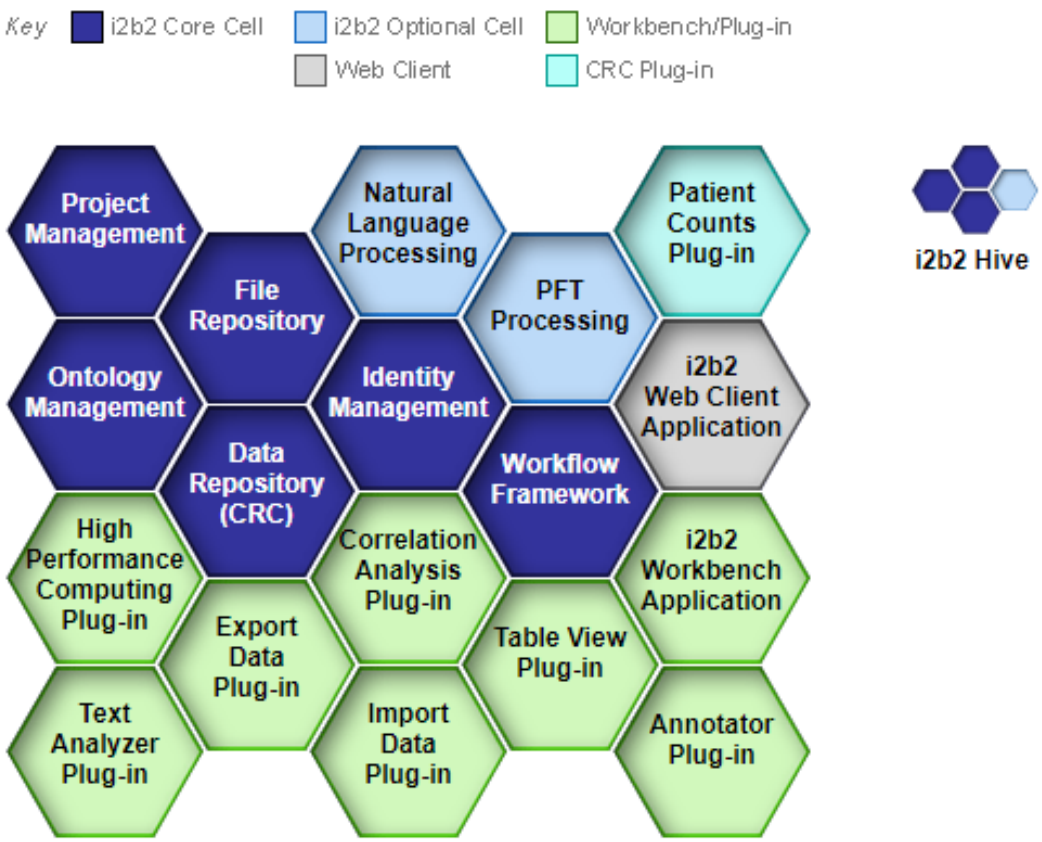


Figure 6: i2b2 Architecture.

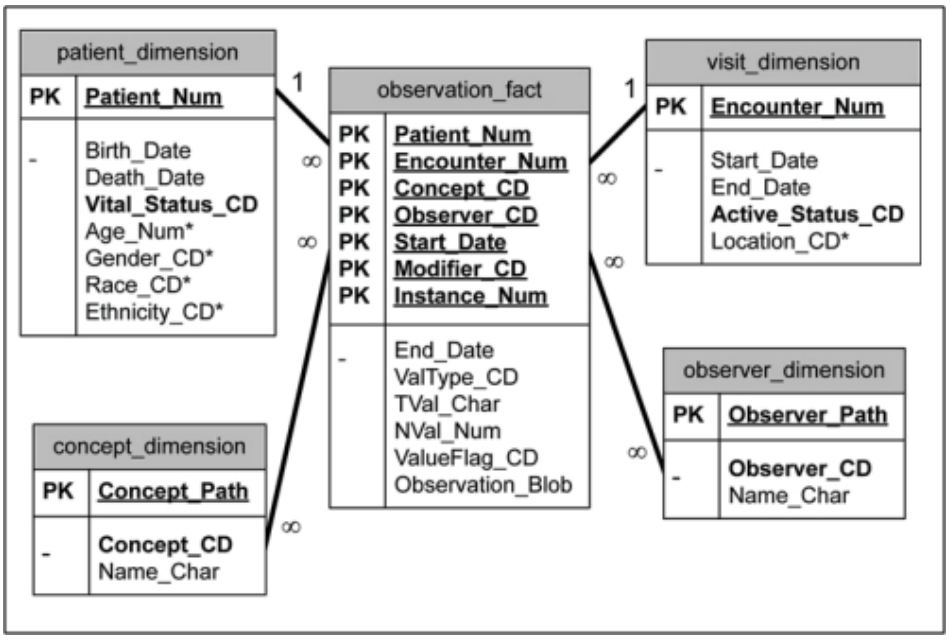


Figure 7: i2b2 Data Model.

installed on the user's desktop computer.

- The Web client[7], which is a thin-client application, which can run on any modern web browser.

Proposed BLAST-i2b2 Approach

The main issue with the current BLAST platform is the storage of BLAST results. Biological sequence databases are periodically updated; thus, BLAST results may be different if the user runs the BLAST against different versions of a specific biological sequence database. In another words, if the user runs BLAST against a biological sequence database, the current platform displays the result and allow the user to download these results to the local machine. If that specific target biological sequence database is updated, the previously downloaded results may be old and the user needs to run the BLAST against the updated version to get an updated result. In this case, the user should enter the query sequence again, the BLAST will be run against the whole sequences in the database each time the database is updated, which is a waste of time because the target database is usually slightly updated. So, we need to run the BLAST against the updated sequences only instead of running it against the whole sequences in order to reduce BLAST run time. To solve this problem, we designed the data warehouse to store BLAST result. In this scenario, if the user runs BLAST against a biological sequence database, the system will display the result and allow the user to save these results into the data warehouse. If that specific target biological sequence database is updated later and the user requests to have an updated result, BLAST runs against the updated sequences only which considerably reduces BLAST run time.

We extended the i2b2 platform by developing a new plugin, ***BLAST-i2b2***. *BLAST-i2b2* provides two main features: saving and updating BLAST results. In order to implement this new tool, we incorporated the BLAST tool into the i2b2 platform, and also added the required features for saving and updating the BLAST results. In this process, the following steps were performed:

- First, we selected which client to use, i2b2 web client or i2b2 workbench. The i2b2 web client was selected because it is easily accessible from web browsers.
- Second, we incorporated NCBI BLAST into the i2b2 platform. In order to do that, we downloaded BLAST executable (command line) programs[8] from NCBI web page. We also downloaded some of BLAST databases[9]. When the user runs BLAST, the system executes BLAST executable against the locally downloaded BLAST databases.
- Third, we modified the *BLAST-i2b2*. In order to save BLAST results, we designed the data warehouse, which stores the query parameters and the query results. To update the BLAST results; we first run a script every three days to download the latest version of the BLAST databases locally. We also give the user the ability to run this feature manually. Second, we allow the user to review queries history. The user can select one of the previous queries and request an update. In this case, if a new BLAST database version is downloaded; the system extracts both deleted and newly added sequences, runs BLAST against the newly added sequences, deletes the stored sequences which are matched the deleted sequences, and updates the result. The next section gives a detailed description of the save and the update processes.

Save process

An innovative functionality, saving the BLAST results, was added to the proposed tool. If the user enters a query sequence and selects a target database; the *BLAST-i2b2* runs the BLAST algorithm to find the similarities between the query sequence and each sequence in the target database, displays the BLAST results, and gives the user the ability to save these results, as shown in Figure 8. The benefits of saving BLAST results are:

- An up-to-date storage of BLAST results as well as the storage of queries in a local data warehouse, which is directly connected to the targeted database for periodic updates.
- Faster access to the stored results and options for running and customizing the previous queries.

7 https://www.i2b2.org/webclient/

8 ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST

9 ftp://ftp.ncbi.nlm.nih.gov/blast/db

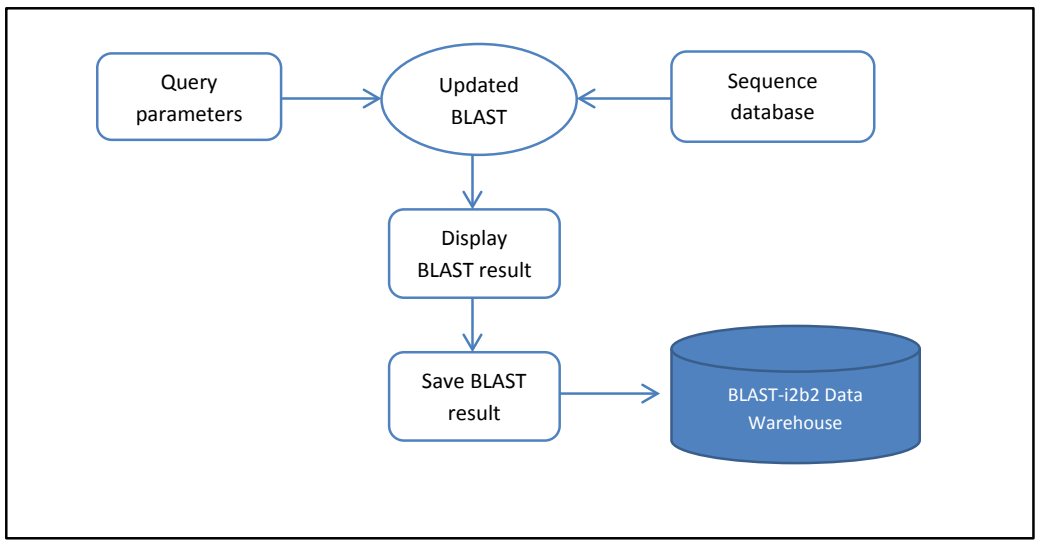


Figure 8: The modified BLAST with "save" feature.

- The ability to utilize the data for data mining, analysis and visualization in the future using the set of client side tools provided by the i2b2 platform

- Reusability of the stored data in the data warehouse for future analytical experimentation and research purposes.

BLAST-i2b2 saves both the query parameters including the query sequence, the target database name and version, the expected threshold value, and the word size value, and the BLAST results of this query. The *BLAST-i2b2* data warehouse design is described in Section 3.3.

Update process

In the original BLAST algorithm process, the user enters a query sequence, selects a target database, and runs BLAST. The BLAST algorithm compares the submitted query sequence to all sequences in the target database and displays the BLAST results as illustrated in Figure 9a. If a new version of that target database is downloaded, the user must enter the query parameters again (because they are not saved) to get updated results. In this case, the BLAST repeats its process and compares the re-submitted query sequence to all sequences in the updated target database as illustrated in Figure 9b, which may take a longer time. Therefore, we propose the 'save process', which saves the query sequence and BLAST results to reduce BLAST run time when the target database is updated.

In *BLAST-i2b2*; the user enters a query sequence, selects a target database, and runs BLAST. *BLAST-i2b2* compares the submitted query sequence to all sequences in the target database, displays the BLAST results, and allows the user to save those results, as illustrated in Figure 10a.

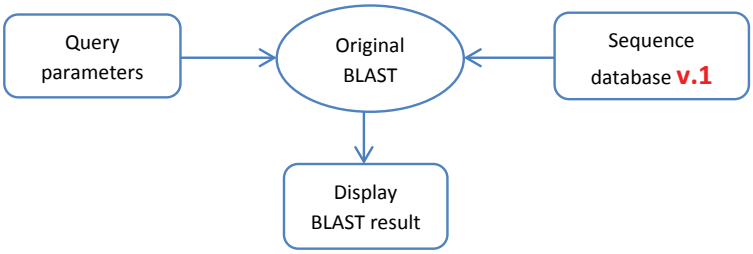
If a new version of that target database is downloaded, the system informs the user, who can simply update the results. In this case, BLAST compares the saved query sequence with the modified sequences only in the updated version of the target database instead of comparing it to all sequences that will reduce the BLAST execution time as illustrated in Figure 10b. To implement the update process, we used the following strategy:

• **Extract the sequence GI number**: The modification of a database could be an added sequence, a deleted sequence, or a modified sequence. Each sequence in a database is identified by GI number. In order to find the modifications between database versions (the old version which the user run a query against and the new version), we extract the GI number for all sequences from both the old and new database versions into two text files as illustrated in Figure 11a.

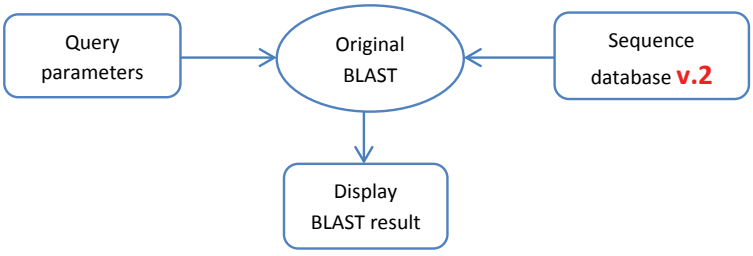
• **Find the modification between database versions:** We compare the two text files to extract the newly added sequences and the deleted sequences into another two files, as illustrated in Figure 11b. One file contains the newly added sequences GI numbers that exist in the new database version and do not exist in the old database version. The other file contains the deleted sequences GI numbers that exist in the old database version and do not exist in the new database version. We do the comparison using UNIX "JOIN" utility[10].

• **Run BLAST against new sequences:** *BLAST-i2b2* runs BLAST against the newly added sequences and displays new BLAST results. If the user wants to save the new BLAST results, the system will performs these two steps:

10 https://linux.die.net/man/1/join

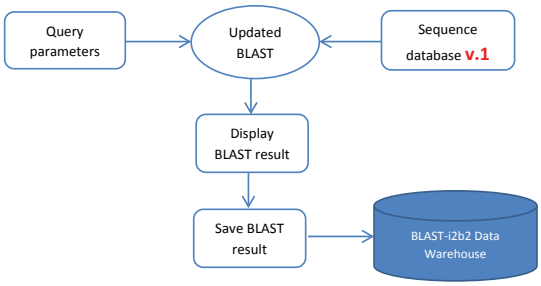


(a) Run BLAST against all sequences in the sequence database (v.1)

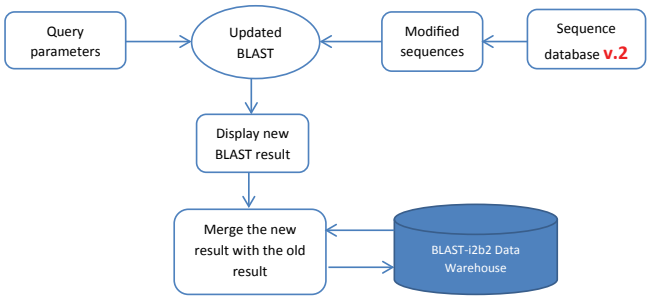


(b) Re-run BLAST against all sequences in the sequence database (v.2) to get updated results

Figure 9: The original BLAST.

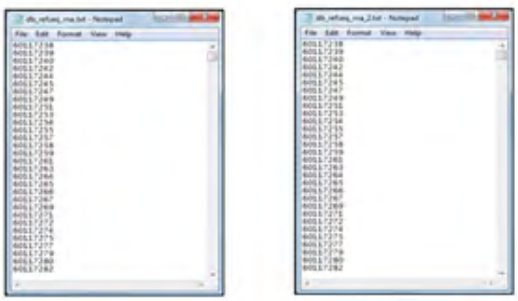


(a) Run BLAST against all sequences in the sequence database (v.1) and save the result in the data warehouse

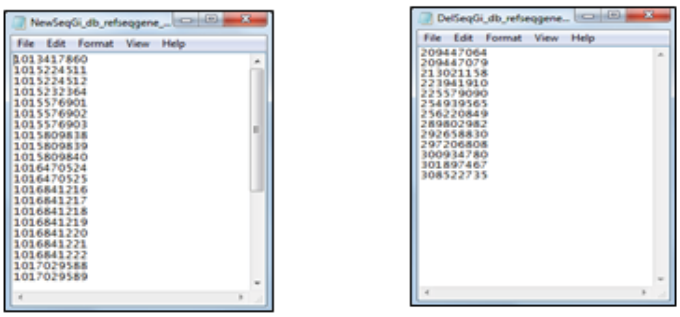


(b) If a new version of the sequence database is downloaded (v.2), BLAST will run against the modified sequences only

Figure 10: The modified BLAST.



(a) Extract GI numbers for both database versions (v.1) and (v.2)



(b) Extract the newly added sequences and the deleted sequences into two files

Figure 11: Extract GI numbers from v.1 and v.2 to find the newly sequences and the deleted sequences.

• It merges the new results with the stored BLAST results.

• It checks for matching between the deleted sequences and the old stored BLAST results. If there are any matches, the system deletes those results from the data warehouse.

BLAST-i2b2 data warehouse design

The *BLAST-i2b2* data warehouse is designed as star schema with four tables: one fact table, BlastResultFact; and three dimension tables, BlastQueryDim, BlastDatabaseDim, and BlastSequenceDim as illustrated in Figures 12. The BlastResultFact table stores the BLAST results. This table contains a detailed description of each hit with the submitted query id; target database id, name, and version as illustrated in Figures 13. The BlastQueryDim table stores each submitted query parameters, which consists of the query sequence, query date, target database name and version, word size, and expect threshold as illustrated in Figure 14. Three extra variables: NewSeqDatabase, DelSeqDatabase, and UpdatedVersion are assigned to NULL unless the user requests an update. In this case, these variables are filled as follows: NewSeqDatabase is filled with the file name that contains the newly sequences that exist in the UpdatedVersion field and do not exist in DatabaseVersion field; DelSeqDatabase is filled with the file name that contains the deleted sequences which exist in DatabaseVersion field and do not exist in UpdatedVersion field; UpdatedVersion will be filled with the updated database version number. The BlastDatabaseDim table, shown in Figure 15, stores a list of all the BLAST databases the user can search with version number and download date.

The BlastSequenceDim table stores information of each version of BLAST databases including: database Id, database version number, database downloading date, database text file name which includes gi numbers of all sequences of that version as illustrated in Figures 16.

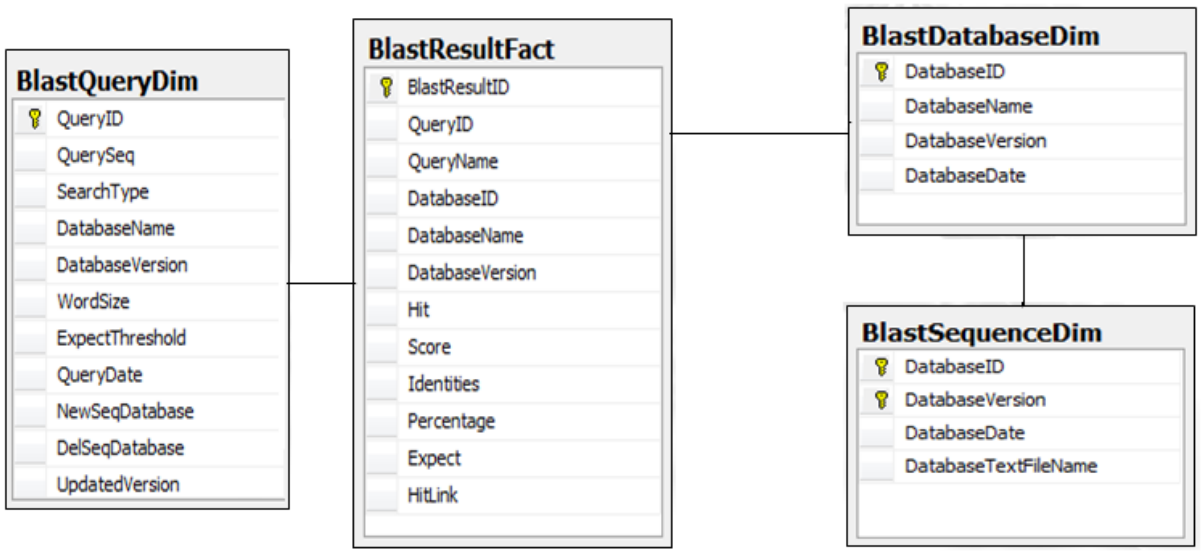


Figure 12: *BLAST-i2b2* Data warehouse.

	BlastResultID	QueryID	DatabaseID	DatabaseName	DatabaseV	Hit	Score	Identities	Percentage	Expect	HitLink
1	1	22	2	refseqgene	1	gi|26447242	181.00	244/336 (368)	72.00	1e-043	http://www.ncbi.nlm.ni...
2	2	22	2	refseqgene	1	gi|7061133	168.00	241/336 (368)	71.00	9e-040	http://www.ncbi.nlm.ni...
3	3	22	2	refseqgene	1	gi|26471297	141.00	223/316 (368)	70.00	1e-031	http://www.ncbi.nlm.ni...
4	4	22	2	refseqgene	1	gi|48583629	127.00	148/199 (368)	74.00	3e-027	http://www.ncbi.nlm.ni...
5	5	22	2	refseqgene	1	gi|26977454	116.00	140/189 (368)	74.00	5e-024	http://www.ncbi.nlm.ni...
6	6	22	2	refseqgene	1	gi|8811316	109.00	141/194 (368)	72.00	8e-022	http://www.ncbi.nlm.ni...

Figure 13: BlastResultFact table records.

QueryID	QuerySeq	SearchType	DatabaseName	DatabaseV	WordSize	ExpectTh	QueryDate	NewSeqDatabase	DelSeqDatabase	UpdatedV
1	ACAAGATGCC	Basic search	gss_annot	1	11	10	2016-04-24	NULL	NULL	NULL
2	ACAAGATGCC	Basic search	refseqgene	1	11	10	2016-04-25	NewSeq_db_refseqgene_1db_refseqgene_2	DelSeq_db_refseqgene_1db_refseqgene_2	2

Figure 14: BlastQueryDim table records.

Results and Discussion

We tested the performance of *BLAST-i2b2* to assess its capability. Since the key feature of *BLAST-i2b2* that it stores BLAST results to reduce BLAST search time by providing the update process; the performance test focuses on the execution time of BLAST against the updated BLAST database. We conducted three different experiments; each one is done against different database size. The following section present the test parameters including the submitted query sequence, searched BLAST databases, expected threshold value, and word size.

Experimental Settings

Query sequence

ACAAGATGCCATTGTCCCCCGGCCTCCTGCTGCTGCTGCTCTCCGGGGCCAC
GGCCACCGCTGCCCTGCCCCTGGAGGGTGGCCCCACCGGCCGAGACAGCGA
GCATATGCAGGAAGCGGCAGGAATAAGGAAAAGCAGCCTCCTGACTTTCCTC
GCTTGGTGGTTTGAGTGGACCTCCCAGGCCAGTGCCGGGCCCCTCATAGGAG
AGGAAGCTCGGGAGGTGGCCAGGCGGCAGGAAGGCGCACCCCCCCAGCAAT
CCGCGCGCCGGGACAGAATGCCCTGCAGGAACTTCTTCTGGAAGACCTTCTC
CTCCTGCAAATAAAACCTCACCCATGAATGCTCACGCAAGTTTAATTACAGA
CCTGA

BLAST databases

Small sized database used in experiment 1.

- RefSeqGene v.1 (121,606 KB=118 MB).
- RefSeqGene v.2 (121,697 KB).

Medium sized database used in experiment 2.

- EST_mouse v.1 (570,478 KB=560 MB).
- EST_mouse v.2 (573,310 KB).

Large sized database used in experiment 3.

- RefSeq_RNA v.1 (7,921,657 KB=7 GB).
- RefSeq_RNA v.2 (7,990,182 KB).
- Expected threshold value (The number of different alignments that are expected to be matched by chance): 10.
- Word size: 11 (system default value).

The performance test was conducted using two scenarios for each database size.

- First Scenario: Run the query sequence against the first version of a BLAST database without saving BLAST results. Then, run the same query sequence against the second version of the same BLAST database.
- Second Scenario: Run the query sequence against the first version of a BLAST database and save BLAST results. Then, update the saved BLAST results using the BLAST-i2b2 update algorithm.

Experimental results

Experiment 1: Testing the execution time of the two scenarios with the small sized database: RefSeqGene (121,606 KB=118 MB - 121,697 KB). In Table 1, the results for the first scenario show that running BLAST against the whole sequences in RefSeqGene v.1 database takes 4.15 sec. If we do not save the BLAST results and a new version of RefSeqGene database is downloaded, running BLAST against the whole sequences in RefSeqGene v.2 database takes 4.18 sec. While the results for the second scenario show that running BLAST against the whole sequences in RefSeqGene v.1 database takes 4.15 sec. If we now save the BLAST results and a new version of RefSeqGene database is downloaded, updating BLAST result takes only 0.65 sec which reduces the run time to more than 84%.

Experiment 2: Testing the execution time of the two scenarios with the medium sized database: EST_mouse (570,478 KB=560 MB - 573,310 KB). In Table 2, the results for the first scenario show that running BLAST against the whole sequences in EST_mouse v.1 database takes 25.85 sec. If we do not save the BLAST results and a new version of EST_mouse database is downloaded, running BLAST against the whole sequences in EST_mouse v.2 database takes 34.04 sec. While the results for the second scenario show that running BLAST against the whole sequences in EST_mouse v.1 database takes 25.85 sec. If we now save the BLAST results and a new version of EST_mouse database is downloaded, updating BLAST result takes only 2.33 sec which reduces the run time to more than 93%.

Experiment 3: Testing the execution time of the two scenarios with large sized database: RefSeq_RNA (7,921,657 KB=7 GB - 7,990,182 KB). In Table 3, the results for the first scenario show that running BLAST against the whole sequences in RefSeq_RNA v.1 database takes 502 sec. If we do not save the BLAST results and a new version of RefSeq_RNA database is downloaded, running BLAST against the whole sequences in RefSeq_RNA v.2 database takes 474 sec. While the results for the second scenario show that running BLAST against the whole sequences in RefSeq_RNA v.1 database takes 502 sec. If we now save the BLAST results and a new version of RefSeq_RNA database is downloaded, updating BLAST result takes only 14 sec which reduces the run time to more than 97%.

Discussion

The experimental results show that the *BLAST-i2b2* update process reduces the search time by 84% for the small sized database: RefSeqGene, by 93% for the medium sized database: EST_mouse, and by 97% for the large sized database: RefSeq_RNA. As shown in Figure 17, the left bars of the figure present the run time against the small sized

	DatabaseID	DatabaseName	DatabaseVersion	DatabaseDate
1	1	gss_annot	1	2016-04-24
2	2	test_na_db	1	2016-03-01
3	3	16SMicrobial	2	2016-04-26
4	4	refseqgene	2	2016-04-26

Figure 15: BlastDatabaseDim table records.

	DatabaseID	DatabaseVersion	DatabaseDate	DatabaseTextFileName
1	1	1	2016-03-01	db_gss_annot.00_1
2	1	2	2016-04-26	db_gss_annot.00_2
3	2	1	2016-03-01	db_test_na_db_1
4	2	2	2016-04-26	db_test_na_db_2
5	3	1	2016-04-01	db_16SMicrobial_1
6	3	2	2016-04-26	db_16SMicrobial_2
7	4	1	2016-04-04	db_refseqgene_1
8	4	2	2016-04-26	db_refseqgene_2

Figure 16: BlastSequenceDim table records.

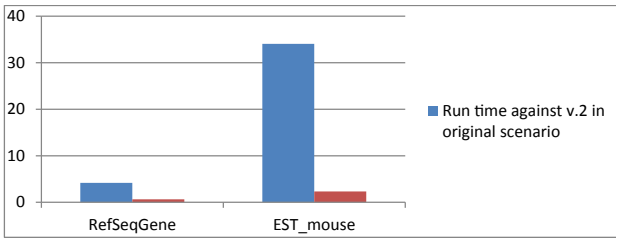


Figure 17: BLAST run time against v.2 in the original scenario and against v.2 in the update scenario for the small and the medium database size, RefSeqGene and EST_mouse respectively.

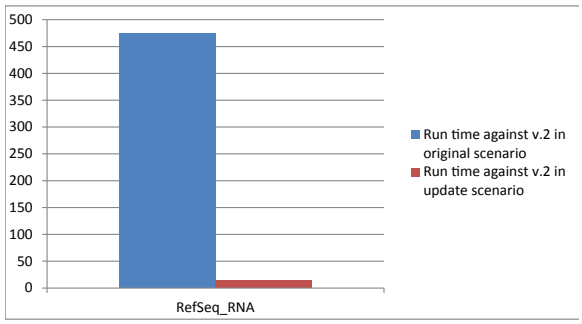


Figure 18: BLAST run time against v.2 in the original scenario and against v.2 in the update scenario for the big database size.

Scenario	Save Result	Test	Database File Size	Execution time
First scenario	×	Run BLAST against the whole RefSeqGene v.1	121,606 kb	4.15 sec
		Run BLAST against the whole RefSeqGene v.2	121,697 kb	4.18 sec
Second scenario	√	Run BLAST against the whole RefSeqGene v.1	121,606kb	4.15 sec
		Update BLAST results	515 kb	0.65 sec

Table 1: Experiment 1 - Testing the execution time of blast-i2b2 against the small database volume.

Scenario	Save Result	Test	Database File Size	Execution time
First scenario	×	Run BLAST against the whole EST_mouse v.1	570,478 kb	25.85 sec
		Run BLAST against the whole EST_mouse v.2	573,310 kb	34.04 sec
Second scenario	√	Run BLAST against the whole EST_mouse v.1	570,478 kb	25.85 sec
		Update BLAST results	2,832 kb	2.33 sec

Table 2: Experiment 2 - Testing the execution time of BLAST against the medium database volume.

Scenario	Save Result	Test	Database File Size	Execution time
First scenario	×	Run BLAST against the whole RefSeq_RNA v.1	7,921,657 kb	502 sec
				=8.22 min
		Run BLAST against the whole RefSeq_RNA v.2	7,990,182 kb	474 sec
				=7.54 min
Second scenario	√	Run BLAST against the whole RefSeq_RNA v.1	7,921,657 kb	502 sec
				=8.22 min
		Update BLAST results	362,052 kb	14 sec

Table 3: Experiment 3 - Testing the execution time of BLAST against the big database volume.

database in the update process and the right bars present the run time against the medium sized database. Running BLAST against the whole sequences in RefSeqGene database v.2 takes 4.18 sec and updating the result takes only 0.65 sec. Running BLAST against the whole sequences in EST_mouse database v.2 takes 34.04 sec and updating the result takes only 2.33 sec. As shown in Figure 18, running BLAST against the whole sequences in RefSeq_RNA database v.2 takes 474 sec and updating the result takes only 14 sec. From these results, we can conclude that *BLAST-i2b2* update process enhances BLAST by reducing its run time and storing the results for further analysis.

Conclusions

In this paper, we introduced biological sequences related concepts, such as sequence similarity and its importance and how to find this similarity using different sequence alignment methods. We discussed the BLAST and its functionality in details. We provided an overview of i2b2 architecture and features. Then, we described the proposed approach of developing *BLAST-i2b2* using i2b2 platform. This new tool provides researchers the ability to compare DNA sequences with some of NCBI DNA databases. We presented *BLAST-i2b2* methodology and the data warehouse design. The paper showed that *BLAST-i2b2* enhanced the functionality for both the BLAST and the i2b2 platform. From the BLAST perspective, the major added features included: the ability to store BLAST results for further future analysis and the ability to update searched query results whenever there are updates in the target database. We modified the BLAST by allowing it to compare the query sequence with the modified sequences only in the target database instead of comparing it with all sequences. From the i2b2 perspective, a new *BLAST-i2b2* plug-in was added to the i2b2 platform that is capable of storing the BLAST search queries and results locally and reusing them in an innovative way to enhance the BLAST functionality. We tested the performance of *BLAST-i2b2* update process against three different database sizes. The experiments results showed that *BLAST-i2b2* update process reduces the BLAST run time to 84% for the small database size, 93% for the medium database size, and 97% for the large sized database. The work in progress focuses on extending the BLAST-i2b2 functionality by importing, mapping and customization of locally stored search results for i2b2 web client application. The future work aims for the utilization and reusability of these stored results for data mining and analytics purpose that will enable users to query, mine, analyze and visualize the BLAST search results using BLAST-i2b2 platform. Due to the generic nature of proposed approach, in the future, the BLAST-i2b2 can also be extended to search the other available DNA databases as well as the protein sequence databases.

References

1. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. J Mol Biol 215: 403-410.
2. Murphy SN, Mendis M, Hackett K, Kuttan R, Pan W, et al. (2007) Architecture of the Open-source Clinical Research Chart from Informatics for Integrating Biology and the Bedside. AMIA Annu Symp Proc 11: 548-552.
3. Mount DW (2004) Bioinformatics: Sequence and Genome Analysis. 2nd edn. Cold Spring Harbor, NY: Cold Spring Harbor Laboratory Press, USA.
4. Dayhoff MO, Schwartz RM, Orcutt BC (1978) A model of evolutionary change in proteins. Atlas of Protein Sequence and Structure 5: 345-351.
5. Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks. Proceedings of the National Academy of Sciences of the United States of America 89: 10915-10919.
6. Korf I, Yandell M, Bedell J (2003) BLAST. 1st edn. Sebastopol, CA: O'Reilly Media, USA.
7. Needleman SB, Wunsch CD (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology 48: 443-453.
8. Smith TF, Waterman MS (1981) Identification of common molecular subsequences. Journal of Molecular Biology 147: 195-197.
9. GenBank Sample Record. Available at: https://www.ncbi.nlm.nih.gov/Sitemap/samplerecord.html
10. Pearson WR, Lipman DJ (1988) Improved tools for biological sequence comparison. Proc Natl Acad Sci 85: 2444-2448.
11. Zhang Z, Schwartz S, Wagner L, Miller W (2000) A greedy algorithm for aligning DNA sequences. J Comput Biol 7: 203-214.
12. Segagni D, Tibollo V, Dagliati A, Perinati L, Zambell A, et al. (2011) The ONCO-I2b2 project: Integrating biobank information and clinical data to support translational research in oncology. Studies in Health Technology and Informatics 169: 887-891.
13. Anderson N, Abend A, Mandel A, Geraghty E, Gabriel D, et al. (2012) Implementation of a deidentified federated data network for population-based cohort discovery. J Am Med Inform Assoc 19: e60-e67.
14. i2b2 Installations. Available at: https://www.i2b2.org/work/i2b2_installations.html
15. Dennis A, Wixom BH, Roth RM (2014) Systems Analysis and Design. 6th edn. New Jersey, US. Wiley.
16. Murphy SN, Weber G, Mendis M, Gainer V, Chueh HC, et al. (2010) Serving the enterprise and beyond with informatics for integrating biology and the bedside (i2b2). J Am Med Inform Assoc 17: 124-130.
17. i2b2 Community Wiki. Available at: http://community.i2b2.org/wiki/display/dash/i2b2+Dashboard+Home/