# NCBI/GenBank BLAST Output XML Parser Tool

**David Ream[1] & Andor J Kiss[1,2]**

[1]Department of Microbiology and [2]Center for Bioinformatics and Functional Genomics, Miami University, Oxford, Ohio, 45056, USA

*Contact*: Andor J Kiss, Center for Bioinformatics and Functional Genomics, 086 Pearson Hall – 700 East High Street, Miami University, Oxford, Ohio 45056, USA.
eMAIL: kissaj@MiamiOH.edu ,Tel:+1 (513) 529-4280, Fax:+1 (513) 529-2431

## Abstract:

We describe a small freely available computer script to extract 'real world' sequence descriptions from the BLASTX results from sequences generated by the stand-alone `ncbi-blast-2.2.26` suite of tools (available from NCBI/GenBank). Our Python (2.7) script is intended to make name extraction feasible for thousands, of hundreds of thousands, of sequences such as that generated by BLASTX analysis of RNA-Seq (transcriptome) obtained cDNAs from next generation sequencing (NGS) experiments. This script facilitates the interrogation of the large BLASTX output of a transcriptome experiment by familiar tools such as Microsoft Excel, or LibreOffice Calc. The script was written and tested on the Linux operating system (Ubuntu 12.04 LTS), but should work in any Python 2.7 compatible environment. We include some example files and help documentation.

*Keywords*: BLAST, BLASTX, Parsing script, RNA-Seq.

## Introduction & Rationale:

The ability to identify nucleotide and proteins sequences by comparison with previously identified sequences deposited within the GenBank database at the National Center for Biotechnology Information (NCBI) housed at the National Library of Medicine (NLM) within The National Institutes of Health (NIH) in Bethesda, MD, USA is one of the great advances in molecular biology (http://www.ncbi.nlm.nih.gov/). Recent price reductions in both chemistries and widespread instrumentation availability coupled with excellent commercial and open-source software (BioLinux [1]) has enabled modest and/or small research programs to exploit RNA-Seq (transcriptomics) technologies. However, annotation of cDNAs *without* a sequenced genome is a barrier for many biologists. We have devised a simple analysis pipeline that circumvents this barrier thus enabling any biologist to identify any known, conserved protein from their organism of study (Fig. 1). Of course, the caveat in this process is that there must be enough conservation between one's query sequence and a sequence in GenBank to facilitate a match. At the amino acid level, this has turned out to be surprisingly good approach with matches of more than 80% of the novel cDNAs from non-model, non-genome sequenced organisms in our hands. This success rate is largely due to the depth of identified sequences available at GenBank.

NCBI hosts many databases that are focused on a diversity of specialties related to bioinformatics and associated information, and information technology (software). The repository of sequence information, both nucleic and amino acid, database is GenBank, which is mirrored at the European Molecular Biology Laboratory Nucleotide Sequence Database (EMBL) and the DNA Databank of Japan (DDBJ). The worldwide network of mirrored databases together compose the International Nucleotide Sequence Database Collaboration (INSDC) to ensure redundancy and mutual backup protection [2]. NCBI provides an intuitive, free to use graphical user interface (GUI) that enables easy uploading of one's query sequence for searching of the GenBank database. While multiple sequences maybe submitted *via* the web interface, submission of large numbers (*e.g.* several thousand as would be generated from current

NGS technologies [3]) of query sequences can significantly reduce performance and maybe rejected by the NCBI server. A solution to this problem is to download the freely available database(s) one wishes to interrogate and perform the search locally. NCBI provides its Basic Local Alignment Search Tool (BLAST) software as an independent download (Linux, Macintosh and Windows) to accomplish just such a task.

One of the consequences of the age of NCBI's GenBank database system is that the archival file format that was chosen at the time of inception predates today's XML metadata format, and thus the format that was originally chosen was Abstract Syntax Notation One (ASN.1). The ASN.1 format is an extremely flexible format that provides the ability to incorporate large amounts of complex and differing data into an easily read and well-defined format [4]. However, the ASN.1 format presents the biologist who cannot write Python or Perl (computer programming languages) scripts to parse the ASN.1 data files, with a non-trivial barrier to obtaining the biologically relevant information in an easily searchable comma separated values (CSV) format – something that can be read and manipulated by Microsoft Excel, or LibreOffice Calc. The ASN.1 archival format used by NCBI can be used with an NCBI provided utility (<`blast_formatter`>) to reformat the ASN.1 archive file into eleven different types of output (Table 1). Direct output of the NCBI Blast results into any of the eleven outputs is also available, but the reader should be cautioned that bypassing the ASN.1 archive generation step precludes the ability of the <`blast_formatter`> tool from being able to then reformat the non-ASN.1 output into any other given format (Table 1). A typical RNA-Seq (also known as a transcriptome, or a cDNA library) dataset may take upwards of several weeks of computational time when using BLASTX (depending on computing resources and number of sequences queried). BLASTX is the variant of the BLAST tool that takes a nucleotide (DNA) sequence and translates in all three frames on the upper strand and all three frames on the lower strand (as one may not know the correct orientation of the cDNA) and compares it with a curated protein database, typically the 'NCBInr', or non-rendundant protein database. In other words, 10,000 nucleotide sequences (cDNAs) are queried as 60,000 amino acid sequences. Thus, a judicious choice of output formats must be made prior to committing such a large amount of computational resources; alternatively by choosing the ASN.1 BLAST archive format, one can reformat the data in a matter of minutes dependent on the desire or requirements post-BLAST. It should be noted that a copy of the NCBInr database used to perform the initial BLAST needs to be in the same directory as the ASN.1 archive file when the <`blast_formatter`> command is executed.

**Table 1:** List of <blast_format> output alignment view options. Using the <blast_format> tool and the ASN.1 archive as the input file permits reformatting into any of the formats (0~11) below.

| Code | Description |
|---|---|
| 0 | pairwise |
| 1 | query-anchored showing identities |
| 2 | query-anchored no identities |
| 3 | flat query-anchored, show identities |
| 4 | flat query-anchored, no identities |
| 5 | XML Blast output |
| 6 | tabular |
| 7 | tabular with comment lines |
| 8 | Text ASN.1 |
| 9 | Binary ASN.1 |
| 10 | Comma-separated values |
| 11 | BLAST archive format (ASN.1) |

We describe here a simple command line interface (CLI) parsing script written in the widely distributed and freely available Python computing language for the non-coding biologist, such that they may generate an easily read CSV based output of their BLASTX results.

Script Usage:

The parsing script was written in Python 2.7 to parse the following fields (query id, subject id, % identity, alignment length, mismatches, gap opens, q. start, q. end, s. start, s. end, evalue, bit score, subject description) from an XML Blast Output file (`-outfmt 5`) that was in turn generated from an ASN.1

BLAST archive format. The command to generate the XML file was:

```
$ blast_formatter -archive infile.asn1 -outfmt 5 -out outfile.xml
```

Using the XML file facilitated a much simpler and smaller starting file for our Python script to parse into CSV format, as well as permitting viewing of the compact XML file with any appropriate interpreter, or text editor (*e.g.* Emacs, XML Copy Editor, gEdit). Our Python script is available as a compressed archive (gzipped tarball) download from (https://github.com/reamdc1/blast_xml_parse). Once downloaded, one needs to uncompress the archive file before attempting to utilise the script contained within.

We have tested and run the script under Ubuntu Linux & BioLinux, but it should work in any Python 2.7 compatible environment in any operating system (http://www.python.org/download/-releases/2.7/). In Linux and Macintosh OS/X[1], to unzip the archive (`NCBI_XML_parser.tar.gz`), one should place the archive in a folder of the user's choice, and either uncompress it by double clicking on it, or by opening a terminal and entering:

```
$ tar -zxvf NCBI_XML_parser.tar.gz
```

To then execute (run) the parser, make sure that your <*.xml> file (reformatted from the ASN.1 archive file) is in the same directory as the parser script (`blast_xml_parse.py`). The command to execute the parser is:

```
$ Python blast_xml_parse.py -i <infile> -o <outfile> -q <query_file>
```

where: the `-i` FILE is the name of the input XML file (converted from the ASN.1 file).
the `-o` FILE is the name (that you assign) of the output file (CSV format).
the `-q` FILE is the original FASTA file used in the BLASTX query. This file is optional, but in large datasets extremely useful to identify the correct cDNA (FASTA sequence) for further analysis and study.

A help guide is available using the `-h` option (`Python blast_xml_parse.py -h`), which will output the options for usage in the terminal window. The script has internal documentation to facilitate editing (*i.e.* changing the fields) by the user. Our script is released under the BioPython license. The script should take a few minutes to run (depending on the size of your XML file) and should produce two CSV files in the same directory. One will be every 'hit' from your BLAST into a CSV file with the last field '`subject description`' as the real world name for the putative BLASTX identification. The second file will be similar with only the best, or top hit, from each query.

Summary:

We present a simple script to be used in conjunction with a simple approach to BLASTX'ing a large number of cDNA sequences to maximize the extraction of the information retrieved by BLAST for a non-bioinformatics, non-computer coding capable end-user (*e.g.* a biologist). We think that being able to rapidly identify individual sequences in a familiar (Excel, Calc) environment is of tremendous benefit and removes yet another access barrier to newer NGS based technologies that are generating thousands of identifiable sequences from non-model organisms.

[1] In the Microsoft Windows operating system, download the free utility **7-Zip** from http://www.7-zip.org/ .

Acknowledgements:

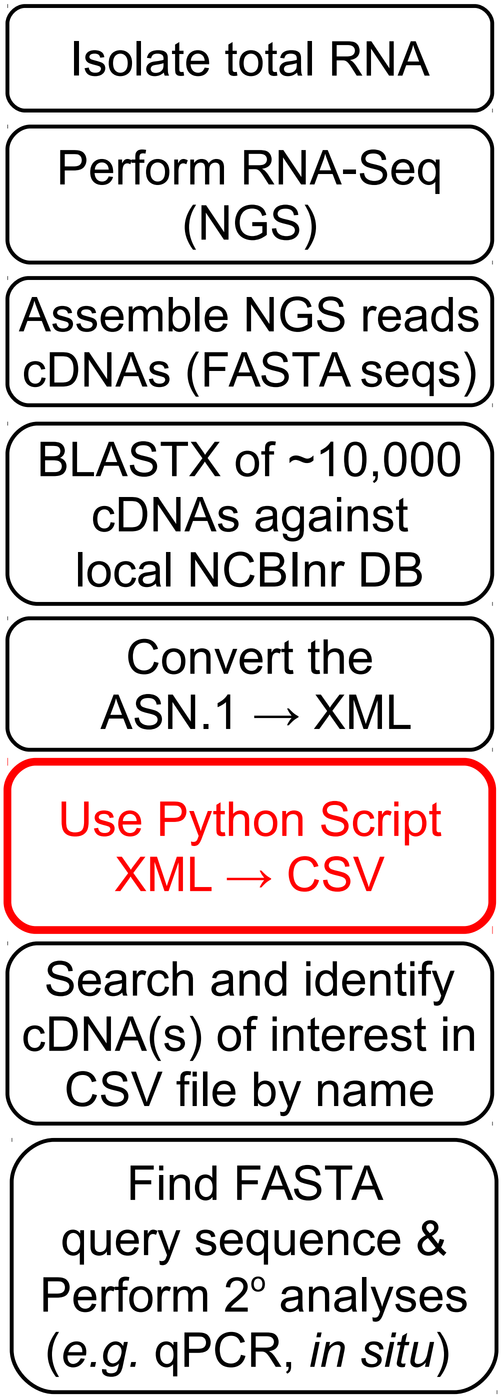
The authors wish to thank Miami University's RedHawk High Performance Computing Cluster, specifically Drs. David Woods and Jens Mueller for their advice and help on the batch ncbi-tools scripts and NCBInr database set-up.

References:

1. Field, D.; Tiwari, B.; Booth, T.; Houten, S.; Swan, D.; Bertrand, N.; Thurston, M. Open software for biologists: from famine to feast. *Nat Biotech* **2006**, *24*, 801–803.
2. Benson, D. A.; Karsch-Mizrachi, I.; Lipman, D. J.; Ostell, J.; Wheeler, D. L. GenBank. *Nucleic Acids Research* **2008**, *36*, D25–D30.
3. Frese, K.; Katus, H.; Meder, B. Next-Generation Sequencing: From Understanding Biology to Personalized Medicine. *Biology* **2013**, *2*, 378–398.
4. Ostell, J. M.; Wheelan, S. J.; Kans, J. A. The NCBI Data Model. In *Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins*; Baxevanis, A. D.; Ouellette, B. F. F., Eds.; John Wiley & Sons, Inc., 2001; pp. 19–43.

Figure Legends:

**Figure 1**: Diagrammatic scheme of identification of expressed cDNAs for a non-model organism with an available annotated genome. The procedure begins at the top and progresses to the bottom. Our script (**Step 6**) facilitates the examination and searching of a large list of putative identified cDNAs as a comma separated values (CSV) file in a familiar program such as Microsoft Excel or LibreOffice Calc.


Isolate total RNA
Perform RNA-Seq (NGS)
Assemble NGS reads cDNAs (FASTA seqs)
BLASTX of ~10,000 cDNAs against local NCBInr DB
Convert the ASN.1 → XML
Use Python Script XML → CSV
Search and identify cDNA(s) of interest in CSV file by name
Find FASTA query sequence & Perform 2° analyses (*e.g.* qPCR, *in situ*)