Cunningham: a BLAST Runtime Estimator

James Robert White†, Malcolm Matalka, W. Florian Fricke & Samuel V. Angiuoli
Institute for Genome Sciences, University of Maryland - School of Medicine, Baltimore, MD 21201
†Contact email: jwhite@som.umaryland.edu

ABSTRACT
BLAST is arguably the single most important piece of software ever written for the biological sciences. It is the core of most bioinformatics workflows, being a critical component of genome homology searches and annotation. It has influenced the landscape of biology by aiding in everything from functional characterization of genes to pathogen detection to the development of novel vaccines. While BLAST is very popular, it is also often one of the most computationally intensive parts of bioinformatics analysis. In our workflows, BLAST typically takes the majority of cpu time, and we need to parallelize to finish in a reasonable time frame. Waiting for BLAST to finish without having any clue of how long it's going to take is kind of depressing, and you could waste a day of work trying to run a job that would never finish. If you feel the same way we do, then check out **Cunningham**, a tool we designed to estimate BLAST runtimes for shotgun sequence datasets using sequence composition statistics. We've trained its models on real metagenomic sequence data using the Amazon EC2 cloud, and it will provide a relatively quick estimate for datasets with up to tens of millions of sequences. It's not perfect, but it'll give you at least some idea of expected runtime, how large a cluster you're going to need, how much you'll need to partition your data, etc. We use it all the time now, so we hope it'll be useful to someone else out there. **Cunningham** has been implemented in CloVR for efficient autoscaling in the cloud and is freely available at http://clovr.org.

INTRODUCTION

We run lots of BLAST searches. Typically, we're searching DNA or protein sequences from a single-genome or metagenomic dataset against a big database. Because of increased throughput from next-generation sequencing technologies, you'll often end up with millions of sequences, so it's no longer possible to BLAST all your sequences against a decent size database on a single machine or even a small cluster. Now, we are leveraging cloud computing to run large-scale parallelized BLAST jobs and then compiling all the results for biologists and performing downstream analysis.

But these days when we throw a bunch of sequences at the cloud (e.g. Amazon EC2), we're still not sure how long it's going to take. How many machines do we need? How many pieces do we need to chunk our sequences into? How much is this going to cost us? The answers to these questions are usually found empirically by running a small set of sequences against the target database and scaling up that estimate based on your dataset size. This approach has several pitfalls, in particular, the composition of your query set has a significant impact on the BLAST runtime. Figure 1 illustrates this situation. Here we took a real dataset of 1000 proteins, and created several new datasets with different amino acid composition but the same sequence lengths. Then we ran BLASTP searches of these datasets against the NCBI COG database and measured each runtime. The longest run ended up being 78% longer than the shortest runtime.

From this we can see that sequence composition matters, and BLAST is so sophisticated (and heuristic) that it is difficult to accurately estimate BLAST runtime for all sequences. After playing around with this a bit, we have found that there is reasonable consistency in sequence composition for most shotgun metagenomic sequence datasets. Therefore, it's possible to use some gross measures to estimate how long a practical BLAST

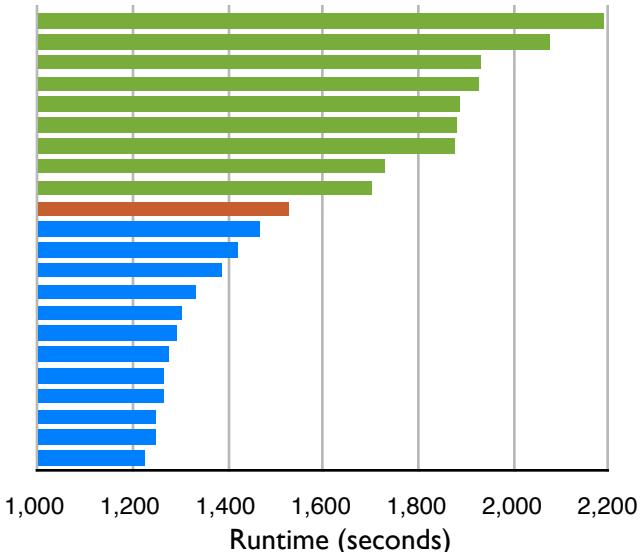

Figure 1. Variable BLAST runtimes due to changes in sequence composition. In this mock example, each bar represents the BLASTP runtime of a query dataset with 1000 protein sequences (of identical lengths) against the NCBI COG database. The original dataset (orange) had a runtime of ~1,532 seconds. By randomly altering the amino acid composition of the dataset, we are able to increase the runtime by up to 43% or decrease the runtime by up to 20%. These dramatic differences indicate that the runtime of a BLAST program is highly sensitive to the composition of the sequence being analyzed.

Database	Sequence type	Programs used
NCBI RefSeq	DNA (genomes)	BLASTN
NCBI COG	Protein	BLASTP, BLASTX
eggNOG	Protein	BLASTP, BLASTX
KEGG genes	Protein	BLASTP, BLASTX

Table1. BLAST databases used for this project.

search would take. Cunningham is a tool we've developed just for this purpose. Given a set of sequences, a blast program (BLAST{N,P,X}), and a pre-specified database (db), Cunningham will rapidly assess the sequence profile of the queries and produce an estimate of BLAST runtime.

What does Cunningham use to estimate runtime? One thing that appears to be crucial is the number of matching words (or *seeds*) between the query sequences and the database itself. It's actually more complicated with BLASTP and BLASTX, which use so-called "neighborhoods" of non-identical words to seed alignments and requirements of two-hits within a region of some length [1]. Nevertheless, the number of identically matching seed words is still informative of overall similarity between query and db. Another factor is sequence length. If all sequences are less than the word size, there won't be any seed matches to explore locally. If the query sequences are very long, there will likely be many more local alignments to perform and this will take additional computational time, so we'll also consider average sequence length within a dataset.

VALIDATION

To demonstrate how these indicators can assist in runtime estimation, we collected several databases (Table 1) and created a series of sequence datasets by randomly sampling from a collection metagenomics projects (Table 2). DNA sequences were generated using Sanger or 454 technology, so there is substantial range in sequence lengths (<100bp to >800bp) for these random datasets. For protein-based searches, we randomly sampled from the over 3 million predicted proteins and protein fragments from the MetaHIT project. I also computed *k*mer profiles for each database (*k*=3 for protein, *k*=11 for dna (including reverse complemented sequence)), so we can determine the number of matching word pairs between a query set and a database. Mathematically, we compute this as:

$$M = \sum_{i=1}^{|W|} Q(w_i) \cdot D(w_i)$$

where *W* is the set of all words, and the number of occurrences of the word *wi* in the query set and database are *Q(wi)* and *D(wi)*, respectively.

Validation BLAST jobs were performed using CloVR (http://clovr.org) on the Amazon EC2 cloud using c1.xlarge on-demand instances, and every job got its own cpu to work on to avoid overloading bias. Default parameters were used with the exception of '-b 1 -e 1e-5 -F F'. Figure 2 shows the resulting runtimes plotted with the corresponding number of matching word pairs. We see strong linear relationships between runtime and *M,* which we can take advantage of using standard regression techniques. **Cunningham** uses a different linear model for BLASTN, BLASTP, and BLASTX, but they all share a common form:

Dataset	Sequence type	Reference
EBPR-Sludge	Sanger shotgun DNA	[2]
Mediterranean gutless worm	Sanger shotgun DNA	[3]
Global ocean sampling (2 samples)	Sanger shotgun DNA	[4]
Nine biomes (microbial samples)	454 shotgun DNA	[5]
Peru margin deep sea sediment	454 shotgun DNA	[6]
Developing infant gut	454 shotgun DNA	[7]
Obese and lean twins	454 shotgun DNA	[8]
MetaHIT predicted proteins	proteins from assembled Illumina	[9]

Table 2. Metagenomic datasets used for this project. DNA and protein sequence datasets were created by randomly selecting sequence sets from compiled projects.

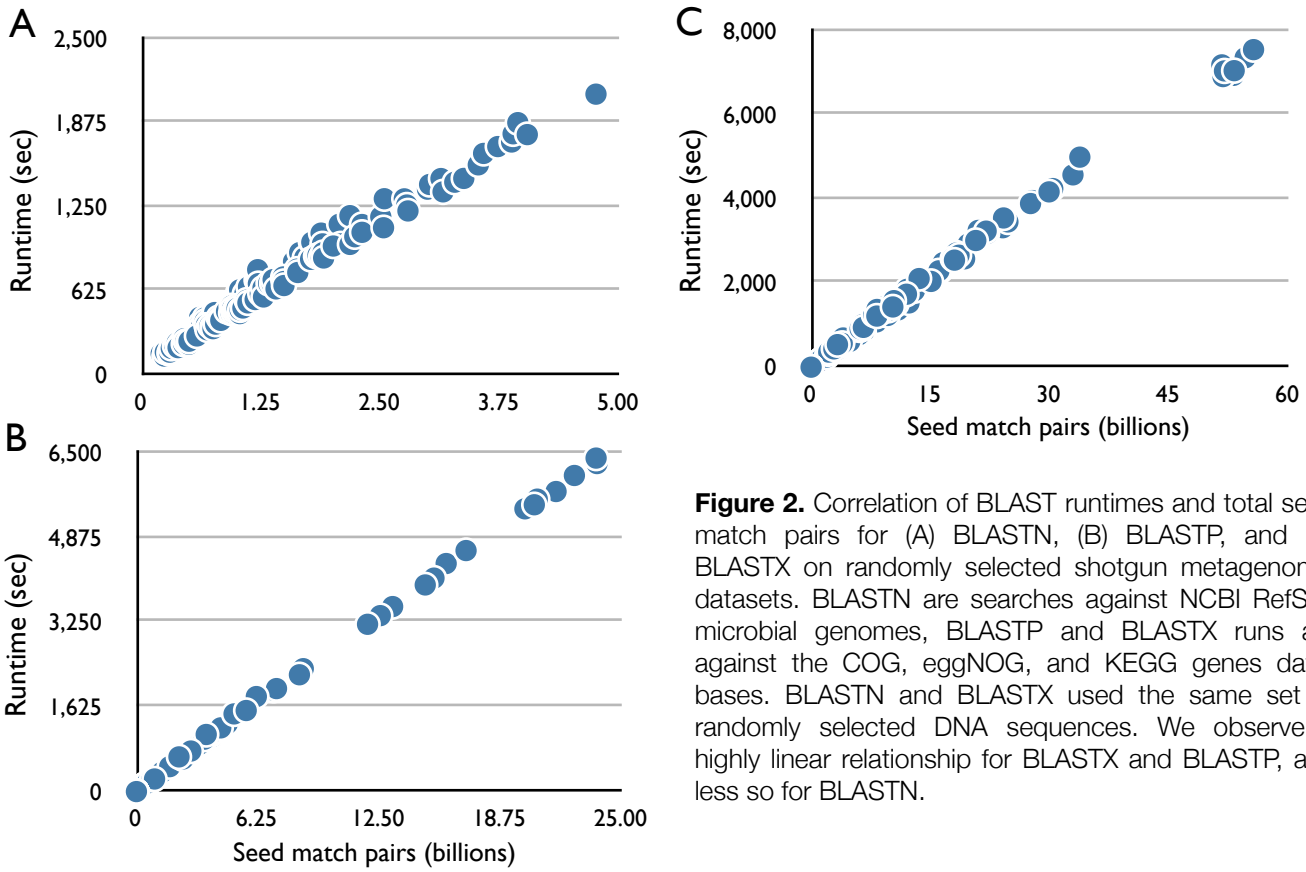

Figure 2. Correlation of BLAST runtimes and total seed match pairs for (A) BLASTN, (B) BLASTP, and (C) BLASTX on randomly selected shotgun metagenomic datasets. BLASTN are searches against NCBI RefSeq microbial genomes, BLASTP and BLASTX runs are against the COG, eggNOG, and KEGG genes databases. BLASTN and BLASTX used the same set of randomly selected DNA sequences. We observe a highly linear relationship for BLASTX and BLASTP, and less so for BLASTN.

$$T \sim \beta_1 M + \beta_2 L$$

where T (the runtime) depends on the number of shared seed pairs M, and the average query sequence length L. Fitting this model to the data from figure 2, we see acceptable prediction accuracy for BLASTN, BLASTP, and BLASTX, with r-squared values of 0.996, 0.999, and 0.998, respectively. In all three models, M and L were both found to be significant (P values < 0.01). Though our sample data is somewhat biased and lim-

ited in runtimes to a few hours, we are encouraged by the excellent prediction accuracy across all three programs (figure 3). So, let's have fun making some predictions of runtime for really large BLAST jobs.

BIG RUNS & SUMMARY

Cunningham is implemented with a set of specific databases, but the indicators could be computed for technically any possible database. Here we'll take some well known datasets/databases and see what Cunningham predicts. Let's also consider how long it takes Cunningham to run on a standard desktop because you can't

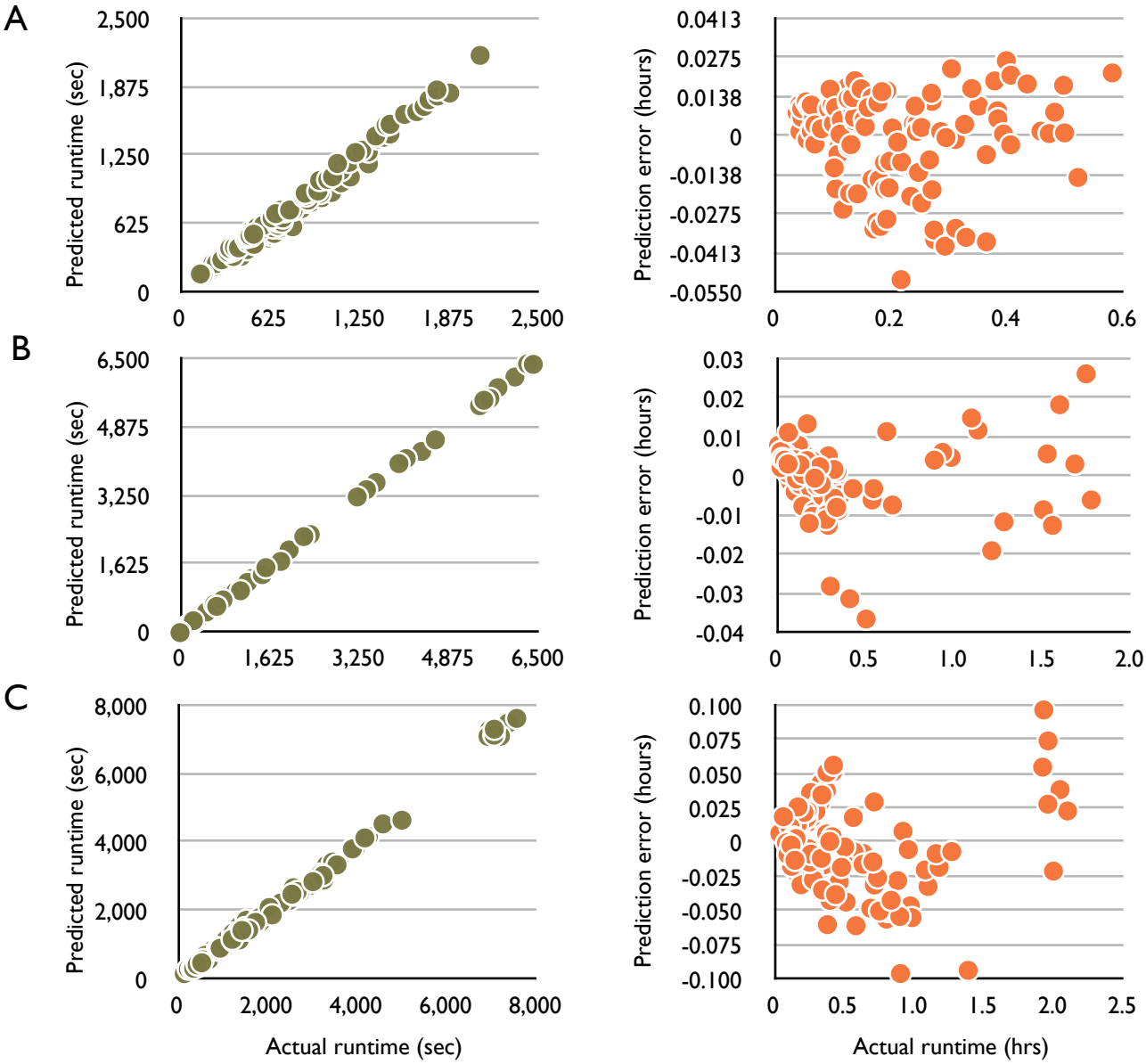

Figure 3. Linear model prediction accuracy for (A) BLASTN, (B) BLASTP, and (C) BLASTX on randomly selected shotgun metagenomic datasets. These linear models utilize the total number of seed matches between a dataset and a database as well as the average query length. R-squared model fit values are: BLASTN -> 0.996, BLASTP-> 0.999, & BLASTX->0.998. Prediction for runtimes were often within 0.05 cpu hours (3 minutes). Cunningham uses the same linear model with different coefficient values for each program.

wait all day right? Table 3 shows the results of Cunningham for different datasets and a few specs about each run. Note when the datasets become too large, counting all the shared seed pairs becomes cumbersome. So as a heuristic, if there are more than 600,000 sequences or over 300 million residues in the query dataset, then Cunningham will randomly subsample from the data and compute seed match pairs for multiple rounds, and finally scale up the estimate based on the average number of seed match pairs and the total number of sequences in the query. If you've gotten this far, it either means you want to try Cunningham or you want to write something that beats it. Well done either way!

Cunningham is open source and currently implemented in the CloVR package for efficient autoscaling in the cloud. Please visit *http://clovr.org* for the free source code.

Acknowledgements

This work was supported with Federal funds from the National Human Genome Research Institute (NHGRI), National Institutes of Health (NIH), under Contract 5RC2HG005597-02 and from the National Science Foundation (NSF) under Contract 0949201.

Dataset	Total residues	BLAST program	Database	Cunningham estimate	Cunningham's runtime
Obese and lean twins	1.83 Gb	BLASTN	RefSeq	647 cpu hours (27 cpu days)	4 min 25 sec
Obese and lean twins	1.83 Gb	BLASTX	COG	1030 cpu hours (43 cpu days)	4 min 21 sec
Nine biomes (microbial samples)	698 Mb	BLASTN	RefSeq	236 cpu hours (10 cpu days)	3 min 14 sec
Nine biomes (microbial samples)	698 Mb	BLASTX	KEGG	6233 cpu hours (260 cpu days)	3 min 8 sec
Developing infant gut	145 Mb	BLASTN	RefSeq	48 cpu hours (2 cpu days)	2 min 4 sec
Developing infant gut	145 Mb	BLASTX	NCBI-NR	6918 cpu hours (288 cpu days)	1 min 40 sec

Table 3. Cunningham predictions for some big datasets. We have not assessed their accuracy.

REFERENCES

[1] Altschul SF, Madden TL, Schaffer AA, Zhang J, Zheng Z, Miller W, Lipman DJ. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Research. 1997 July; 25(17):3389-3402.

[2] Garcia Martin H, Ivanova N, Kunin V, Warnecke F, Barry KW, McHardy AC, et al. Metagenomic analysis of two enhanced biological phosphorus removal (EBPR) sludge communities. Nature biotechnology. 2006 Oct;24(10):1263-9.

[3] Woyke T, Teeling H, Ivanova NN, Huntemann M, Richter M, Gloeckner FO, et al. Symbiosis insights through metagenomic analysis of a microbial consortium. Nature. 2006 Oct 26;443(7114):950-5.

[4] Rusch DB, Halpern AL, Sutton G, Heidelberg KB, Williamson S, Yooseph S, et al. The Sorcerer II Global Ocean Sampling expedition: northwest Atlantic through eastern tropical Pacific. PLoS Biol. 2007 Mar;5(3):e77.

[5] Dinsdale EA, Edwards RA, Hall D, Angly F, Breitbart M, Brulc JM, et al. Functional metagenomic profiling of nine biomes. Nature. 2008 Apr 3;452(7187):629-32.

[6] Biddle JF, Fitz-Gibbon S, Schuster SC, Brenchley JE, House CH. Metagenomic signatures of the Peru Margin subseafloor biosphere show a genetically distinct environment. Proc Natl Acad Sci U S A. 2008 Jul 29;105(30):10583-8.

[7] Koenig JE, Spor A, Scalfone N, Fricker AD, Stombaugh J, Knight R, et al. Microbes and Health Sackler Colloquium: Succession of microbial consortia in the developing infant gut microbiome. Proc Natl Acad Sci U S A. 2010 Jul 28.

[8] Turnbaugh PJ, Hamady M, Yatsunenko T, Cantarel BL, Duncan A, Ley RE, et al. A core gut microbiome in obese and lean twins. Nature. 2009 Jan 22;457(7228):480-4.

[9] Qin J, Li R, Raes J, Arumugam M, Burgdorf KS, Manichanh C, et al. A human gut microbial gene catalogue established by metagenomic sequencing. Nature. 2010 Mar 4;464(7285):59-65.