pfastGO: A fast way to get GO-term annotations for bacterial protein sequences

Daniel Wüthrich[1] and Rémy Bruggmann[1*]

[1]Interfaculty Bioinformatics Unit and Swiss Institute of Bioinformatics, University of Bern, Bern, Switzerland

Abstract

Summary: Projects with large numbers of *de novo* sequenced bacterial genomes are now very common in microbial research. Recently released tools make it already very straightforward to annotate and analyze large numbers of newly sequenced and assembles bacterial strains. However, the functional annotation, and especially the assignment of GO terms, is rather poorly covered with analysis tool. This is especially useful for studies with large numbers of bacterial genomes, as manual analysis is not efficient enough to study hundreds of genomes. Here we are introducing the functional annotation pipeline pfastGO. It assigns GO terms to amino acid sequences using homology search and the detection of conserved domains within a time frame that allows the annotation of hundreds of genomes.

Availability and implementation: The functional annotation pipeline pfastGO is implemented in Bash and Python. The code is freely available on github: https://github.com/danielwuethrich87/pfastGO

INTRODUCTION

In the last years many studies with large numbers of bacterial genomes were published (Sun *et al.*, 2015; Hilty *et al.*, 2014). Recently released tools like Prokka (Seemann, 2014) and Roary (Page *et al.*, 2015) made the analysis of big datasets more efficient and accessible. Even though, valuable insights into evolutionary processes and the conservation of genes were provided by these studies, the functionality of the identified genes was limited to specific genomic regions or specific gene types. Gene ontology terms (GO terms) (Gene and Consortium, 2008) are a practical tool to study the functionally of larger gene sets. Using these GO terms overrepresented functionalities in a subset of genes can be detected using GO enrichment analysis using tools such as topGO (Alexa and Rahnenfuhrer, 2016). Furthermore, also functionalities can be mapped in pathways databases like KEGG (Ogata *et al.*, 1999), MetaCyc (Caspi *et al.*, 2012) and Reactome (Croft *et al.*, 2011) on whole genome levels.

Several tools are already available that have the ability to assign GO terms to amino acid sequences. Blast2GO (Conesa *et al.*, 2005) is the most popular one. Even though it has a sophisticated algorithm to assign GO terms, it is computationally expensive and is primarily used for the annotation for eukaryotic genomes. To same is true for the also popular Trinotate (Grabherr *et al.*, 2011), as it was developed for transcriptome functional annotation and analysis of eukaryotes. Taken together the tools are not optimized for large bacterial datasets.

We developed pfastGO, a command line functional annotation pipeline, that is able to assign GO terms automatically to the genes of large bacterial genomic datasets. The pfastGO pipeline is also easy to use and install, as we also included the databases and the needed tools into the installation script. With this pipeline, we are able to annotated 2,000 CDSs of a bacterial genome, using 4 threads in less than 20 minutes.

DESCRIPTION

1 Input

The input for the pipeline is a mutifasta file, containing the amino acid sequences of the predicted CDSs of a bacterial genome. For our testing, we used the output of the Prokka annotation pipeline. *Ab initio* annotation tools like GeneMark (Borodovsky and Mcininch, 1993), Glimmer (Delcher *et al.*, 1999) and Prodigal (Hyatt *et al.*, 2010) are also appropriate to produce the input.

2 Functional annotation

The pfastGO pipeline is build out of three core elements. First, the amino acid sequences of the studied genes are aligned against the bacterial entries of the manually curated protein database of Swiss-Prot (Boeckmann *et al.*, 2003) using BLASTP (Altschul SF, Gish W, Miller W, Myers EW, 1990). The resulting blast hits are then grouped using the machine learning algorithm DBSCAN (Ester *et al.*, 1996) based on the alignment length and the alignment identity (Figure 1). Afterwards, pfastGO selects the group that contains the best blast hit (e<10^6), to exclude alignments that show significant lower homology than the top blast hit. The GO terms of the best Blast hits (up to 20) of this group will then be assigned to the amino acid sequence.

In the second step the amino acid sequence is compared to the Pfam protein family database (Finn *et al.*, 2014) using PfamScan that is based on HMMER (Eddy, 2011). The GO terms of all found protein families (e<10^6) is then assigned to the amino acid sequence.

In the final step the results of the BLASTP and PfamScan analyses are fused and redundant GO terms removed. Subsequent the found GO terms are mapped to Enzyme Commission numbers (EC), to reaction numbers from the MetaCyc database and to KEGG reaction entries, what simplifies downstream analyses. Additionally, a product name is assigned to the amino acid sequence. Therefore, the name of the selected blast hits is assigned as long it is not a hypothetical or a protein with unknown function. If this is the case for any of the selected blast-hits, the family of the protein is created based on the PfamScan results.

3 Output

The output consists out of three tab separated tables. Two include the raw results of the BLASTP and the PfamScan analyses what allows a manual inspection if needed. The third table is a list of all submitted genes of the input multi fasta file. It provides the original name from the input, a newly assigned name, and also the assigned GO terms, ECs, MetaCyc reaction ids and KEGG entries.

RESULTS

We developed pfastGO in order to annotate datasets of a few hundred bacterial genomes in parallel within a useful time, with a fully automatic and easy to used command line tool. The pipeline is optimized for 64 bit Linux systems and also support parallelization using GNU parallel. We already annotated hundreds of genome assemblies within reasonable time. For example, it took 16 minutes 58 seconds, to functionally annotate the genes of the *Lactbacillus helveticus* strain CNRZ 32 (genome size 2,225,962 bp) (Broadbent *et al.*, 2013), using four threads. Finally, we also found the integration of the machine learning algorithm DBSCAN as very useful, as it reduces many false positives homologues. As a showcase example, we selected the automatically annotated Aldehyde-alcohol dehydrogenase (A4ZH65_LACHE) from CNRZ 32. The blast search against the Swiss-Prot database resulted in 73 blast hits (e<10^6). By taking a look at the query coverage and the alignment identity, we found that three hits show much higher values in both features (Figure 1). However, the DBSCAN clustering algorithm is able to separate the blast hits into two groups and pfastGO only takes the three blast hits from cluster 1 for our annotation. By taking a look at the top 20 blast hits (table 1) we see that the one from cluster 1 are all annotated to be an Aldehyde-alcohol dehydrogenase, whereas the other 17 blast hits are annotated as various dehydrogenases or other proteins. With this we see that DBCSCAN allows us to exclude blast hits that might lead to false annotations.

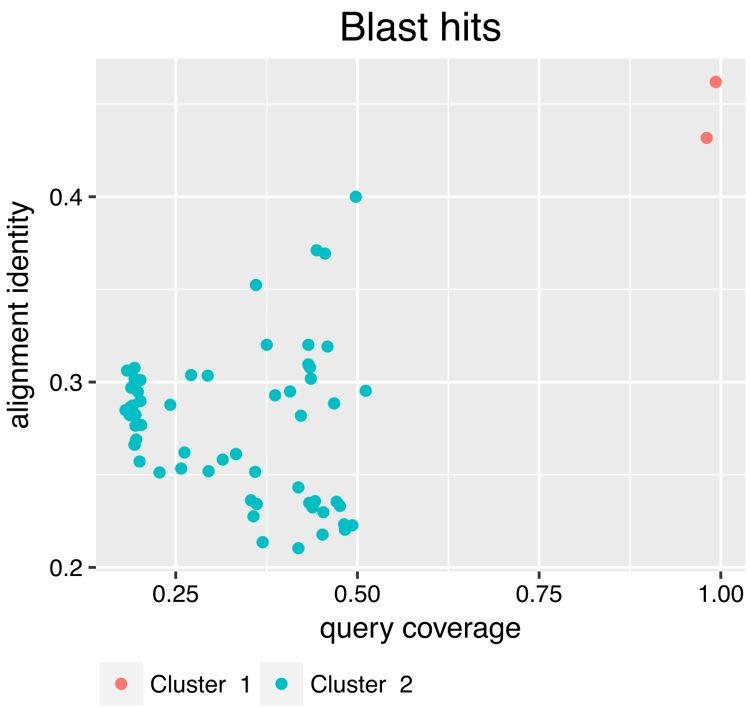

Figure 1: Blast hit clustering using DBSCAN. The axes of the scatter plot indicate the alignment identity and query coverage of the blast hit from A4ZH65_LACHE found in the Swiss-Prot database. The colors represent the cluster to which the blast hits were assigned using DBSCAN. Cluster 1 contains the blast hits with the lowest e-values.

Table 1: Best 20 Blast hits of A4ZH65_LACHE from the Swiss-Prot database

Cluster	query coverage	alignment identity	function	UniProtID
1	0.99	0.46	Aldehyde-alcohol dehydrogenase	ADHE_ECOLI
1	0.99	0.46	Aldehyde-alcohol dehydrogenase	ADHE_ECO57
1	0.98	0.43	Aldehyde-alcohol dehydrogenase	ADHE_CLOAB
2	0.50	0.40	Succinate-semialdehyde dehydrogenase (acetylating)	SUCD_CLOK5
2	0.44	0.37	NADPH-dependent butanol dehydrogenase	ADH1_CLOSA
2	0.46	0.37	Sulfoacetaldehyde dehydrogenase (acylating)	SAUS_CUPNH
2	0.46	0.32	Ethanolamine utilization protein EutE	EUTE_ECOLI
2	0.51	0.30	Ethanolamine utilization protein EutE	EUTE_SALTY
2	0.43	0.32	Probable alcohol dehydrogenase	ADH2_ECOLI
2	0.43	0.31	Alcohol dehydrogenase 2	ADH2_ZYMMA
2	0.44	0.30	NAD-dependent methanol dehydrogenase	MEDH_BACMT

2	0.36	0.35	Ethanolamine utilization protein EutG	EUTG_ECOLI
2	0.43	0.31	Alcohol dehydrogenase 2	ADH2_ZYMMO
2	0.38	0.32	Ethanolamine utilization protein EutG	EUTG_SALTY
2	0.43	0.31	Lactaldehyde reductase	FUCO_ECOLI
2	0.43	0.31	Lactaldehyde reductase	FUCO_ECO57
2	0.47	0.29	1,3-propanediol dehydrogenase	DHAT_KLEPN
2	0.47	0.29	1,3-propanediol dehydrogenase	DHAT_CITFR
2	0.41	0.29	Long-chain-alcohol dehydrogenase 1	ADH1_GEOTN
2	0.42	0.28	Alcohol dehydrogenase	GBSB_BACSU

Alexa,A. and Rahnenfuhrer,J. (2016) Gene set enrichment analysis with topGO.

Altschul SF, Gish W, Miller W, Myers EW,L.D. (1990) Basic local alignment search tool. *J Mol Biol.*, **Oct 5**, 403–10.

Boeckmann,B. *et al.* (2003) The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. *Nucleic Acids Res.*, **31**, 365–370.

Borodovsky,M. and Mcininch,J. (1993) GeneMark: parallel gene recognition for both DNA strands. *Comput. Chem.*, **17**, 123–133.

Broadbent,J.R. *et al.* (2013) Complete Genome Sequence for Lactobacillus helveticus CNRZ 32, an Industrial Cheese Starter and Cheese Flavor Adjunct. *Genome Announc.*, **1**, e00590–13–e00590–13.

Caspi,R. *et al.* (2012) The MetaCyc database of metabolic pathways and enzymes and the BioCyc collection of pathway / genome databases. *Nucleic Acids Res.*, **40**, 742–753.

Conesa,A. *et al.* (2005) Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research. *Bioinformatics*, **21**, 3674–6.

Croft,D. *et al.* (2011) Reactome: A database of reactions, pathways and biological processes. *Nucleic Acids Res.*, **39**.

Delcher,A.L. *et al.* (1999) Improved microbial gene identification with GLIMMER. *Nucleic Acids Res.*, **27**, 4636–4641.

Eddy,S.R. (2011) Accelerated profile HMM searches. *PLoS Comput. Biol.*, **7**.

Ester,M. *et al.* (1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*., pp. 226–231.

Finn,R.D. *et al.* (2014) Pfam: The protein families database. *Nucleic Acids Res.*, **42**.

Gene,T. and Consortium,O. (2008) The Gene ontology project in 2008. *Nucleic Acids Res.*, **36**, 440–444.

Grabherr,M.G. *et al.* (2011) Full-length transcriptome assembly from RNA-Seq data without a reference genome. *Nat. Biotechnol.*, **29**, 644–652.

Hilty,M. *et al.* (2014) Global Phylogenomic Analysis of Nonencapsulated Streptococcus pneumoniae Reveals a Deep-Branching Classic Lineage That Is Distinct from Multiple Sporadic Lineages. *Genome Biol. Evol.*, **6**, 3281–94.

Hyatt,D. *et al.* (2010) Prodigal: prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics*, **11**, 119.

Ogata,H. *et al.* (1999) KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.*, **27**, 29–34.

Page,A.J. *et al.* (2015) Roary: Rapid large-scale prokaryote pan genome analysis. *Bioinformatics*, **31**, 3691–3693.

Seemann,T. (2014) Prokka: rapid prokaryotic genome annotation. *Bioinformatics*, **30**, 2068–9.

Sun,Z. *et al.* (2015) Expanding the biotechnology potential of lactobacilli through comparative genomics of 213 strains and associated genera. *Nat. Commun.*, **6**, 8322.