Institute of Biotechnology and
Department of Biosciences, Division of Genetics
Faculty of Biological and Environmental Sciences
University of Helsinki

and

Integrative Life Science Doctoral Program (ILS)

Functional Sequence Annotation in an Error-prone Environment

Patrik Koskinen

ACADEMIC DISSERTATION

To be presented, with the permission of the Faculty of Biological and Environmental Sciences of the University of Helsinki, for public examination in Auditorium 2, Viikki Infocenter Korona, on 22nd of August 2014, at 12 o'clock.

Helsinki 2014

Supervised by

Professor Liisa Holm
Department of Biosciences & Institute of Biotechnology
University of Helsinki
Finland

Reviewed by

Professor Tapio Salakoski
Department of Information Technology
University of Turku
Finland

and

Professor Erik Sonnhammer
Department of Biochemistry and Biophysics
Stockholm University
Sweden

Opponent

Dr. Reinhard Schneider
Centre for Systems Biomedicine
University of Luxembourg
Luxembourg

Custos

Professor Jukka Finne
Department of Biosciences
University of Helsinki
Finland

ISBN 978-951-51-0008-5 (pbk.)
ISBN 978-951-51-0009-2 (PDF)

http://ethesis.helsinki.fi
Hansaprint Oy, Helsinki 2014

To my dearest,
Sanna

"*If I have seen further it is by standing on the shoulders of giants*"
-Isaac Newton

Table of Contents

List of original publications

Abstract

Abbreviations

1 Introduction 1
- 1.1 Information gap 1
- 1.2 How to define function 2
- 1.3 Sequence similarity implies function similarity 4
  - 1.3.1 Orthologue and paralogue 5
  - 1.3.2 Methods for sequence similarity 6
- 1.4 Review of function annotation methods 8
  - 1.4.1 Description prediction methods 9
    - 1.4.1.1 Nearest neighbour 9
    - 1.4.1.2 K-nearest neighbours 10
  - 1.4.2 Gene ontology prediction methods 10
  - 1.4.3 Family, feature and profile prediction methods 13
- 1.5 Errors in databases 15

2 Aims of the present study 16

3 Materials & methods 17
- 3.1 SANS 17
- 3.2 PANNZER 19
- 3.3 CAFA 2011 Challenge 20

4 Results and discussion 22
- 4.1 SANS results 22
- 4.2 PANNZER results 23
  - 4.2.1 Description prediction 23
    - 4.2.1.1 Prokaryotic dataset 24
    - 4.2.1.2 Eukaryotic dataset 24
  - 4.2.2 Gene ontology prediction and CAFA 2011 challenge 25
- 4.3 Practical applications in genome analysis 27
  - 4.3.1 OrthoSANS 27
  - 4.3.2 SANSMapper 29

5 Conclusions and perspectives 30

Acknowledgements 32

References 33

List of original publications

This thesis is based on the following articles, which are referred to in the text by their Roman numerals.

1. **Koskinen JP**, Holm L. **SANS: high-throughput retrieval of protein sequences allowing 50% mismatches.** *Bioinformatics.* 2012 Sep 15;28(18):i438-i443.

2. **Koskinen JP**, Törönen P, Nokso-Koivisto J, Holm L. **PANNZER – Functional annotation of uncharacterized proteins in an error-prone environment**. *Manuscript.*

3. Radivojac P, Clark WT, Oron TR, Schnoes AM, Wittkop T, Sokolov A, Graim K, Funk C, Verspoor K, Ben-Hur A, Pandey G, Yunes JM, Talwalkar AS, Repo S, Souza ML, Piovesan D, Casadio R, Wang Z, Cheng J, Fang H, Gough J, **Koskinen JP**, Törönen P, Nokso-Koivisto J, Holm L, Cozzetto D, Buchan DW, Bryson K, Jones DT, Limaye B, Inamdar H, Datta A, Manjari SK, Joshi R, Chitale M, Kihara D, Lisewski AM, Erdin S, Venner E, Lichtarge O, Rentzsch R, Yang H, Romero AE, Bhat P, Paccanaro A, Hamp T, Kaßner R, Seemayer S, Vicedo E, Schaefer C, Achten D, Auer F, Boehm A, Braun T, Hecht M, Heron M, Hönigschmid P, Hopf TA, Kaufmann S, Kiening M,Krompass D, Landerer C, Mahlich Y, Roos M, Björne J, Salakoski T, Wong A, Shatkay H, Gatzmann F, Sommer I, Wass MN, Sternberg MJ, Škunca N, Supek F, Bošnjak M,Panov P, Džeroski S, Šmuc T, Kourmpetis YA, van Dijk AD, ter Braak CJ, Zhou Y,Gong Q, Dong X, Tian W, Falda M, Fontana P, Lavezzo E, Di Camillo B, Toppo S, LanL, Djuric N, Guo Y, Vucetic S, Bairoch A, Linial M, Babbitt PC, Brenner SE,Orengo C, Rost B, Mooney SD, Friedberg I. **A large-scale evaluation of computational protein function prediction.** *Nature Methods.* 2013 Mar;10(3):221-7.

Additional unpublished data will be presented

Abstract

As more and more sequences are submitted to public databases, so will grow more computationally challenging sequence retrieval systems. When for example the UniProtKB/TrEMBL doubles in size annually, the tools used today might not be sufficient tomorrow. Faster and computationally lighter methods are needed for sequence retrieval. This study presents a computationally more efficient tool. The Suffix Array Neighbourhood Search (SANS) tool is a hundred fold faster than the most commonly used tool BLAST.

The sequence databases do not only grow in size but also in the number of different functional annotations they contain. Recent studies have shown that a large number of these annotations are assigned incorrectly. When the error level of functional annotations in the databases grows to a statistically significant figure, better methods and the use of error detection statistics are highly recommended. In the present study we introduce novel methods for weighted statistical testing of functional annotations. Also novel methods for the calculation of information content value are presented. The information content value enables the discrimination of informative from uninformative annotations.

A growing number of functional annotation tools are introduced annually. Since no gold standard evaluation sets exist, it is impossible to determine the reliability of the different methods. The Critical Assessment of Functional Annotations (CAFA) challenge is the first attempt to evaluate functional annotation tools by using 'blind testing' on a large scale. The first CAFA challenge included the evaluation of 54 state-of-the-art methods in two different Gene Ontology categories. The results show that there is a plenty of room for improvement in the prediction accuracy of the existing tools.

Abbreviations

BBH	Best BLAST Hit
BIBH	Best Informative BLAST Hit
BLASR	Basic Local Alignment with Successive Refinement
BLAST	Basic Local Alignment Search Tool
BPO	Biological Process Ontology
BWA	Burrows-Wheeler Aligner
CAFA	Critical Assesment of Functional Annotations
CAMERA	Cyberinfrastructure for Advanced Microbial Ecology Research and Analysis
CCO	Cellular Component Ontology
CLR	Continous Long Read
COGIC	Combined Graph-Information Content
DAG	Directed Acyclic Graph
DB	Database
DE	DEscription line
DSM	Description Similarity Measure
e.g.	exempli gratia
EC	Enzyme Commission
etc.	et cetera
FunCat	Functional Catalogue
GO	Gene Ontology
GOA	Gene Ontology Annotation
GSZ	Gene Set Z-score
HAMAP	High-quality Automated and Manual Annotation of Proteins
HSP	High-scoring Segment Pair
i.e.	id est
IEA	Inferred from Electronic Annotation
ISA	Inverted Suffix Array
KNN	K-Nearest Neighbour
MFO	Molecular Function Ontology
MIPS	Munich Information Center for Protein Sequences
NR	Not Recorded / Non-Redundant
PacBio	Pacific Biosciences
PAMGO	Plant Associated Microbe Gene Ontology
PANNZER	Protein ANNotation with Z-scoRE
ProDom	Protein Domain
PSI-BLAST	Position Specific Iterative Basic Local Alignment Search Tool
PSSM	Position Specific Scoring Matrix
RAST	Rapid Annotation using Subsystem Technology
RBH	Reciprocal Best hit
SA	Suffix Array
SANS	Suffix Array Neighbourhood Search
SAP	Suffix Array vs. Proteins
SMART	Simple Modular Architecture Research Tool
SSRL	Sequence Similarity Result List
TF-IDF	Term Frequency – Inverse Document Frequency
WS	Word Score
WWS	Weighted Word Score

1 Introduction

1.1 Information gap

The last decade has seen a vast increase in the production of sequencing data. This has resulted in a remarkable growth of both manually curated and automatically annotated protein databases. Some of the sequence databases have doubled in size every year. Figure 1 shows the exponential increase of protein sequences submitted in the most comprehensive automatically annotated database called UniProtKB/TrEMBL, which today holds (Release 2013_10) 44,746,523 sequence entries. In comparison the manually curated database UniProtKB/Swiss-Prot holds 541,561 sequence entries which represent only a fraction (1.2%) of the total amount of protein sequences in UniProtKB.

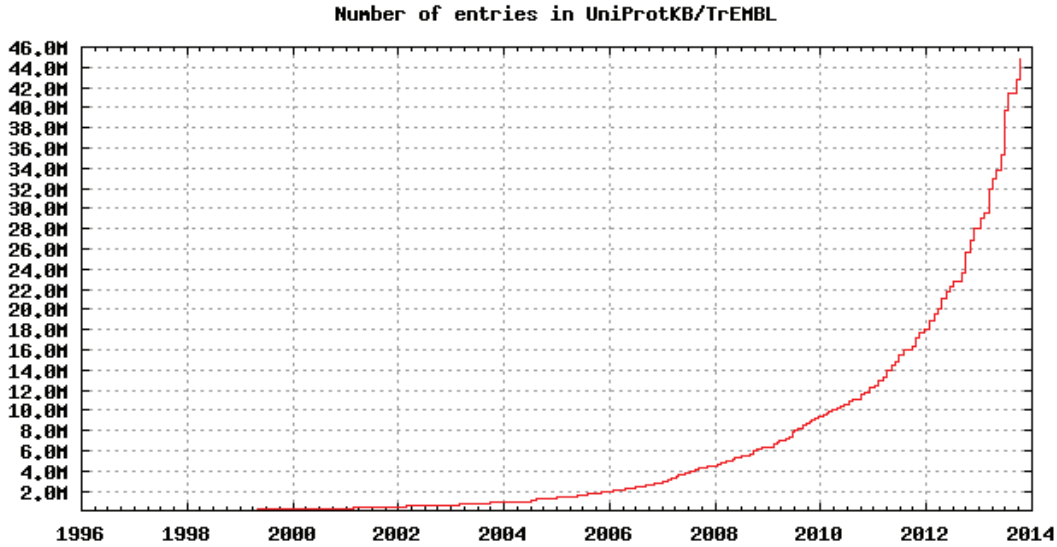

Figure 1. *The growth of the UniProtKB/TrEMBL database (UniProt, 2013).*

Automatically annotated protein databases have been a much needed solution to manage the increased amout of sequence data. However, it is not without problems. As much as 32% of sequences in UniProtKB/TrEMBL have been described as "Uncharacterized protein" or alike uninformative description. 41% of the sequences have no Gene Ontology (GO) class (See: How to define function) assigned and only 0.6% of sequences have experimentally defined GO annotation. In addition UniProtKB/TrEMBL or any other computationally annotated non-reviewed database, relies on sequence similarity to assign functional annotation to an uncharacterized sequence. This results in mistakes and errors and causes noise in functional annotations in the databases (See: Errors in databases)

With the growing number of sequence data produced it is not possible to rely solely on manually curated databases, but the impacts of falsely annotated proteins can be drastic to results and can ultimately lead to publication of false research outcomes (Percudani et al. 2013, Brenner 1999, Liberal et al. 2013, Nadzirin et al. 2012). Therefore methods for high throughput functional annotation with reliable results are needed. This can be accomplished by utilizing more sensitive algorithms in sequence similarity searches as well as using more sophisticated methods for protein function definition.

1.2 How to define function

Protein function can be defined in various ways. The most commonly used is the human written description line (DE) that defines the function in one free text sentence. These can be obtained from for example, an UniProtKB entry. In computational analysis DE line usage is challenging because of its discrete vocabulary and synonyms that are hard to render computer legible. Therefore a controlled vocabulary with well-defined relationships in describing functions is needed.

One controlled vocabulary is the Enzyme Commission classification (EC) which defines enzyme-catalyzed reactions in a well defined, structured way (Barrett, 1995). EC classification is based on a four level hierarchy where the first level is the most abstract, defining the general class of the enzyme group and level four is the most specific in details. EC numbers are a convenient way of describing a function in a computationally readable way, but EC numbers are really limited in databases. The Enzyme database (Bairoch, 2000) is the most comprehensive database for EC annotation, but it only has 224 202 proteins annotated with 5277 active entries (Release 13-Nov-13).

Another vocabulary is the MIPS Functional Catalogue (FunCat) which was originally was created at the Munich Information Center for Protein Sequences (MIPS) for the purposes of the *Saccharomyces cerevisiae* genome project and at that time it contained only categories required to describe yeast biology (Ruepp, 2004). Today MIPS FunCat contains 27 main classes, each organized in a hierarchical tree topology. The main classes cover features like *metabolism, development, transcription* etc. Terms are presented with double digits and the usage of terms is similar to the EC numbering system. Each double digit number is presented as a series from the most abstract to the most specific. For example main class *metabolism* has a term 01, the *amino acid metabolism* class has 01.01 and the most specific class 01.01.13 is *regulation of amino acid metabolism*. The number of levels is not restricted, but in most cases three or four levels are used. Some cases go up to six levels deep in hierarchy.

The dominant approach for machine-readable function annotations is Gene Ontology classification (GO) (Ashburner, 2000) which has more diverse nomenclature than EC. GO class defines proteins´ roles in Molecular Function (MFO) and Biological Process (BPO) categories. GO also includes Cellular Component (CCO) category that defines the subcellular location of a protein. GO classes are usually species independent, but there are some exceptional subgroups e.g. PAMGO that is Plant-Associated Microbe Gene Ontology group.

GO vocabulary terms are arranged in a hierarchical structure using a Directed Acyclic Graph (DAG). The main categories in a graph are the previously stated MFO, BPO and CCO. Protein function can be described by using one or more terms from each main category. Even though the CCO category is not directly connected to protein functionality, it is found to be important since protein do not work solely in a vacuum or a saline solution, they can only work within the context of a living cell (Friedberg, 2006). The CCO category also helps to functionally annotate moonlighting proteins, proteins that have multiple functionalities according to their subcellular localization (Jeffrey, 2003 & 2009).

DAG structure presents GO terms as nodes in the graph where each node may have one or more parents forming a tree like structure. Each parent node has a child node that defines a more specific function than the parent. An example of GO structure is presented in Figure 2. From a leaf node one could move only towards the root node using parent classes. Due to the acyclic nature of the graph, looping structures are not allowed.

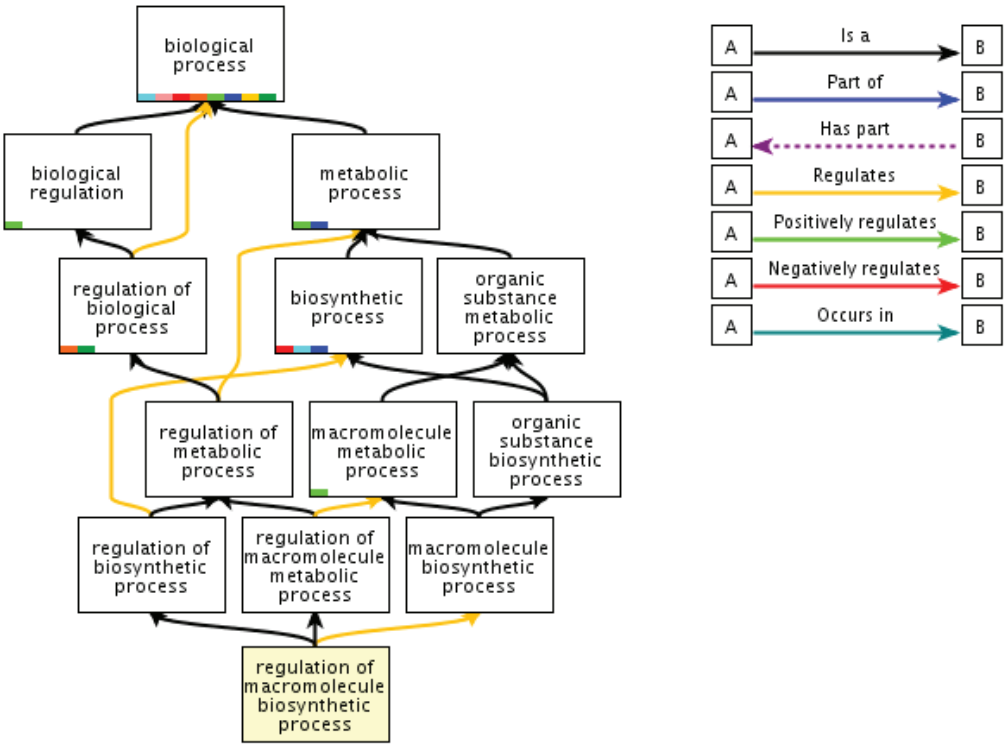

Figure 2. *Example of a GO structure with a GO class "regulation of macromolecule biosynthetic process" (GO:0010556) that is a term under "biological process" category. (AmiGO, http://amigo.geneontology.org)*

When a GO term is assigned as an annotation to a protein sequence, an evidence code is also assigned to show the method of annotation. For example a computationally assigned GO class would get an IEA (Inferred from Electronic Annotation) evidence code. A list of GO evidence codes and a brief explanation of evidence codes can be found in Table 2 (source: http://www.geneontology.org).

Table 1. *Gene Ontology evidence codes (source: http://www.geneontology.org/GO.evidence.shtml)*

Experimental Evidence Codes	
EXP	Inferred from Experiment
IDA	Inferred from Direct Assay
IPI	Inferred from Physical Interaction
IMP	Inferred from Mutant Phenotype
IGI	Inferred from Genetic Interaction
IEP	Inferred from Expression Pattern

Computational Analysis Evidence Codes	
ISS	Inferred from Sequence or Structural Similarity
ISO	Inferred from Sequence Orthologuey
ISA	Inferred from Sequence Alignment
ISM	Inferred from Sequence Model
IGC	Inferred from Genomic Context
IBA	Inferred from Biological aspect of Ancestor
IBD	Inferred from Biological aspect of Descendant
IKR	Inferred from Key Residues
IRD	Inferred from Rapid Divergence
RCA	inferred from Reviewed Computational Analysis
Author Statement Evidence Codes	
TAS	Traceable Author Statement
NAS	Non-traceable Author Statement
Curator Statement Evidence Codes	
IC	Inferred by Curator
ND	No biological Data available
Automatically-assigned Evidence Codes	
IEA	Inferred from Electronic Annotation
Obsolete Evidence Codes	
NR	Not Recorded

Evidence codes are a convenient way to estimate the reliability of a GO annotation. IEA is the only evidence code that is not validated by an expert and is therefore the most unreliable. Experimentally validated GO classes (other than IEA or NR) are rare (0.6% of UniProtKB entries) because of complexity and expensiveness of experimental annotation methods. It is noteworthy that the evidence code is not GO class specific, but sequence specific.

As only 59% of sequences in databases (UniProt) have any GO classes assigned and from those the majority are unreliable due to non-experimental validation there is a need for a method that is able to improve the quality of protein function definition. A sophisticated method is described in Publication II.

Since this thesis is focused on DE and GO annotation, EC and MIPS FunCat prediction are not covered in this thesis.

1.3 Sequence similarity implies function similarity

The central hypothesis in functional annotation via homology has been that during speciation corresponding proteins between species tend to keep their original functionality. Therefore functionally characterized proteins from species X could be used for functional annotation of homologue uncharacterized proteins in species Y.

One of the first practical examples of functional annotation by sequence homology was published 1983, when Gallwitz et al. deduced p21 products of the human c-has/bas proto-oncogenes by using sequence homology between the yeast and human. (Gallwitz et al. 1983)

1.3.1 Orthologue and paralogue

Homologous genes descend from a common ancestor and should not be mixed with analogous genes that were developed similar independently from each other. Homologous proteins are divided into two subgroups: orthologue and paralogue (Figure 3). Orthologues and paralogues are expected to be two fundamentally different types of homologous genes that evolved, correspondingly, by vertical lineage from a single ancestral gene and by duplication (Koonin et al. 2005). For example, the β chain of hemoglobin is a paralogue of the hemoglobin α chain and of myoglobin as they evolved from the same ancestral globin gene through repeated gene-duplication events. (Gogarten et al. 1999)

Paralogues can be divided into two subgroups: in-paralogues and out-paralogues. In-paralogues are the result of a gene duplication that occurred after any speciation events and the paralogue pair is found only within one species. Out-paralogues are then genes that duplicated before speciation and the paralogue pair can be found in multiple species.

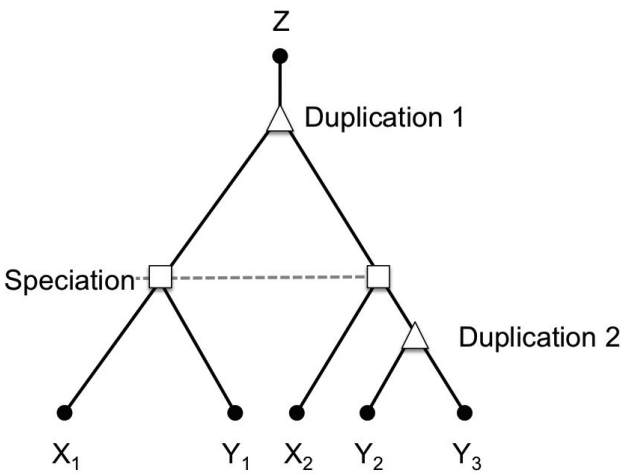

Figure 3. *Definition of orthologues and paralogues. In this a hypothetical tree, the ancestral gene in species Z is located in the root and it undergoes gene duplication (duplication 1). Next a speciation event occurs and leads to species X and Y. Finally a gene duplication event (duplication 2) in species Y occurs. Genes X_1 and Y_1 are one-to-one orthologues. X_1 and X_2 are out-paralogues since the gene duplication event occurred before the speciation event. Genes Y_2 and Y_3 are in-paralogues as the gene duplication happened after speciation; they are also co-orthologues to X_2.*

Especially orthologues are expected to keep the original functionality because of selective pressure. When orthologues continue in their original roles in an organism, paralogues are released from selective pressure and are expected to diverge by function from the original copy. However, this hyphotesis has been questioned over the last few years. It has shown that the mechanism is not as simple as expected. According to some studies orthologues and paralogues may differ or conserve their original functionality at the same rate (Studer et al. 2009). In some studies paralogues have been found to be more similar by function than orthologues (Nehrt et al. 2011).

1.3.2 Methods for sequence similarity

Homology assignment using computational methods requires statistically significant sequence similarity between two sequences. Searching for homology candidates starts with the alignment of the query sequence against every other sequence in the sequence database. Some of the first sequence alignment tools that became popular in research community were based on Needleman-Wunsch (Needleman & Wunsch 1970) and Smith-Waterman (Smith & Waterman 1981) algorithms.

The Needleman-Wunsch algorithm is used in bioinformatics to create optimal global alignment of nucleotide or amino acid sequences. Global alignment means that two sequences are aligned at full sequence length. Needleman-Wunsch was the first algorithm that utilized dynamic programming in sequence alignment. Dynamic programming was first presented in its modern form at 1954 (Bellman et al. 1954) and it is a method that breaks down the alignment problem into simpler subproblems and tries to find the optimal overall solution by combining results from subproblems. An example of dynamic programming alignment score matrix between sequences a = AGCACACGA and b = ACACACTGA is visualized in Table 2.

Dynamic programming works as follows: the matrix is first built so that every cell $S(i,j)$ in matrix $S(a,b)$ is given a score that is

$$(1) \qquad max\{S(i\text{-}1,j)\text{-}p, S(i,j\text{-}1)\text{-}p, S(i\text{-}1,j\text{-}1)+w\}$$

Where p is the gap penalty and w is the score for a match or a mismatch. The matrix is filled starting from upper left corner.

When the whole matrix is filled with corresponding scores, the shortest path is searched for between cells $S(0,0)$ and $S(n,m)$. The shortest path starts from $S(n,m)$, where n is the length of a and m is the length of b. The shortest path is searched for by backtracking from $S(n,m)$ and moving into a cell that has the $max\{S(i\text{-}1,j), S(i,j\text{-}1), S(i\text{-}1,j\text{-}1)\}$ until $S(0,0)$ is reached.

Table 2. *Example of dynamic programming. Here the gap penalty is -1 and the score for a match is +2. A green colour indicates the shortest path between S(0,0) and S(n,m). Negative cells are rendered to zero.*

$S(a,b)$ =

	-	A	C	A	C	A	C	T	G	A
-	0	0	0	0	0	0	0	0	0	0
A	0	2	1	3	2	4	3	2	1	3
G	0	1	1	2	2	3	3	2	4	3
C	0	0	3	2	4	3	5	4	3	3
A	0	2	2	5	4	6	5	4	3	5
C	0	1	4	4	6	5	8	7	6	5
A	0	3	3	6	5	7	7	7	6	8
C	0	2	5	5	7	6	9	8	7	7
G	0	1	4	4	6	6	8	8	10	9
A	0	3	3	6	5	8	7	7	9	12

The alignment is constructed from the shortest path as follows: Diagonal movement implies a match or a mismatch, vertical movement implies a deletion and horizontal movement implies an insertion.

a: AGCACAC-GA
b: A-CACACTGA

The Smith-Waterman algorithm is a variation of Needleman-Wunsch algorithm and is used for local alignments whereas Needleman-Wunsch is a global alignment algorithm. The main difference between these two methods is that in the Smith-Waterman algorithm all negative scoring cells are rendered to zero and the backtracking starts from the highest scoring cell and backtracking continues until a cell with zero score is met.

Smith-Waterman and Needleman-Wunsch algorithms give optimal global and local alignments. However, as they are computationally demanding, they are not useful tools for querying huge sequence databases. Therefore the most used tool for large database querying is the Basic Local Alignment Search Tool (BLAST) (Altschul, 1997), which relies on heuristic operations to enable fast query times.

In the BLAST algorithm a set of *k* length seed words are created by sliding *k* length window through the query word. For example from the query word `PROTEIN` we acquire the following 3 length seed words:

```
Query word: PROTEIN
Seed word1: PRO
Seed word2:  ROT
Seed word3:   OTE
Seed word4:    TEI
Seed word5:     EIN
```

In the next phase the BLAST algorithm makes modifications to the original seed sequences creating new neighbourhood seed words and keeps only those new seed words whose alignment score against original seed word exceeds the preset threshold *T*. BLAST uses substitution matrixes (PAM, BLOSUM, etc.) in scoring the alignments between new and original seed words.

Substitution matrixes describe how likely a nucleotide or an amino acid is to be replaced by another over evolutionary time. Substitutions occur more frequently between residues that have similar biochemical properties, e.g. a hydrophilic residue such as Asparagine is more likely to be replaced by another residue with similar biochemical properties like Glutamine. Substitution matrixes are used in sequence alignment to score mismatching residues.

Then BLAST scans the database sequences with the complete seed word set. Only exact matches are taken into account in the scanning process. Sequences that have exact match between them are considered as High-scoring Segment Pairs (HSP). (See Figure 4) Exact matches clustered tightly within one target sequence are merged as a longer new region. Finally the alignments are extended from both ends of the new regions until no sequence similarity is detected.

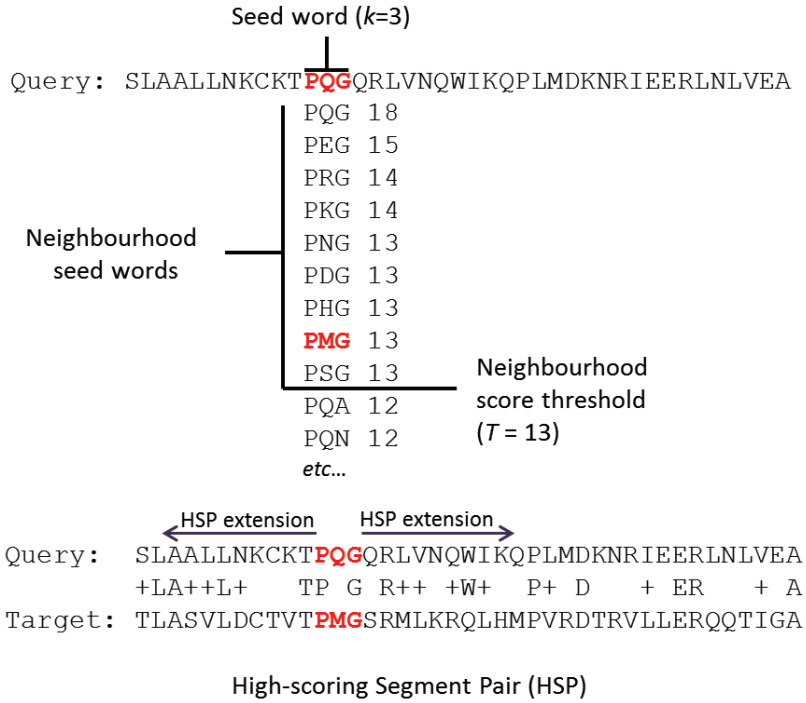

Figure 4. *Example of BLAST seed word construction and HSP matching. In this HSP match it is noteworthy that the matching seed word (**PMG**) is a modification of original seed word (**PQG**). (Picture modified from http://www.ncbi.nlm.nih.gov/Class/MLACourse/Modules/BLAST/images/BLAST_algorithm.gif)*

Next the significancy of the alignment is evaluated by calculating expectation value from the HSP alignment score. Expectation value of a HSP is the number of times that an unrelated database sequence with higher or same HSP score would occur by chance. In last stage gapped Smith-Waterman alignments are calculated between the query sequence and the HSP target sequences.

BLAST uses heuristics in the finding of similar sequences by using short exact matches instead of complete full length alignment. Therefore the BLAST result list may not be optimal but in most cases it is sufficient and fast compared to Smith-Waterman and Needleman-Wunsch algorithms. BLAST is still the most commonly used tool for sequence similarity search even though more sophisticated methods exist (e.g. Kent 2002, Edgar 2010).

The growing number of sequence data sets demands for even faster algorithms. A competing method 100 times faster than BLAST is described in Publication I.

1.4 Review of function annotation methods

Functional annotation has become one of the cornerstones in bioinformatics research. Again, due to high-throughput sequencing methods, demand for reliable methods for functional annotations of produced sequence has significantly increased. Figure 5 shows the growth of numbers of publications per year in the field of functional annotation and highlights the growing interest in this field.

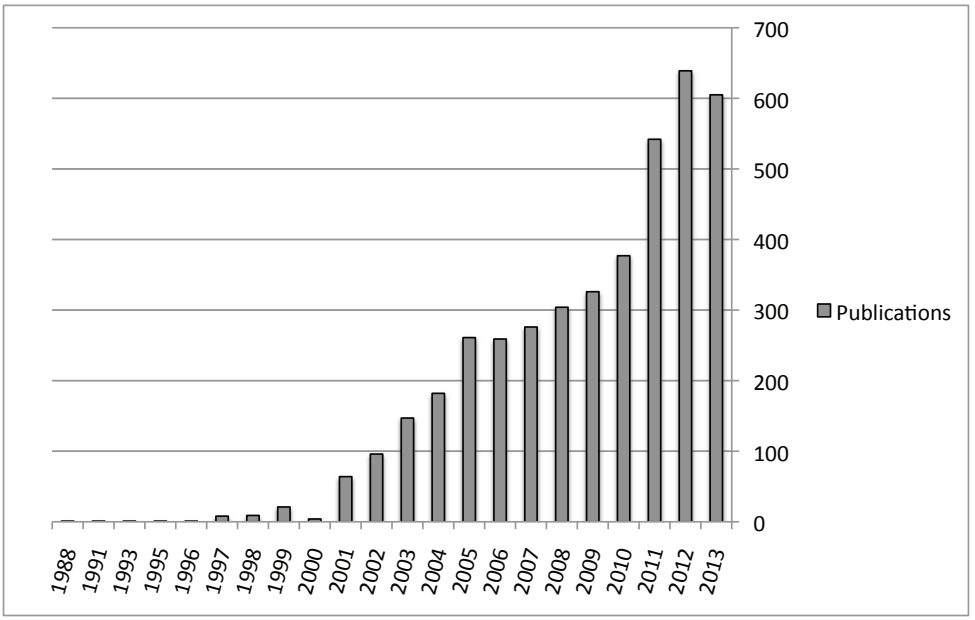

Figure 5. *Publications in PubMed per year with keywords "functional" and "annotation", December 2013. (source: PubMed, https://www.ncbi.nlm.nih.gov/pubmed)*

This section discusses the different methods used for description prediction, gene ontology prediction and family, feature and profile prediction methods in use today.

1.4.1 Description prediction methods

BLAST is one of the first and most frequently used tools for querying similar sequences from databases. The fact that the original Gapped-BLAST and PSI-BLAST article by Altschul et al. (Altschul et al. 1997) has been cited almost 49000 times by November 2013 gives some perspective of how routinely BLAST algorithm is used. The main reason for researchers using BLAST to query protein sequences is to find information about functionality (Friedberg et al. 2006).

1.4.1.1 Nearest neighbour

There are multiple ways to extract information from a Blast result list. One of the most used is the nearest neighbour method a.k.a. Best Blast Hit (BBH) where only the first hit from the BLAST result list is considered has become almost a standard when retrieving information from a BLAST result list (Radivojac et al. 2013). BBH methodology implies that the best hit is the closest relative in the database to the query sequence and therefore the best source to transfer functional information.

Since many sequences lack the informative annotation (DE or GO), the best Informative BLAST Hit (BIBH) method is used instead BBH. In the BIBH method the result list is processed from best first order and if the sequence has an uninformative (e.g. "Uncharaterized protein") or no annotation, it will be discarded. This process continues until the first informative hit is found. Informative hit includes here informative description or GO classes assigned to a sequence. One way to calculate description information value is described in Publication II.

Position-Spesific Iterated BLAST, or PSI-BLAST is a tool to find distant homologues. PSI-BLAST is more sensitive than regular BLAST in finding distant evolutionary relationships: A search starts with a normal BLAST query for the sequence database, but on the next iteration round PSI-BLAST calculates a position-specific scoring matrix (PSSM) or a profile from the multiple sequence alignment of BLAST result sequences. PSI-BLAST then performs a new search against the database using PSSM or a profile as a query. Depending on the number of iteration rounds used, PSI-BLAST updates the PSSM or the profile at every turn based on newly detected result sequences. (Altschul et al. 1997) BBH or BIBH methods are also used with PSI-BLAST.

1.4.1.2 K-nearest neighbours

When using BBH or BIBH methods with BLAST or PSI-BLAST results, there is a notable risk of acquiring incorrect annotations in cases where the source is misannotated (See: Errors in databases). Errors and lack of information causes noise in databases that can be difficult to detect with the BBH or BIBH methods. The assumption that the BBH or BIBH is the closest homologue in the database proves to be wrong in many cases, particularly when there are only few homologues in the database (Koski et al. 2001).

The most successful methods using BLAST result list as an input for functional annotation are based on the processing of the whole list or the k-nearest neighbours instead of using BBH or BIBH. As Rentzsch et al. have stated: "It seems intuitive that more, and more correct, annotations could be transferred by looking at all relatives in this list – not only the top hit." (Rentzsch, 2009) A method that uses k-nearest neighbours (KNN) is described in Publication II.

The KNN method means in practice that a k amount of sequences in the best first order is selected from the full result list. KNN produces a list of hits that might be quite heterogeneous by functional annotations. Therefore some postprosessing is usually needed.

Using KNN methods brings some new problems when selecting related information. Some of the information could be in conflict with others and therefore can not be pooled together. Various methods are used to group information. Clustering sequences by similar GO composition is one of the most used (e.g. Kankainen, 2013, Pehkonen, 2005, Falda, 2012), but with the right tools text similarity has proved to be an even more efficient method because of its better coverage of DE annotations (Publication II).

Clustering data enables statistical testing against the database or the rest of the result list. Clustering and statistical testing is computationally more demanding than using BBH or BIBH. However in noisy databases clustering of similar sequences into functionally congruent groups and statistical testing has proven to enhance the prediction accuracy remarkably. Therefore the use of computationally demanding methods is justified (Publication III).

1.4.2 Gene ontology prediction methods

Since GO is the dominant vocabulary for functional annotation, most of the function prediction tools are optimized strictly to predict GO classes only. Multiple methods to exploit GO have been introduced, including functional annotation from sequence homology (Jensen, 2002, Wass, 2008, Martin, 2004, Hawkins, 2006, Clark, 2011), evolutionary relationships and the genomic context (Pellegrini, 1999, Marcotte, 1999, Enault, 2005, Engelhardt, 2005, Gaudet, 2011, Zhao, 2013), protein-protein interactions linkages (Deng, 2003, Letovsky, 2003, Vazquez, 2003, Nabieva, 2005, Hung, 2012), protein structure data (Pazos, 2004, Pal, 2005, Laskowksi, 2005), microarrays (Huttenhower, 2006) or a combinations of different methods and datatypes (Troyanskaya, 2003,

Lee, 2004, Costello, 2009, Kourmpetis, 2010, Sokolov, 2010). This section discusses some of the GO prediction tools. More GO prediction tools are presented in Publication III.

Blast2GO (Conesa, 2005) is one of the most frequently used methods for GO annotation. Blast2GO analysis is based on enrichment analysis of GO classes mapped to BLAST result hits. For enrichment analysis Blast2GO uses the Gossip software package (Blüthgen, 2005). The resulting GO classes are listed in smallest hypergeometric p-value first order.

Another widely used GO annotation tool using the whole Blast result list as an input is GOtcha (Martin, 2004). The GOtcha method uses hierarchical GO structure (DAG, See: How to define function) in calculation of scores for a GO class. For each result in the Blast list, Gotcha maps all the hits to GO classes. Each GO class is assigned a R-score:

(2) R-score = max{–log(Blast expect value),0}

For each parent GO class the R-scores from leaf nodes are cumulated so that the root node will have a cumulative sum of all the R-scores from the GO leaf nodes in the tree. The cumulative R-score at the root node is called the C-score. In the last step all the scores in the tree are normalized by using the C-score as a divider. The GOtcha method is described in Figure 6.

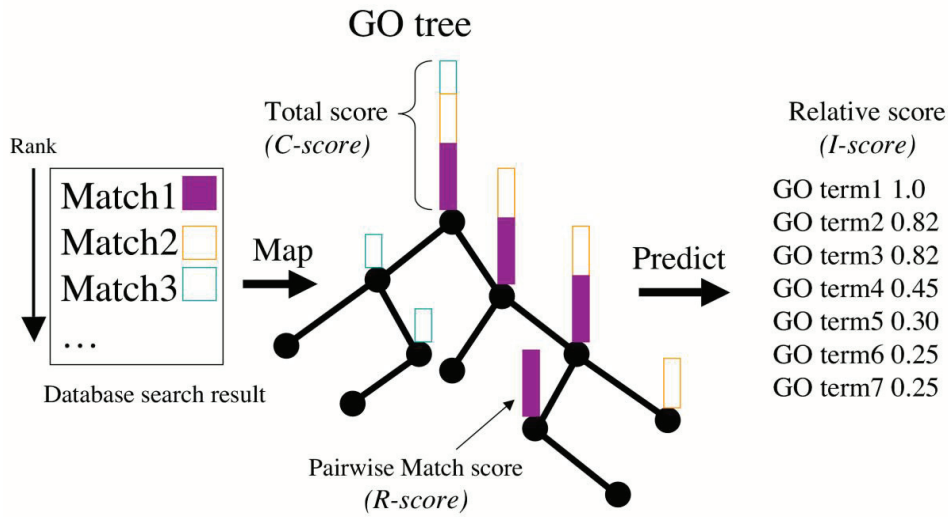

Figure 6. *The Gotcha method. The database search results are Blast alignments. The R-score from each Blast result is cumulated via parent nodes to the root of the tree. The cumulative R-score at the root of the tree is called the C-score. The I-score is calculated by dividing the R-scores in every node with the C-score. (Martin, 2004)*

Argot2 (Falda, 2012) is a modification of GOtcha methodology. In the Argot2 method the input is acquired from BLAST against UniProtKB query and HMMER against a Pfam (See Table 3) query. Since the GOtcha method favours large GO classes near the root node, Argot2 uses the Information Content (IC) score in counter-weighting of Z-scores (equivalent to the R-score in GOtcha) (Figure 7):

(3) IC = -log p(c)

where p(c) is the relative frequency of all of the descendants of the GO term c in Gene Ontology Annotation (GOA) database (Dimmer, 2011). Argot2 groups similar GO terms together by using GO semantic similarity distances (Dimmer, 2011 & Falda, 2012).

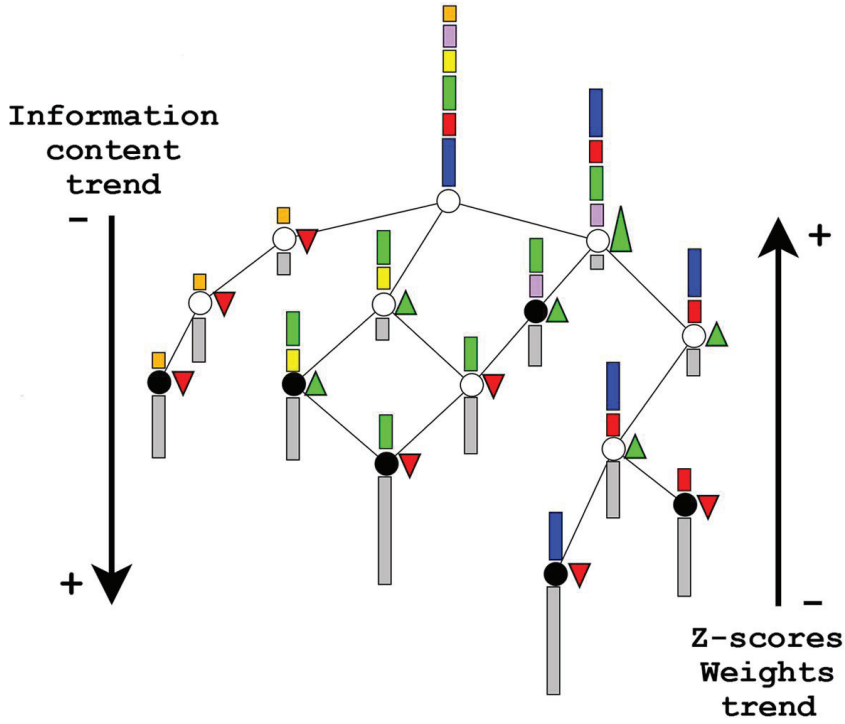

Figure 7. *Weighting the GO nodes and GO Information content, the Argot2 algorithm calculates the Information Content (IC) (gray bars) of the nodes in the graph and their cumulative weights (coloured bars) derived from the BLAST scores. (Fontana, 2009)*

Jones-UCL is a method that uses multiple data sources as an input and makes predictions using a score function called Combined Graph-Information Content similarity (COGIC) (Cozzeto, 2013). COGIC uses the data sources described in Table 3 in the prediction of GO classes:

Table 3. *Data sources used in Jones-UCL GO prediction method. (source: Cozzetto, 2013)*

Method:	Description:
PSI-BLAST	GO term mapping to PSI-BLAST search results with 3 iterations against Uniref90 database (Suzek,2007).
UniProtKB/Swiss-Prot keywords	GO term mapping to Swiss-Prot entry descriptive text, keywords or comments using the naïve Bayes text-mining approach (McCallum, 1998).
Amino acid trigram mining	GO term mapping to amino acid trigrams from UniProtKB/Swiss-Prot amino acid sequences using naïve Bayes classification.
Sequence features	GO term prediction using the FFPRED program (Jensen, 2003). FFPRED uses Support Vector Machines (SVMs) to predict GO classes from feature combinations of secondary structure elements, signal peptides, glygosylation sites and several others.
Orthologous groups	GO term predictions using eggNOG database collection of orthologueous groups (Muller, 2010).
Profile-profile comparisons	PSSM profile for every sequence in UniProtKB/Swiss-Prot is calculated as well for query sequences. After profile-profile comparisions, GO terms were predicted by evaluation profile-profile alignments with a neural network.
High-throughput data sources	GO prediction using the FunctionSpace method (Lobley, 2010). FunctionSpace uses SVMs to predict GO terms from an 11-dimensional feature space. The 11 dimensions represent pairwise sequence similarity, predicted cellular localization, secondary structure similarity, transmembrane topology, disordered segment features, sequence-derived domain architecture, structure-based domain architecture, sequence domain fusion patterns, structural domain fusion patterns, protein-protein interactions and microarray data.

The Jones-UCL program is not yet publicly available (21.11.2013).
Our method called PANNZER is described in publications II and III.

1.4.3 Family, feature and profile prediction methods

Another widely used tool for the functional annotation of protein is InterProScan (Quevillon et al. 2005) which categorizes proteins into families and recognizes protein domains, repeats and functional sites from peptide sequences. Instead of querying the neighbours of every query separately, InterProScan uses precalculated models of families and features. It incorporates protein features from 11 InterPro member databases (Table 4).

Table 4. *The databases included in InterPro Consortium. (source:http://www.ebi.ac.uk/interpro)*

Database:	Description:
PROSITE	PROSITE is a database of protein families and domains. It consists of biologically significant sites, patterns and profiles that help to reliably identify to which known protein family a new sequence belongs.
PRINTS	PRINTS is a compendium of protein fingerprints. A fingerprint is a group of conserved motifs used to characterise a protein family or domain.
ProDom	The ProDom protein domain database consists of an automatic compilation of homologous domains. Current versions of ProDom are built using a novel procedure based on recursive PSI-BLAST searches.
Pfam	Pfam is a large collection of multiple sequence alignments and hidden Markov models covering many common protein domains.
SMART	SMART (a Simple Modular Architecture Research Tool) allows the identification and annotation of genetically mobile domains and the analysis of domain architectures.
HAMAP	HAMAP stands for High-quality Automated and Manual Annotation of Proteins. HAMAP profiles are manually created by expert curators. They identify proteins that are part of well-conserved proteins families or subfamilies.
TIGRFAMs	TIGRFAMs are collections of protein families, featuring curated multiple sequence alignments, hidden Markov models (HMMs) and annotation, which provides a tool for identifying functionally related proteins based on sequence homology.
PIRSF	The PIRSF protein classification system is a network with multiple levels of sequence diversity from superfamilies to subfamilies that reflects the evolutionary relationship of full-length proteins and domains.
SUPERFAMILY	SUPERFAMILY is a library of profile hidden Markov models that represent all proteins of known structure. The library is based on the SCOP classification of proteins: each model corresponds to a SCOP domain and aims to represent the entire SCOP superfamily that the domain belongs to.
Gene3D	The CATH-Gene3D database describes protein families and domain architectures in complete genomes. Protein families are formed using a Markov clustering algorithm, followed by multi-linkage clustering according to sequence identity.
PANTHER	PANTHER is a large collection of protein families that have been subdivided into functionally related subfamilies, using human expertise. These subfamilies model the divergence of specific functions within protein families, allowing more accurate association with function, as well as inference of amino acids important for functional specificity.

Protein signatures, including domains, repeats, motifs, etc., are predictive models based on similarities among proteins that have the same structure or function. The signatures are in some cases manually curated with GO classes. The InterPro consortium annotates all the sequences in UniProtKB annually and provides the annotations in the Interpro database (Mulder et al. 2007).

One of the biggest restrictions of feature-based annotation is that the methods used to assign functionality are based on the discovery of each functional sites separately. For example if domain A is annotated with GO:X and domain B is annotated with GO:Y, then GO classes X and Y are assigned to a query sequence if domains A and B are found. This method does not take into account the combined effects when domain A and domain B appear together. Domains A and B together could expose GO:Z, which is not recognized if the domains are discovered separately. Domains can be imagined as Lego bricks that can be combined in various ways to build protein with completely new functions (Buljan et al. 2009). The function of a mature protein can be more than the sum of its parts and therefore a protein and its functional subunits should be inspected as a whole, not separately (Forslund et al. 2011).

An advantage of family databases is that since every family member is annotated annually with family annotation, over time there are fewer uncharacterized protein entries. Another advantage is that the use of precalulated models from families and features is a more sensitive method than using sequence similarity. In precalculated families, family boundaries are more easily revealed because of the global clustering of family members.

1.5 Errors in databases

There are lots of errors in non-redundant (NR) unreviewed sequence databases like UniProtKB/TrEMBL or GenBank. Schnoes et al. estimated the error level of functional annotations in these databases to be 5%-63% (Schnoes, 2009). The current trend with correct and incorrect annotations in public databases is presented in Figure 8. One of the earliest papers suggest that the error rate is at least 8% (Brenner 1999), but further research has shown that in reality it is close to 37% (Devos et al. 2001). In the Gene Ontology database the error rate has been estimated with curated GO classes at between 28% and 30%, and with computationally created annotations as high as 49% (Jones et al. 2007). As more sequences enter the database, more are annotated using best BLAST hit based function transfer; errors begin to amplify throughout the database and degrade the quality of increasing number of annotations (Gilks et al. 2002 & 2005, Bork et al. 2000).

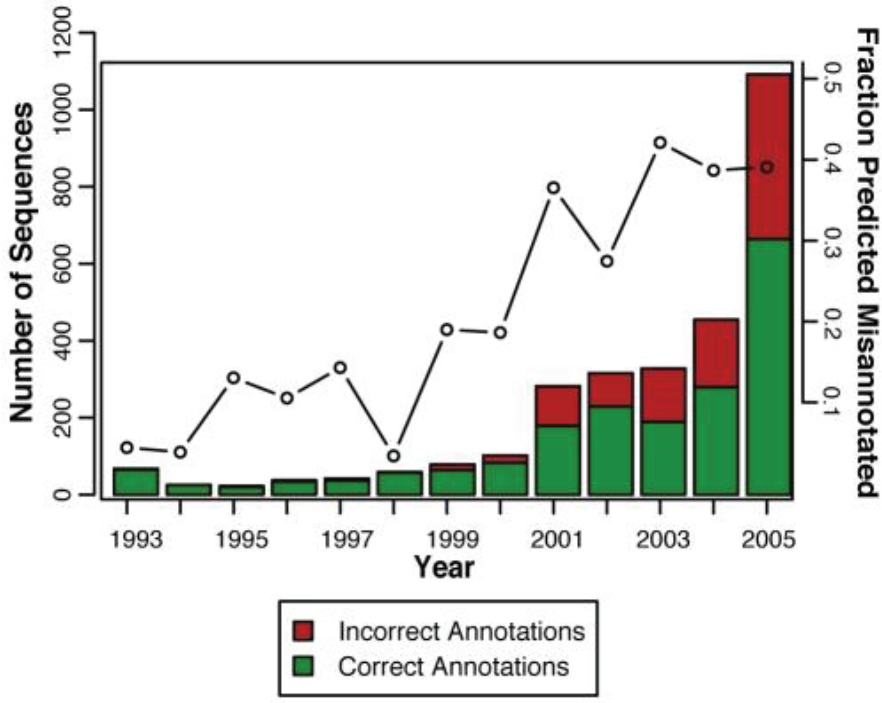

Figure 8. *Sequences are plotted by the publication year when they were submitted to the database. The number of sequences found to be correctly annotated is indicated with green. The number of sequences found to be misannotated is indicated with red. (Schnoes et al. 2009)*

2 Aims of the present study

The focus of this study was to develop an improved functional annotation tool for prokaryotic and eukaryotic organisms. The motivation came from the genome projects that took place at University of Helsinki, Viikki biocampus between the years 2008 to 2012. The research covered improvements in computational performance and prediction accuracy. This thesis is constructed from the following publications:

I Publication I covers the main problem in high-throughput functional annotation. Commonly used methods do not meet all the requirements, so new fast and sensitive methods are needed. The published method improved performance speed of sequence retrival hundred fold compared to the most commonly used tool BLAST and is as sensitive as BLAST when sequence similarity is above 50%.

II Publication II covers the problem of retrieving relevant information from result lists created with sequence retrieval tools. The results show that k-nearest methods improve the prediction accuracy significantly combined with statistical methods for error detection. The publication presents novel methods for weighted statistical testing and calculation of information value for descriptions. The results show that the PANNZER method is significantly better than traditionally used methods e.g. BBH and BIBH.

III Publication III covers the evaluation of functional annotation tools on a large scale by using 'blind testing'. In total 54 state-of-the-art methods participated in CAFA challenge and the results show that there is lots of room for improvements. The overall prediction accuracy was unsatisfactory even with the best method. Biological Process categories were especially difficult to predict correctly. The publication gives a snapshot from the field of computational function prediction.

3 Materials & methods

3.1 SANS

As sequence databases grow in size so rapidly, algorithms used in sequence retrieval today may not be sufficient tomorrow. Faster and faster methods are needed to meet growing demand. The Suffix Array Neighbourhood Search (SANS) is a sophisticated sequence retrieval tool that uses suffix arrays in searching for matches. The SANS algorithm is orders of magnitude faster than the most popular methods used today.

Suffix arrays are an extremely powerful way to find exact matches between the query sequence and target sequences in a database. Reconstruction of the suffix array starts with an array of integers pointing to starting locations of suffixes in the original string `SEQ` (Table 5). The starting locations in the array are sorted by lexicographical order of suffixes resulting the final suffix array (Table 6).

Table 6. *Example of suffix arrays. The word `BANANAS$` has the following suffixes. `$` is a sentinel character that terminates the word.*

Suffix	Sequential index
bananas$	0
ananas$	1
nanas$	2
anas$	3
nas$	4
as$	5
s$	6
$	7

Table 6. *Suffixes are sorted in lexicographic order to acquire Lexical indexes of suffixes. The `$` character has the lexicographically smallest value. The sequential indexes sorted in lexical order is the suffix array.*

Suffix	Sequential index	Lexical index
$	7	0
ananas$	1	1
anas$	3	2
as$	5	3
bananas$	0	4
nanas$	2	5
nas$	4	6
s$	6	7

The query sequences and the sequences from the target database are concatenated into strings SEQ_q and SEQ_t respectively. The locations of the protein sequences in SEQ_q and SEQ_t are stored as pointers in an array.

The SANS algorithm uses two suffix arrays, the first is the regular suffix array (SA) where sequential indexes are sorted in lexicographical order (*l*) and the second is the inverse suffix array (ISA) where lexical indexes are in sequential order (*s*) (Table 7). These two arrays (SA & ISA) make it possible to jump between sequential and lexical orders. For construction of the SA the recursive sais-lite algorithm (Nong, 2009) is used. ISA is generated by scanning the SA using the relation ISA[SA[*l*]] = *l*.

Table 7. *Corresponding Sequential indexes and Lexical indexes form two arrays SA and ISA.*

	B	A	N	A	N	A	S	$
SA:	7	1	3	5	0	2	4	6
ISA:	4	1	5	2	6	3	7	0

An auxiliary index array SAP (Suffix Array vs. Proteins) is then created to store protein information of suffixes in lexicographical order. To compare SEQ_q and SEQ_t, all the locations for SEQ_q suffixes are determined in the SA and locations are stored in the ISA_{mapped} array (Figure 9A). According to the found SEQ_q suffix locations in ISA_{mapped}, k-neighbouring proteins are selected around the suffix. SANS then scores selected target proteins according to match counts between all the query sequence suffixes and target proteins suffixes (Figure 9B) and lists the target proteins largest score first order.

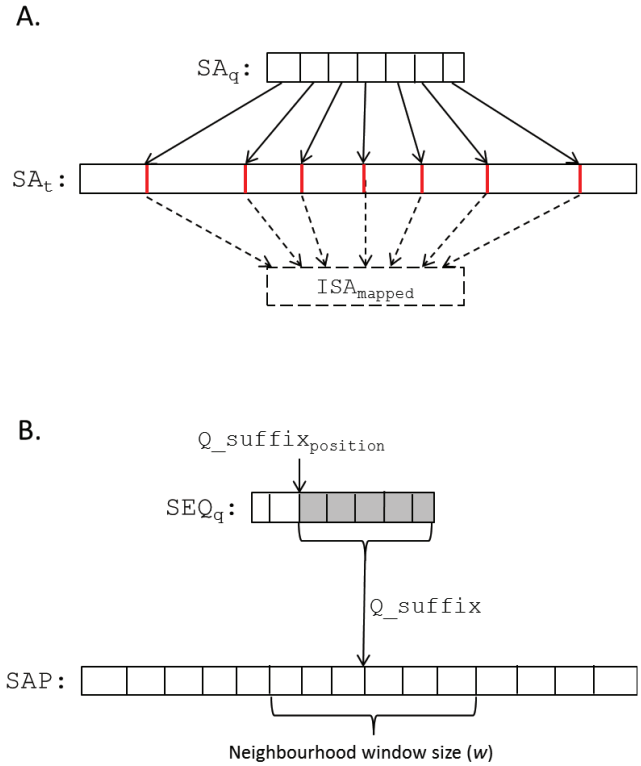

Figure 9. *A. Suffixes from SEQ_q are determined in SA_t and the locations are stored in ISA_{mapped} array. B. Searching for similar protein sequences. Location of query suffix is determined in the SAP array and the w amount of neighbouring target proteins are taken into consideration. For every target protein within the search window, matching suffixes are counted. Target proteins are printed according to the largest matching suffix count first order.*

3.2 PANNZER

The Protein ANNotation with Z-scoRE (PANNZER) methodology relies on making decisions based on k-nearest neighbours from a Sequence Similarity Result List (SSRL). An SSRL is obtained by using tools like BLAST or SANS against a UniProtKB protein database. The k-nearest neighbours are selected by using thresholds in sequence similarity, alignment coverage, bitscore and information content.

In the PANNZER methodology the obtained set of k-neighbours are re-scored by regression score that is calculated using sequence identity percentage, alignment coverage over query and target sequence and non-linear scoring for taxonomic distance between query and target species (Publication II, supplementary material).

In clustering of sequences from SSRL to functionally congruent groups, PANNZER uses the hierarchical clustering method with average linkage criteria. Functional similarity between two descriptions is calculated using a weighting score called "Term Frequency – Inverse Document Frequency" (TF-IDF) and cosine similarity. TF-IDF weight is a statistical measure used to evaluate how essential a word is to a document (in this case description) in a corpus (i.e. database). The importance of the word increases *pro rata* to the number of times a word occurs in the description but is counterweighted by the frequency of the word in the corpus. TF-IDF weighted cosine similarity in this thesis is called the Description Similarity Measure (DSM). DSM 1 means that two descriptions are identical and 0 denotes that two desriptions are completely different compared to each other. A few examples of DSM scores between descriptions are presented in Table 8.

Table 8. *Example of TF-IDF measures between description pairs in different bins.*

Description 1	DSM	Description 2	Bin
AMY-1-associating protein expressed in testis 1	0.971	AMY-1-associating protein expressed in testis 1-like	**>0.9**
Putative aryl-alcohol dehydrogenase AAD15	0.820	Similar to aryl-alcohol dehydrogenase	**0.9-0.8**
Acetoacetyl-CoA synthetase	0.704	Acetoacetyl-coenzyme A synthetase	**0.8-0.7**
Biotin carboxylase, chloroplastic	0.643	Acetyl-CoA carboxylase, biotin carboxylase	**0.7-0.6**
Xanthoxin dehydrogenase	0.557	Alcohol dehydrogenase	**0.6-0.5**
Benzoate--CoA ligase, peroxisomal	0.435	3-methylmercaptopropionyl-CoA ligase	**0.5-0.4**
Ethylene-responsive transcription factor ABR1	0.346	Wound-responsive AP2 like factor 1	**0.4-0.3**
Agamous-like MADS-box protein AGL18	0.237	Putative MADS domain transcription factor GGM9	**0.3-0.2**
Adrenodoxin-like protein, mitochondrial	0.146	Probable YAH1-Ferredoxin of the mitochondrial matrix	**0.2-0.1**
Protein arginine N-methyltransferase 1.6	0.051	Putative uncharacterized protein	**<0.1**

The clusters are scored by using second level regression score that is calculated by using Word Score (WS), Weighted Word Score (WWS) and Gene Set Z-score (GSZ). WS is a scoring function where every term in an SSRL is scored by summing bitscores from the descriptions where the term occurs. Then bitscore sums for every term in description are summed and the sum-score is normalized to the cumulative sum of bitscores of every description in the SSRL. WWS is a weighted version of WS score where weight is acquired from the Jaccard similarity index between

term occurrences in the cluster and the corpus. GSZ is a weighted version of the hypergeometric Z-score (Törönen, 2009).

Clusters are sorted in largest regression score first order and members of the best cluster are mapped to GO classes and EC numbers. The PANNZER methodology is presented in Figure 10.

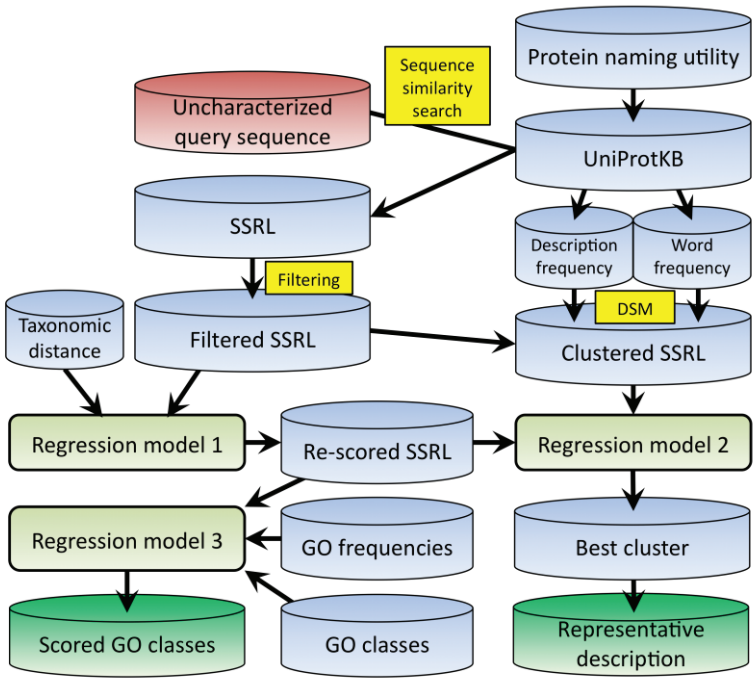

Figure 10. *The PANNZER program schema. The figure is from Publication II.*

3.3 CAFA 2011 Challenge

As more and more functional annotation tools are introduced annually, it is difficult to compare these methods in an unbiased manner. On 15th September 2010 a Critical Assessment of Functional Annotation (CAFA) experiment was released with 48,298 uncharacterized protein sequences that had to be functionally annotated with Molecular Function and Biological Process GO classes. The deadline for result submission was the 18th January 2011. After the submission deadline the target accumulation phase started. This ended in 14th December 2011. During the accumulation phase 866 out of 48,298 proteins were experimentally annotated with GO classes. This set was used as an evaluation set in the experiment.

In total 54 state-of-the-art methods from 23 research groups participated in the CAFA challenge. Since the correct answers were not known at the time the results were submitted, this testing can be considered to a blind experiment for all the participating methods. The evaluation was done using maximum value (F_{max}) out of F-measure that is a harmonic mean between precision and recall values (See: Publication III Online methods). F_{max} considers predictions in the full range from low to high sensitivity. The used evaluation metrics has its faults when large GO classes close to the root node are included (such as: Protein binding). The fact that F_{max} measure favours large GO classes close to the root node gives biased results in some cases. Large

GO classes near the root are the most abstract ones and therefore the easiest to predict. Very specific (i.e. more informative) classes deep in a tree are much challenging to predict correctly. Excluding targets with a large GO class as their only annotation from the evaluation set solved some of these problems.

A more detailed description about the CAFA 2011 challenge can be found in Publication III.

4 Results and discussion

4.1 SANS results

Evaluation of the SANS algorithm was done using three different datasets: Genome (4 173 proteins), Metagenome (6 050 065 proteins) and UniprotKB (18 748 263 proteins). SANS was the only method over SSEARCH, BLAST and USEARCH that was able to go through the whole UniprotKB dataset. Metagenome alignments were done against the UniprotKB/Swiss-prot database (533 049 proteins).

The Figure 11 shows that the SANS algorithm is almost as sensitive as BLAST when sequence identity is over 50%. Minimum sequence identities above the 40% to 70% level have been recommended for use in function transfer by various researchers (Devos and Valencia, 2000; Friedberg, 2006; Lee et al., 2006; Rost, 2002). The sensitivity rates shown fulfil the needed requirements with SANS.

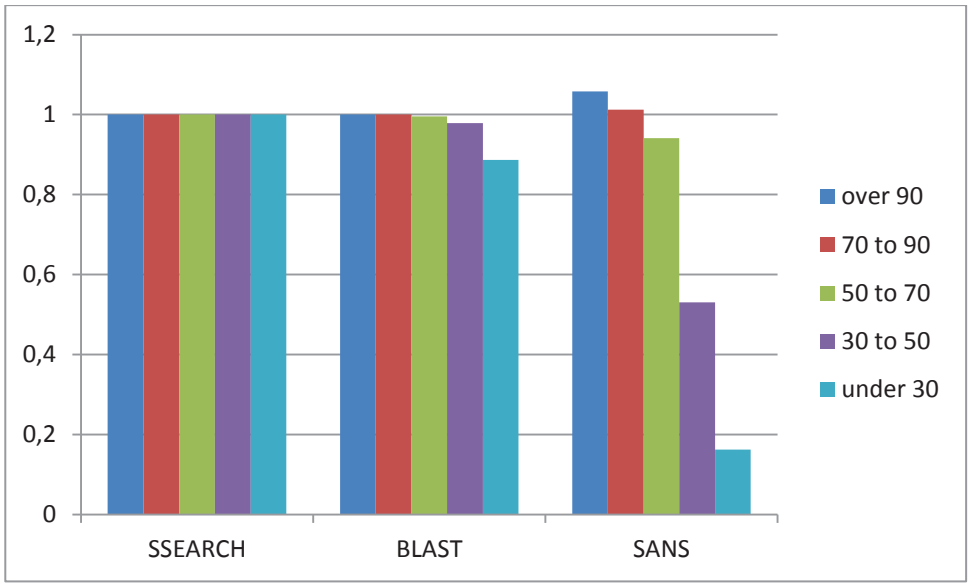

Figure 11. *Relative sensitivity of SSEARCH, BLAST, SANS. The evaluation was done against genome dataset. The top-1000 hits per query are evaluated. Hits are distributed in different bins by sequence identity. Bins are normalized to SSEARCH. The surprising super-performance of SANS in the "over 90" bin is due to different criteria being used to sort the result list.*

The time complexity of the database indexing is $\Theta(N+M)$, where N is the length of the query sequence and M is the length of the target database. The time complexity of the search is $\Theta(WN)$ where W is the width of the suffix array neighborhood and N is the size of the query set. The time complexity of database indexing (merge-sort) is linear, meaning that the overall execution time of the SANS algorithm grows linearly with data size. For example the expected-time computational complexity of BLAST is approximately $\Theta(aW + bN + cNW/20^W)$, where W is the number of seed words generated, N is the total number of amino acid residues in the database and a, b and c are constants (Altschul,1990). Then the main difference between SANS and BLAST time

complexity is that SANS time complexity is dependent on query set size, where as the BLAST time complexity of one search is dependent on the whole target database size.

With our evaluation datasets SANS proved to be the fastest sequence retrieval algorithm. SANS is able to calculate a score that is equivalent to a similarity score without traditional sequence alignment (See: Sequence similarity implies function similarity). This advantage makes SANS 10 times faster than USEARCH and 100 times faster than BLAST (Table 9).

Table 9. *Comparison of running times with SANS, BLAST, USEARCH and SSEARCH.*

Query set	Database	Program	Parameters	Time		
				Indexing[a]	Search	Alignment
Uniprot	uniprot	SANS	$W = 100$	3 h 22 m	30 h 4 m	2 h 19 m
Metagenome	swissprot	BLAST	–b 250		380 d	
		USEARCH	$a = 1$ $r = 8$		19 h 55 m	
		SANS	$W = 1$	57 m	13 m	48 m
	uniprot	SANS	$W = 1$	4 h 52 m	28 m	32 m
Genome	uniprot	SSEARCH	–s BL62 –f –11 –g –1 –E 1.0 –m 9C –z 3 –d 0		640 d	
		BLAST	–b 1000		100 h	
		parallel BLAST	10 processors, –b 1000		max 13 h/processor	
		SANS	$W = 100$	3 h 57 m	12 m	10 m
			$W = 1000$		23 m	52 m
			$W = 2000$		33 m	1 h 40 m

[a]Indexing time includes indexing both the query set and database from scratch.

Suffix arrays are considered to be excessive in memory usage. SANS implementation solves this problem by loading segments of the suffix array sequentially so that the whole array is eventually processed but the occupied memory stays within bounds. The used memory size is user defined and fixes the part size that is loaded into memory.

4.2 PANNZER results

4.2.1 Description prediction

Evaluation of the PANNZER tool was done using two different datasets. The first was a set of 2954 manually curated protein sequences of prokaryotic origin and the second was a set of 5115 manually curated protein sequences from eukaryotes. Both sets were extracted from UniProtKB/Swiss-Prot due to the low error rate in their functional annotations (Schnoes, 2009). Sequence and description redundancy were removed in order to avoid bias caused by large protein families (See: Publication II, Evaluation datasets). In a following prediction is considered to be correct if its DSM against the original annotation is more than 0.7 and incorrect if its DSM against the original annotation is less than 0.3. See Table 8 for examples of DSM scores between descriptions. The evaluation results are given below.

4.2.1.1 Prokaryotic dataset

The prokaryotic dataset evaluation was done by comparing PANNZER created results to results predicted with RAST (Aziz et al. 2008), BBH, BIBH and Blannotator (Kankainen et al. 2012) methods. It can be seen in Figure 12 that the PANNZER tool is able to predict most of the annotations correct (56.2%) and thus is the best performing method compared to RAST, BBH, BIBH and Blannotator. The second best prediction tool for DE annotation is Blannotator, that was able to give 50.6% correct annotations. Both PANNZER and Blannotator use k-nearest neighbours in function prediction. BBH and BIBH predicts 43.8% and 46.7% correctly, respectively. The RAST tool in this evaluation performs the worst, only having 31.5% of predictions correct. The low performance of the RAST tool could partly be explained by different phrasing of functions (synonymous words, euphemisms, etc.) in the FIGfam database compared to UniProtKB/Swiss-Prot. The PANNZER tool is able to predict 5.6% - 24.7% more cases correctly than the competing methods.

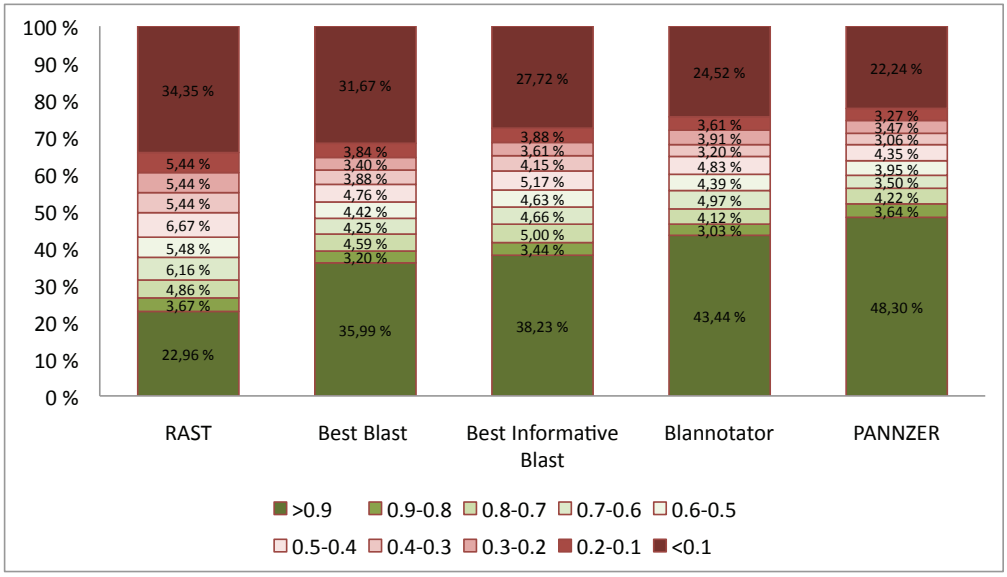

Figure 12. *Prediction results with the prokaryotic dataset. Hits with DSM score above 0.7 are considered to be correct and hits below 0.3 incorrect.*

4.2.1.2 Eukaryotic dataset

Since RAST and Blannotator tools are restricted only to prokaryotes, the eukaryotic dataset was annotated only using BBH, BIBH and PANNZER. Figure 13 shows that the PANNZER method outperforms BBH and BIBH significantly. The PANNZER method is able to predict 51.5% of the annotations correct, when BBH and BIBH predicts only 14.7% and 39.8% correctly, respectively. The PANNZER method is then able to predict 11.7% to 38.8% more correct annotations than BBH and BIBH.

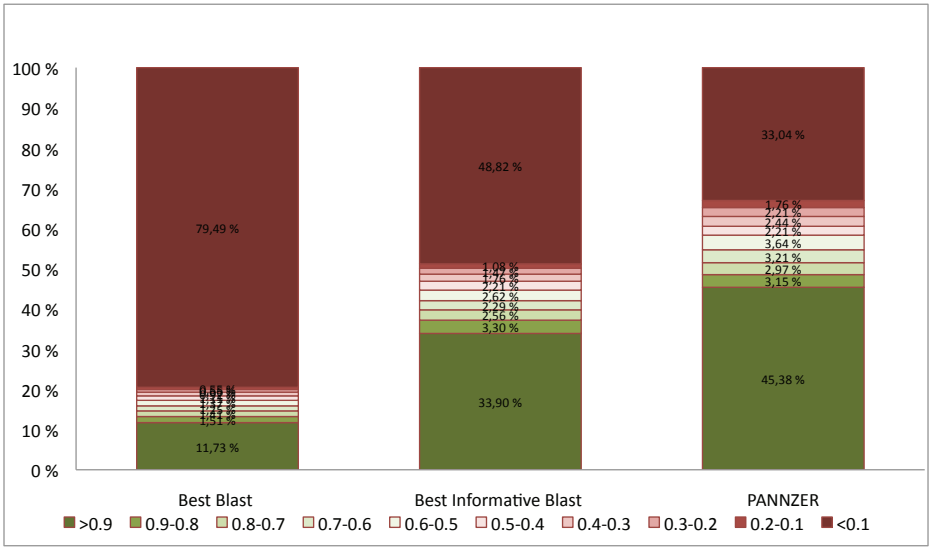

Figure 13. *Prediction results with the eukaryotic dataset. Hits with DSM score above 0.7 are considered to be correct and hits below 0.3 incorrect.*

4.2.2 Gene ontology prediction and CAFA 2011 Challenge

In Critical Assessment of Functional Annotation (CAFA) challenge 2011 54 GO prediction methods were evaluated by using 'blind testing' with an evaluation set of 866 proteins from 11 organisms. The best results were obtained with the Jones-UCL method, second best with Argot2 and the third best with PANNZER (Figure 14, Publications II & III).

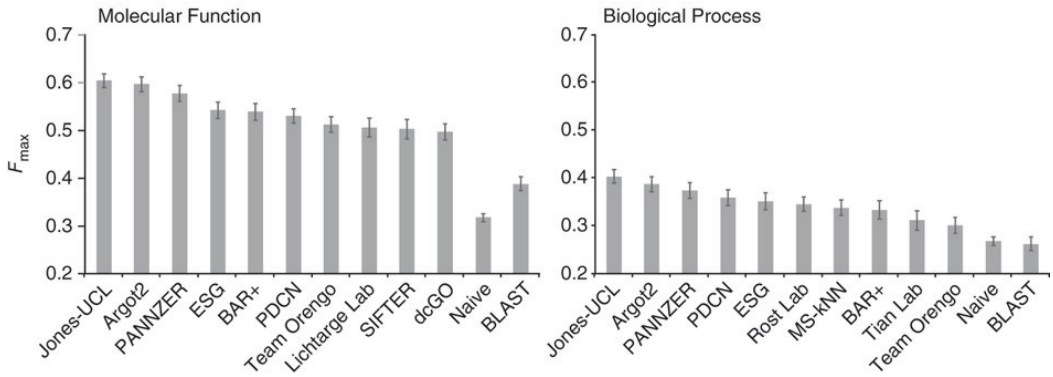

Figure 14. *Results from CAFA 2011 challenge (The figure is taken from Publication III).*

The differences between top-3 methods were small. It is noteworthy that all the methods in the top 10 perform significantly better than traditional BBH (BLAST). The downside is that even with the best method prediction accuracy is still quite weak, especially when predicting Biological Processes. Since most of the methods participated in CAFA 2011 challenge relied on

sequence similarity, the question arises whether Biological Process can be predicted solely from sequence similarity.

Since the PANNZER version that participated in the CAFA 2011 challenge was a pre-release version, we were able to improve prodiction accuracy even further with subsequent release versions. The precision-recall curve in Figures 15 & 16 shows that the current version (New full & New no-neg models) of the regression model is significantly better when predicting Molecular Function and Biological Process GO classes than the model PANNZER used in CAFA 2011 challenge (CAFA model). The difference between the full and no-neg models is that in the no-neg regression model all the variables with negative weight were excluded. The release version of the PANNZER uses the full model by default. In the ongoing CAFA 2014 challenge the Cellular Component GO category is included and the new regression model outperforms the old model also in this category (Publication II).

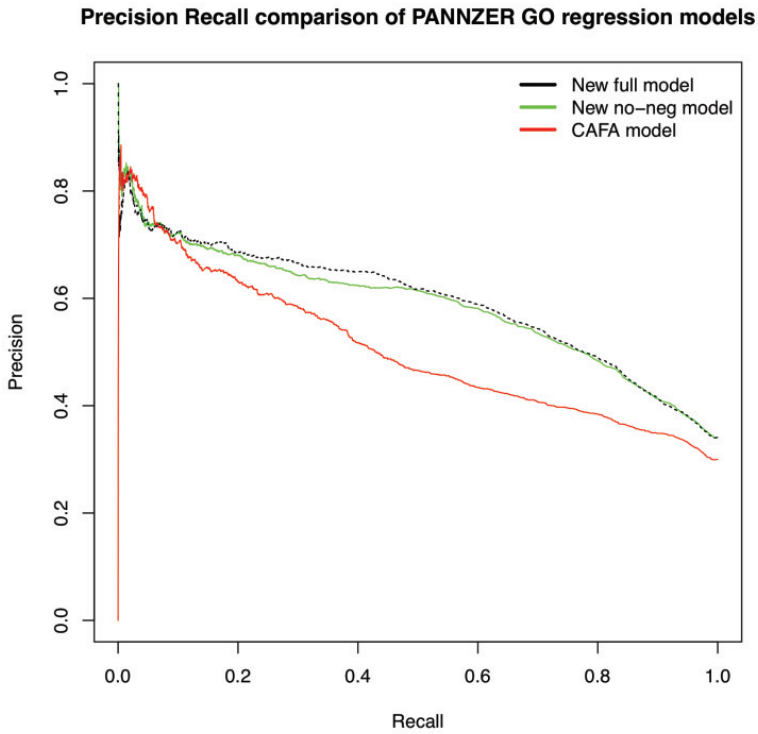

Figure 15. *Improvement of the Molcular Function GO prediction regression model between the pre-release PANNZER version (CAFA model) and the current version (New full model and New no-neg model). The figure is taken from Publication II.*

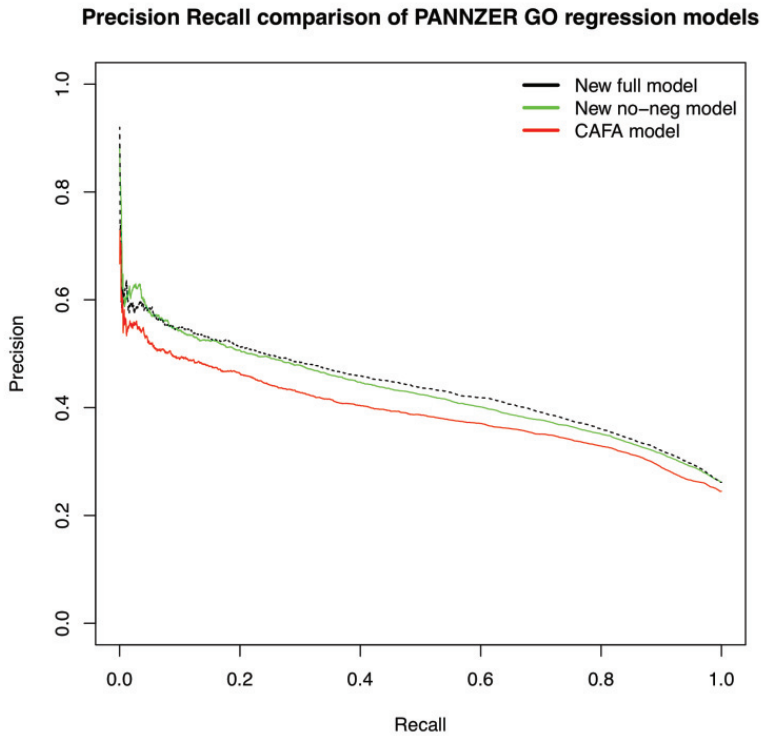

Figure 16. *Improvement of Biological Process GO prediction regression model between pre-release PANNZER version (CAFA model) and current version (New full model and New no-neg model). Figure is taken from Publication II.*

As results indicate, there is plenty of room for improvements and hopefully the ongoing CAFA 2014 will introduce better methods for functional annotation.

4.3 Practical applications in genome analysis

The SANS algorithm has proved to have multiple different usages. So far the SANS program has been used to e.g. find orthologous groups (Unpublished; Nykyri et al. 2012) and the mapping of long reads sequenced with Pacific Biociences PacBio RS II (Unpublished).

4.3.1 OrthoSANS

The SANS algorithm for orthologous groups (OrthoSANS) works as follows: First, the complete proteomes of selected species are aligned using the SANS tool against every proteomes in the set. One-to-one orthologues of proteins were determined using the reciprocal best hit (RBH) criterion. Protein P.1 from proteome 1 and protein P.2 from proteome 2 are reciprocal best hits, if their P.2 is the best match of P.1 in proteome 2 and P.1 is the best match of P.2 in proteome 1. If intraspecies hits are found e.g. with protein P.1 from proteome 1 that are closer to P.1 than P.2, the found hits are considered to be paralogues to a query sequence. Orthologues and paralogues are then converted into an orthologues vs. the species (OvsS) matrix where columns represents species and rows orthologueous groups. The OvsS matrix can be sorted by ordering

similar columns and rows next to each other. Ordering in OrthoSANS program is done using a hierarchical cluster tree from the rows and columns. The similarities of the columns are based on the Pearson correlation, while the similarities of the rows are based on the cosine similarity. The resulting matrix then presents species in phylogenetic order by proteome similarities. Figure 17A shows an example of ordered OvsS matrix visualized as a heatmap. Figure 17B represents Pearson's correlations between proteomes as a heatmap.

OrthoSANS works well if the species are closely related, meaning that protein sequences are >50% identical (See: 4.1 SANS results).

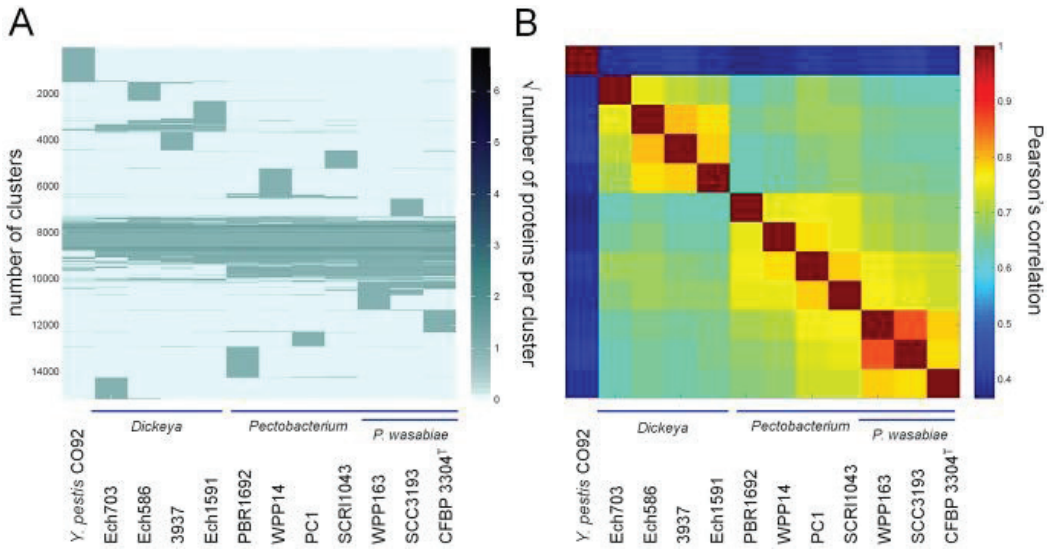

Figure 17. *(A) OvsS matrix visualized as a heatmap. The core genome is visualized in the middle of the figure, and species and strain-specific protein clusters can be found above and below the core. (B) The Pearson's correlations between proteomes are calculated and visualized to indicate their phylogenetic relationships. (Nykyri, 2012)*

4.3.2 SANSMapper

The SANS algorithm has proven also to work well as a mapper of the PacBio sequenced CLR long reads. The SANSmapper was evaluated against popular read aligning tools BWA-MEM (Li, 2013), BWA-SW (Li, 2010) and BLASR (Chaisson, 2012). Evaluation was done by simulating 5000bp long CLR reads. The simulated CLR reads were extracted from the *A. thaliana* genome and the location where the read was originally extracted was saved. Then different amounts of errors that are characteristic of PacBio RS II CLRs were added to the reads. In final stage the reads were mapped back to *A. thaliana* genome with four different mappers. Since the original loci of the reads were saved, we were able to monitor the accuracy of the mappers. A read was counted as correct if the mapped location overlapped with the real location. As the results in Table 10 shows, the SANSmapper outperformed other mappers in accuracy when error level rose above 15%, which is quite a characteristic error rate for PacBio RS II CLR (Au, 2012).

Table 10. *Simulated 5000 bp x 100 000 reads from A. thaliana. Shows the percentage of reads that each mapper could map correctly with different PacBio type error percentages and what percentages were left unmapped.*

Dataset Error-%	SANSmapper Correct% (Unmapped%)	BWA-MEM Correct% (Unmapped%)	BWA-SW Correct% (Unmapped%)	BLASR Correct% (Unmapped%)
0	96.89 (0.04)	**98.52 (0.04)**	97.9 (0.04)	35.03 (0.05)
5	96.43 (0.04)	**98.44 (0.04)**	93.41 (0.04)	39.67 (0.04)
10	95.54 (0.04)	**96.48 (0.04)**	93.21 (0.08)	46.39 (0.04)
15	**94.59 (0.04)**	92.46 (0.04)	67.38 (26.37)	53.13 (0.04)
20	**93.39 (0.04)**	91.89 (0.04)	18.28 (79.82)	59.11 (0.04)
25	**92.25 (0.04)**	91.33 (0.05)	3.37 (96.23)	64.56 (0.04)
30	**91.06 (0.16)**	85.05 (5.62)	0.53 (99.41)	71.60 (0.04)

5 Conclusions and perspectives

Since sequence databases are growing with increasing speed, sequence retrieval algorithms also need to evolve to meet the demands of the fast usage of databases. Due to high cost of computational resources, some of the public services have had to close down their options for free BLAST services. For example the Community Cyberinfrastructure for Advanced Microbial Ecology Research and Analysis (CAMERA) suite noted that starting from 1st January 2014, the usage of BLAST algorithm is restricted in their services because of high computational costs (http://camera.calit2.net/transition_notification.shtm). Also PairsDB service (Heger et al. 2008) had to close their services due to the high computational expense of BLAST.

The SANS algorithm introduced in Publication I meets most of the requirements needed in faster sequence retrieval from large sequence databases. In this thesis and in Publication I we have demonstrated that the SANS algorithm is 10 – 100 times faster and when sequence identity is above 50%, as sensitive as other commonly used tools. The SANS algorithm is computationally less expensive than competing methods and therefore provides a more cost efficient solution for sequence retrieval.

Sequence databases don't solely grow in size, but also in the different functional annotations they contains. Large number of these annotations are incorrect. Traditional methods only propagate errors to newly introduced sequences if no error detecting statistics are used. In Publication II we present an annotation tool that uses novel statistics in testing the reliability of a function to be transferred. Publication III shows that this kind of approach increases annotation accuracy significantly over the first generation methods that are commonly used (e.g. best BLAST hit).

As the results in Publication III indicate, the overall prediction accuracy in functional annotation tools today is still quite unsatisfactory. Especially prediction of Biological Process GO classes seems to be a hard task for new generation tools. Since most of the top 10 algorithms in the CAFA 2011 challenge are based on sequence similarity, a valid question might be: is Biological Process Gene Ontology predictable solely from sequence homology? Maybe smaller functional elements such as domains or motifs (e.g. signal peptides) could be a better indication of Biological Process. Biological Process seems to be linked to subcellular localization, so maybe Biological Process is something that can be predicted via both Molecular Function and Cellular Component.

The PANNZER method (Publication II) is now on the proof-of-concept level and is in future going to be implemented by using even faster methodology. At the moment PANNZER uses BLAST as a default sequence retrieval tool, but SANS (Publication I) would be a better choice because of its fast speed and sufficient sensitivity. Other modifications need to be made to PANNZER if response time enabling interactive usage experience is desired. The PANNZER tool is quite fast as it is, but changing the currently used MySQL database to a faster NoSQL database (e.g. MongoDB) would decrease query times of the database. Also the problem is natively parallelized and therefore parallel programming would speed up the program significantly.

There are also many ways to increase the prediction accuracy of the PANNZER tool. Incorporation of domains and motifs in function prediction could enable significant improvement in the separation of irrelevant sequences from the relevant. Also, the use of semantic similarities and natural language processing methods in description analysis would make it possible to cluster together cohesive sequences with different nomenclature or euphemisms used in their descriptions.

The high error rate in public databases indicates that better methods in functional annotation are needed. The results shown in Publication III confirm that even the state-of-the-art methods today perform quite poorly in function annotation. This means that eventually errors propagate throughout the databases and corrupt the information they contain. If corrective measures are not undertaken, at some point the electronically annotated sequences would be so corrupted that their use would no longer be recommended. In future 99.4% (amount of electronic annotations) of the functional annotations in databases could be excluded from further use.

There is a long way to travel until satisfactory results in prediction accuracy are achieved. Better algorithms and annual or biennial independent testing is needed and therefore the CAFA challenge has a part to play in this development.

Acknowledgements

This thesis was carried out in the Department of Biosciences at the University of Helsinki, during the years 2008-2014. The funding for this study has been provided by the University of Helsinki and the Institute of Biotechnology. Over the time I have received several travel grants from the University of Helsinki and the Finnish Doctoral Programme in Computational Sciences (FICS). Thanks to these grants I have been able to attend international conferences and get valuable help to boost my academic career.

Many people have contributed to this work by providing their expertise when I have been stuck. First I want to thank my supervisor prof. Liisa Holm for taking me under your wing. You have provided me invaluable help and guidance throughout my whole PhD project. The reviewers of this thesis, Professor Tapio Salakoski and Professor Erik Sonnhammer, are gratefully acknowledged for their expertise and constructive comments on the manuscript.

I would like to thank all my many co-authors and collaborators. I owe the most to Dr. Petri Törönen and Jussi Nokso-Koivisto for carrying through the PANNZER project. It really was a pleasure to work with you both during these years and having all those many inspiring brainstorming sessions. Thank you guys, I still have many ideas for the future in my backpocket. To Matti Kankainen I owe huge thanks for being my friend and introducing me the wonderful world of functional annotation. Dr. Hung Ta I would like to thank for your friendship and those tough badminton games. I hope that we continue our fruitful collaboration in future also.

I would also like to thank my other co-authors Outi Niemi and Dr. Johanna Nykyri for brilliant collaboration in *Pectobacterium* project. Prof. Tapio Palva and Prof. Minna Pirhonen for taking me in your inspiring projects. Big thanks to Prof. Ilkka Hanski for having me in the Glanville Fritillary genome project and to my all colleagues in Metapopulation Research Group, especially Panu Somervuo, Virpi Ahola, Pasi Rastas, Leena Salmela and Rainer Lehtonen. I really enjoyed my time in MRG. Thanks for the mug!

My gratitude also goes to Dr. Päivi Onkamo for guiding me in the rambling world of teaching. Without the motivational buttkicking from Dr. Janne Ravantti this thesis would not be done today. Huge thanks to you both!

I would like to thank also all my current co-workers in the DNA Sequencing and Genomics Lab and the Saimaa Ringed Seal Genome Project. Especially Dr. Petri Auvinen and Prof. Jukka Jernvall for believing in me and trusting the data analysis in my hands. Also Lars Paulin, Olli-Pekka Smolander, Juhana Kammonen, Olli Lounela, Kaisa Koskinen, Teija Ojala, Pedro Pereira, Velma Aho, Anne Ylinen, Margarita Andreevskaya and the rest of the crew for your warm welcome to the group.

I would like to thank my family for extremely valuable support I have got from you. Without the support from my parents Pentti and Irene Koskinen this would not have been possible. Thanks for my sister Jonna Koskinen for being in my life. Last but definitely not least I would like to thank the most important person, my dear wife Sanna Jormakka for keeping me sane during all these years. You are the guiding light in my life. I can't thank you enough. I love you!

Helsinki, June 2014

Patrik Koskinen

References

Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., et al. (1997). Gapped BLAST and PSI-BLAST: A new generation of protein database search programs *Nucleic Acids Research, 25*(17), 3389-3402.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., et al. (2000). Gene ontology: Tool for the unification of biology. the gene ontology consortium *Nature Genetics, 25*(1), 25-29. doi:10.1038/75556

Au, K. F., Underwood, J. G., Lee, L., & Wong, W. H. (2012). Improving PacBio long read accuracy by short read alignment *PloS One, 7*(10), e46679. doi:10.1371/journal.pone.0046679; 10.1371/journal.pone.0046679

Aziz, R. K., Bartels, D., Best, A. A., DeJongh, M., Disz, T., Edwards, R. A., et al. (2008). The RAST server: Rapid annotations using subsystems technology *BMC Genomics, 9*, 75-2164-9-75. doi:10.1186/1471-2164-9-75; 10.1186/1471-2164-9-75

Bairoch, A. (2000). The ENZYME database in 2000 *Nucleic Acids Research, 28*(1), 304-305.

Barrett, A. J. (1995). Nomenclature committee of the international union of biochemistry and molecular biology (NC-IUBMB). enzyme nomenclature. recommendations 1992. supplement 2: Corrections and additions (1994) *European Journal of Biochemistry / FEBS, 232*(1), 1-6.

Bellman, R. (1954). Dynamic programming and a new formalism in the calculus of variations *Proceedings of the National Academy of Sciences of the United States of America, 40*(4), 231-235.

Bluthgen, N., Brand, K., Cajavec, B., Swat, M., Herzel, H., & Beule, D. (2005). Biological profiling of gene groups utilizing gene ontology *Genome Informatics.International Conference on Genome Informatics, 16*(1), 106-115.

Bork, P. (2000). Powers and pitfalls in sequence analysis: The 70% hurdle *Genome Research, 10*(4), 398-400.

Brenner, S. E. (1999). Errors in genome annotation *Trends in Genetics : TIG, 15*(4), 132-133.

Buljan, M., & Bateman, A. (2009). The evolution of protein domain families *Biochemical Society Transactions, 37*(Pt 4), 751-755. doi:10.1042/BST0370751; 10.1042/BST0370751

Chaisson, M. J., & Tesler, G. (2012). Mapping single molecule sequencing reads using basic local alignment with successive refinement (BLASR): Application and theory *BMC Bioinformatics, 13*, 238-2105-13-238. doi:10.1186/1471-2105-13-238; 10.1186/1471-2105-13-238

Clark, W. T., & Radivojac, P. (2011). Analysis of protein function and its prediction from amino acid sequence *Proteins, 79*(7), 2086-2096. doi:10.1002/prot.23029; 10.1002/prot.23029

Conesa, A., Gotz, S., Garcia-Gomez, J. M., Terol, J., Talon, M., & Robles, M. (2005). Blast2GO: A universal tool for annotation, visualization and analysis in functional genomics research *Bioinformatics (Oxford, England), 21*(18), 3674-3676. doi:10.1093/bioinformatics/bti610

Costello, J. C., Dalkilic, M. M., Beason, S. M., Gehlhausen, J. R., Patwardhan, R., Middha, S., et al. (2009). Gene networks in drosophila melanogaster: Integrating experimental data to predict gene function *Genome Biology, 10*(9), R97-2009-10-9-r97. Epub 2009 Sep 16. doi:10.1186/gb-2009-10-9-r97; 10.1186/gb-2009-10-9-r97

Cozzetto, D., Buchan, D. W., Bryson, K., & Jones, D. T. (2013). Protein function prediction by massive integration of evolutionary analyses and multiple data sources *BMC Bioinformatics, 14 Suppl 3*, S1-2105-14-S3-S1. Epub 2013 Feb 28. doi:10.1186/1471-2105-14-S3-S1; 10.1186/1471-2105-14-S3-S1

Deng, M., Zhang, K., Mehta, S., Chen, T., & Sun, F. (2003). Prediction of protein function using protein-protein interaction data *Journal of Computational Biology : A Journal of Computational Molecular Cell Biology, 10*(6), 947-960. doi:10.1089/106652703322756168

Devos, D., & Valencia, A. (2001). Intrinsic errors in genome annotation *Trends in Genetics : TIG, 17*(8), 429-431.

Dimmer, E. C., Huntley, R. P., Alam-Faruque, Y., Sawford, T., O'Donovan, C., Martin, M. J., et al. (2012). The UniProt-GO annotation database in 2011 *Nucleic Acids Research, 40*(Database issue), D565-70. doi:10.1093/nar/gkr1048; 10.1093/nar/gkr1048

Edgar, R. C. (2010). Search and clustering orders of magnitude faster than BLAST *Bioinformatics (Oxford, England), 26*(19), 2460-2461. doi:10.1093/bioinformatics/btq461; 10.1093/bioinformatics/btq461

Edgar, R. C. (2010; 2010). Search and clustering orders of magnitude faster than BLAST *Bioinformatics, 26*(19), 2460 <last_page> 2461. doi:10.1093/bioinformatics/btq461

Enault, F., Suhre, K., & Claverie, J. M. (2005). Phydbac "gene function predictor": A gene annotation tool based on genomic context analysis *BMC Bioinformatics, 6*, 247. doi:10.1186/1471-2105-6-247

Engelhardt, B. E., Jordan, M. I., Muratore, K. E., & Brenner, S. E. (2005). Protein molecular function prediction by bayesian phylogenomics *PLoS Computational Biology, 1*(5), e45. doi:10.1371/journal.pcbi.0010045

Falda, M., Toppo, S., Pescarolo, A., Lavezzo, E., Di Camillo, B., Facchinetti, A., et al. (2012). Argot2: A large scale function prediction tool relying on semantic similarity of weighted gene ontology terms *BMC Bioinformatics, 13 Suppl 4*, S14-2105-13-S4-S14. doi:10.1186/1471-2105-13-S4-S14; 10.1186/1471-2105-13-S4-S14

Forslund, K., Pekkari, I., & Sonnhammer, E. L. (2011). Domain architecture conservation in orthologues *BMC Bioinformatics, 12*, 326-2105-12-326. doi:10.1186/1471-2105-12-326; 10.1186/1471-2105-12-326

Friedberg, I. (2006). Automated protein function prediction--the genomic challenge *Briefings in Bioinformatics, 7*(3), 225-242. doi:10.1093/bib/bbl004

Gallwitz, D., Donath, C., & Sander, C. (1983). A yeast gene encoding a protein homologous to the human c-has/bas proto-oncogene product *Nature, 306*(5944), 704-707.

Gaudet, P., Livstone, M. S., Lewis, S. E., & Thomas, P. D. (2011). Phylogenetic-based propagation of functional annotations within the gene ontology consortium *Briefings in Bioinformatics, 12*(5), 449-462. doi:10.1093/bib/bbr042; 10.1093/bib/bbr042

Gilks, W. R., Audit, B., De Angelis, D., Tsoka, S., & Ouzounis, C. A. (2002). Modeling the percolation of annotation errors in a database of protein sequences *Bioinformatics (Oxford, England), 18*(12), 1641-1649.

Gilks, W. R., Audit, B., de Angelis, D., Tsoka, S., & Ouzounis, C. A. (2005). Percolation of annotation errors through hierarchically structured protein sequence databases *Mathematical Biosciences, 193*(2), 223-234. doi:10.1016/j.mbs.2004.08.001

Gogarten, J. P., & Olendzenski, L. (1999). Orthologues, paralogues and genome comparisons *Current Opinion in Genetics & Development, 9*(6), 630-636.

Hawkins, T., Luban, S., & Kihara, D. (2006). Enhanced automated function prediction using distantly related sequences and contextual association by PFP *Protein Science : A Publication of the Protein Society, 15*(6), 1550-1556. doi:10.1110/ps.062153506

Heger, A., Korpelainen, E., Hupponen, T., Mattila, K., Ollikainen, V., & Holm, L. (2008). PairsDB atlas of protein sequence space. *Nucleic acids research*, *36*, D276-D280.

Huttenhower, C., Hibbs, M., Myers, C., & Troyanskaya, O. G. (2006). A scalable method for integration and functional analysis of multiple microarray datasets *Bioinformatics (Oxford, England), 22*(23), 2890-2897. doi:10.1093/bioinformatics/btl492

Jeffery, C. J. (2003). Moonlighting proteins: Old proteins learning new tricks *Trends in Genetics, 19*(8), 415 417. doi:10.1016/S0168-9525(03)00167-7

Jeffery, C. J. (2009). Moonlighting proteins—an update *Molecular BioSystems, 5*(4), 345. doi:10.1039/b900658n

Jensen, L. J., Gupta, R., Blom, N., Devos, D., Tamames, J., Kesmir, C., et al. (2002). Prediction of human protein function from post-translational modifications and localization features *Journal of Molecular Biology, 319*(5), 1257-1265. doi:10.1016/S0022-2836(02)00379-0

Jones, C. E., Brown, A. L., & Baumann, U. (2007). Estimating the annotation error rate of curated GO database sequence annotations *BMC Bioinformatics, 8*, 170. doi:10.1186/1471-2105-8-170

Kankainen, M., Ojala, T., & Holm, L. (2012). BLANNOTATOR: Enhanced homology-based function prediction of bacterial proteins *BMC Bioinformatics, 13*, 33-2105-13-33. doi:10.1186/1471-2105-13-33; 10.1186/1471-2105-13-33

Kent, W. J. (2002). BLAT--the BLAST-like alignment tool *Genome Research, 12*(4), 656-664. doi:10.1101/gr.229202. Article published online before March 2002

Koonin, E. V. (2005). Orthologues, paralogues, and evolutionary genomics *Annual Review of Genetics, 39*, 309-338. doi:10.1146/annurev.genet.39.073003.114725

Koski, L. B., & Golding, G. B. (2001). The closest BLAST hit is often not the nearest neighbor *Journal of Molecular Evolution, 52*(6), 540-542. doi:10.1007/s002390010184

Kourmpetis, Y. A., van Dijk, A. D., Bink, M. C., van Ham, R. C., & ter Braak, C. J. (2010). Bayesian markov random field analysis for protein function prediction based on network data *PloS One, 5*(2), e9293. doi:10.1371/journal.pone.0009293; 10.1371/journal.pone.0009293

Laskowski, R. A., Watson, J. D., & Thornton, J. M. (2005). Protein function prediction using local 3D templates *Journal of Molecular Biology, 351*(3), 614-626. doi:10.1016/j.jmb.2005.05.067

Lee, D., Redfern, O., & Orengo, C. (2007). Predicting protein function from sequence and structure *Nature Reviews.Molecular Cell Biology, 8*(12), 995-1005. doi:10.1038/nrm2281

Lee, I., Date, S. V., Adai, A. T., & Marcotte, E. M. (2004). A probabilistic functional network of yeast genes *Science (New York, N.Y.), 306*(5701), 1555-1558. doi:10.1126/science.1099511

Letovsky, S., & Kasif, S. (2003). Predicting protein function from protein/protein interaction data: A probabilistic approach *Bioinformatics (Oxford, England), 19 Suppl 1*, i197-204.

Li, H. *Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM.* Retrieved 12/9/2013, 2013, from http://arxiv.org/abs/1303.3997

Li, H., & Durbin, R. (2010). Fast and accurate long-read alignment with burrows-wheeler transform *Bioinformatics (Oxford, England), 26*(5), 589-595. doi:10.1093/bioinformatics/btp698; 10.1093/bioinformatics/btp698

Liberal, R., & Pinney, J. W. (2013). Simple topological properties predict functional misannotations in a metabolic network *Bioinformatics (Oxford, England), 29*(13), i154-61. doi:10.1093/bioinformatics/btt236; 10.1093/bioinformatics/btt236

Lobley, A. E., Nugent, T., Orengo, C. A., & Jones, D. T. (2008). FFPred: An integrated feature-based function prediction server for vertebrate proteomes *Nucleic Acids Research, 36*(Web Server), W297 <last_page> W302. doi:10.1093/nar/gkn193

Marcotte, E. M., Pellegrini, M., Ng, H. L., Rice, D. W., Yeates, T. O., & Eisenberg, D. (1999). Detecting protein function and protein-protein interactions from genome sequences *Science (New York, N.Y.), 285*(5428), 751-753.

Martin, D. M., Berriman, M., & Barton, G. J. (2004). GOtcha: A new method for prediction of protein function assessed by the annotation of seven genomes *BMC Bioinformatics, 5*, 178. doi:10.1186/1471-2105-5-178

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive bayes text classification. Paper presented at the *AAAI-98 Workshop on Learning for Text Categorization, , 752*. pp. 41-48.

Mulder, N. J., Apweiler, R., Attwood, T. K., Bairoch, A., Bateman, A., Binns, D., et al. (2007). New developments in the InterPro database *Nucleic Acids Research, 35*(Database issue), D224-8. doi:10.1093/nar/gkl841

Muller, J., Szklarczyk, D., Julien, P., Letunic, I., Roth, A., Kuhn, M., et al. (2010). eggNOG v2.0: Extending the evolutionary genealogy of genes with enhanced non-supervised orthologueous groups, species and functional annotations *Nucleic Acids Research, 38*(Database issue), D190-5. doi:10.1093/nar/gkp951; 10.1093/nar/gkp951

Nabieva, E., Jim, K., Agarwal, A., Chazelle, B., & Singh, M. (2005). Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps *Bioinformatics (Oxford, England), 21 Suppl 1*, i302-10. doi:10.1093/bioinformatics/bti1054

Nadzirin, N., & Firdaus-Raih, M. (2012). Proteins of unknown function in the protein data bank (PDB): An inventory of true uncharacterized proteins and computational tools for their analysis *International Journal of Molecular Sciences, 13*(10), 12761-12772. doi:10.3390/ijms131012761; 10.3390/ijms131012761

Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins *Journal of Molecular Biology, 48*(3), 443-453.

Nehrt, N. L., Clark, W. T., Radivojac, P., & Hahn, M. W. (2011). Testing the orthologue conjecture with comparative functional genomic data from mammals *PLoS Computational Biology, 7*(6), e1002073. doi:10.1371/journal.pcbi.1002073; 10.1371/journal.pcbi.1002073

Nykyri, J., Niemi, O., Koskinen, P., Nokso-Koivisto, J., Pasanen, M., Broberg, M., et al. (2012). Revised phylogeny and novel horizontally acquired virulence determinants of the model soft rot phytopathogen pectobacterium wasabiae SCC3193 *PLoS Pathogens, 8*(11), e1003013. doi:10.1371/journal.ppat.1003013; 10.1371/journal.ppat.1003013

Pal, D., & Eisenberg, D. (2005). Inference of protein function from protein structure *Structure (London, England : 1993), 13*(1), 121-130. doi:10.1016/j.str.2004.10.015

Pazos, F., & Sternberg, M. J. (2004). Automated prediction of protein function and detection of functional sites from structure *Proceedings of the National Academy of Sciences of the United States of America, 101*(41), 14754-14759. doi:10.1073/pnas.0404569101

Pehkonen, P., Wong, G., & Toronen, P. (2005). Theme discovery from gene lists for identification and viewing of multiple functional groups *BMC Bioinformatics, 6*, 162. doi:10.1186/1471-2105-6-162

Pellegrini, M., Marcotte, E. M., Thompson, M. J., Eisenberg, D., & Yeates, T. O. (1999). Assigning protein functions by comparative genome analysis: Protein phylogenetic profiles *Proceedings of the National Academy of Sciences of the United States of America, 96*(8), 4285-4288.

Percudani, R., Carnevali, D., & Puggioni, V. (2013). Ureidoglycolate hydrolase, amidohydrolase, lyase: How errors in biological databases are incorporated in scientific papers and vice versa *Database : The Journal of Biological Databases and Curation, 2013*, bat071. doi:10.1093/database/bat071; 10.1093/database/bat071

Quevillon, E., Silventoinen, V., Pillai, S., Harte, N., Mulder, N., Apweiler, R., et al. (2005). InterProScan: Protein domains identifier *Nucleic Acids Research, 33*(Web Server issue), W116-20. doi:10.1093/nar/gki442

Radivojac, P., Clark, W. T., Oron, T. R., Schnoes, A. M., Wittkop, T., Sokolov, A., et al. (2013). A large-scale evaluation of computational protein function prediction *Nature Methods, 10*(3), 221-227. doi:10.1038/nmeth.2340; 10.1038/nmeth.2340

Rentzsch, R., & Orengo, C. A. (2009). Protein function prediction--the power of multiplicity *Trends in Biotechnology, 27*(4), 210-219. doi:10.1016/j.tibtech.2009.01.002; 10.1016/j.tibtech.2009.01.002

Rost, B. (2002). Enzyme function less conserved than anticipated *Journal of Molecular Biology, 318*(2), 595-608. doi:10.1016/S0022-2836(02)00016-5

Ruepp, A., Zollner, A., Maier, D., Albermann, K., Hani, J., Mokrejs, M., et al. (2004). The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes *Nucleic Acids Research, 32*(18), 5539-5545. doi:10.1093/nar/gkh894

Schnoes, A. M., Brown, S. D., Dodevski, I., & Babbitt, P. C. (2009). Annotation error in public databases: Misannotation of molecular function in enzyme superfamilies *PLoS Computational Biology, 5*(12), e1000605. doi:10.1371/journal.pcbi.1000605; 10.1371/journal.pcbi.1000605

Smith, T. F., & Waterman, M. S. (1981). Identification of common molecular subsequences *Journal of Molecular Biology, 147*(1), 195-197.

Sokolov, A., & Ben-Hur, A. (2010). Hierarchical classification of gene ontology terms using the GOstruct method *Journal of Bioinformatics and Computational Biology, 8*(2), 357-376.

Studer, R. A., & Robinson-Rechavi, M. (2009). How confident can we be that orthologues are similar, but paralogues differ? *Trends in Genetics : TIG, 25*(5), 210-216. doi:10.1016/j.tig.2009.03.004; 10.1016/j.tig.2009.03.004

Suzek, B. E., Huang, H., McGarvey, P., Mazumder, R., & Wu, C. H. (2007). UniRef: Comprehensive and non-redundant UniProt reference clusters *Bioinformatics (Oxford, England), 23*(10), 1282-1288. doi:10.1093/bioinformatics/btm098

Ta, H. X., Koskinen, P., & Holm, L. (2011). A novel method for assigning functional linkages to proteins using enhanced phylogenetic trees *Bioinformatics (Oxford, England), 27*(5), 700-706. doi:10.1093/bioinformatics/btq705; 10.1093/bioinformatics/btq705

Toronen, P., Ojala, P. J., Marttinen, P., & Holm, L. (2009). Robust extraction of functional signals from gene set analysis using a generalized threshold free scoring function *BMC Bioinformatics, 10*, 307-2105-10-307. doi:10.1186/1471-2105-10-307; 10.1186/1471-2105-10-307

Troyanskaya, O. G., Dolinski, K., Owen, A. B., Altman, R. B., & Botstein, D. (2003). A bayesian framework for combining heterogeneous data sources for gene function prediction (in saccharomyces cerevisiae) *Proceedings of the National Academy of Sciences of the United States of America, 100*(14), 8348-8353. doi:10.1073/pnas.0832373100

Vazquez, A., Flammini, A., Maritan, A., & Vespignani, A. (2003). Global protein function prediction from protein-protein interaction networks *Nature Biotechnology, 21*(6), 697-700. doi:10.1038/nbt825

Wass, M. N., & Sternberg, M. J. (2008). ConFunc--functional annotation in the twilight zone *Bioinformatics (Oxford, England), 24*(6), 798-806. doi:10.1093/bioinformatics/btn037; 10.1093/bioinformatics/btn037