BIOINFORMATICS

Vol. 00 no. 00 2005
Pages 1–7

An extensible automated protein annotation tool: standardizing input and output using validated XML

S. Vishnu V. Deevi[a] and Andrew C. R. Martin[b]*

[a]School of Animal and Microbial Sciences, The University of Reading, P.O.Box 228, Whiteknights, Reading RG6 6AJ. [b]Department of Biochemistry and Molecular Biology, University College London, Gower Street, London WC1E 6BT

ABSTRACT

Motivation: There is a frequent need to apply a large range local or remote prediction and annotation tools to one or more sequences. We have created a tool able to dispatch one or more sequences to assorted services by defining a consistent XML format for data and annotations.

Results: By analyzing annotation tools, we have determined that annotations can be described using one or more of 6 forms of data: numeric or textual annotation of residues, domains (residue ranges) or whole sequences. With this in mind, XML DTDs have been designed to store the input and output of any server. Plug-in wrappers to a number of services have been written which are called from a master script. The resulting APATML is then formatted for display in HTML. Alternatively further tools may be written to perform post-analysis.

Availability: The 'Automated Protein Annotation Tool' (APAT) can be downloaded from `http://www.bioinf.org.uk/apat/` As well as source code and documentation, a demonstration web server is provided which uses APAT to perform a number of annotations of a sequence.

Contact: andrew@bioinf.org.uk –or– martin@biochem.ucl.ac.uk

1 INTRODUCTION

In the analysis of sequence data, whether from genomics, transcriptomics, proteomics, or a more specific interest in a small set of proteins from a single pathway or targeted to an organelle, there is a frequent need to apply a wide range of prediction and annotation tools to one or more sequences. Using numerous web-based or local tools, and collating and comparing their outputs is a laborious and error-prone task.

Given a protein sequence, one generally starts by looking in SwissProt (Bairoch and Apweiler, 2000) to see whether the sequence has already been annotated by expert hand-curation (Boeckmann *et al.*, 2003). Failing that, a close homologue may be available from which annotations can be transferred. However, if the sequence (or a close homologue) is not present in SwissProt, or the specific type of required annotation is not included, then one may need to run a selection of prediction tools, either across the web, or locally.

A number of web-based tools exist which provide integrated annotation/prediction systems. However, all of these systems suffer restrictions of one form or another. Very few are extensible: many provide pre-calculated annotations (frequently at the genome or complete proteome level), or provide only a fixed set of tools that can be run on a protein sequence. Where a sequence can be submitted for predictions to be made, it is rare that more than one sequence can be submitted at a time, especially where more than one type of annotation is provided.

There are many examples of pre-calculated annotations. For example, ENSEMBL (Hubbard *et al.*, 2002), the eukaryotic genome database project, provides annotations of genome data including limited annotation of the translated proteins. DAS (`http://www.biodas.org/`) is a distributed annotation system that allows pre-calculated annotation of genomes (including of the encoded proteins) to be decentralized among multiple third-party annotators and integrated by client-side software (Dowell *et al.*, 2001). ENSEMBL also provides annotations served via DAS. In principle, this allows anyone to add their own annotations, but this requires pre-calculation to be performed in-house — there is no support for on-the-fly annotations. PEDANT (Protein Extraction, Description and ANalysis Tool) (Frishman *et al.*, 2001) also provides pre-calculated annotations for a wide range of complete and incomplete genomes with integration of both functional and structural information. Another server, Integr8 (`http://www.ebi.ac.uk/integr8/`) assigns annotations from various sources including InterPro (Apweiler *et al.*, 2001) and Gene Ontology (GO) (Ashburner *et al.*, 2000) terms to gene products in completed genomes and proteomes.

Similarly there are tools that work at the protein level. For example PDBSUM (Laskowski, 2001) is a pre-calculated set of annotations of structures from the Protein Data Bank; GRASS (Nayal *et al.*, 1999) provides graphical representation and analysis of structures; SAS (Milburn *et al.*, 1998) and STING-M (Neshich *et al.*, 2003) are web-based tools for integrating structural information with sequence analysis and alignment. The 'Predict Protein Server' (Rost, 1996) can take a protein sequence and perform predictions using a set of tools, but this toolset cannot be extended and only one sequence can be processed at a time.

In addition there are large numbers of individual servers which allow a sequence to be submitted over the web and predictions of properties such as secondary structure, post-translational modification sites, solvent accessibility and transmembrane regions. Representative lists are available at `http://www.expasy.org/tools/`

*to whom correspondence should be addressed

and http://www.up.univ-mrs.fr/~wabim/english/logligne.html. A few servers, such as DAS-TM (Cserzo *et al.*, 2004) and NetPhos 2.0 (Blom *et al.*, 1999) allow a batch of sequences to be submitted.

Another aspect of this wide variety of tools is that the results are all presented in different forms. It would be much easier for the biologist wanting to scan the results if different tools provided results in a consistent format. Similarly for the bioinformatician wishing to write code to integrate and analyze the results from a number of different prediction tools, it would be advantageous if the results were available in a consistent form.

A number of proposals have been made for XML formats in which to store sequence data and related annotation information. One example is the DAS XML specification (http://www.biodas.org/). Others include the GAME XML specification implemented by flybase (http://flybase.bio.indiana.edu/annot/) and OmniGene (http://omnigene.sourceforge.net/). While the DASGFF and DASSTYLE elements of the DAS XML specification come close to our requirements, they still do not provide a simple, concise and consistent annotation format that can be used for the output of a large range of protein sequence annotation tools where results may need to be represented as numbers, text and graphs. Another XML specification for annotation of sequences has been used for PathPort/ToolBus (Eckart and Sobral, 2003) which is described below.

One attempt to integrate diverse tools is Taverna (Oinn *et al.*, 2004) which is part of the ^{my}Grid project (http://www.mygrid.org.uk). This system provides a graphical tool for creating and running arbitrarily complex bioinformatics workflows consisting of interlinked processing units each of which transforms a set of input data into a set of output data. Workflows are created in a language called Scufl. Oinn *et al.* (2004) list six types of supported Taverna 'processors'. *1. Arbitrary WSDL types* allow the use of tools provided as Web-services; *2. Soaplab types* allow local tools to be wrapped within a Web-service (Senger *et al.*, 2003) and servers available via web pages may be wrapped using the Gowlab tool of SoapLab. According to the SoapLab web pages (http://www.ebi.ac.uk/soaplab/Gowlab.html), this requires one to *"download Soaplab, install Tomcat and optionally [the] mySQL database [...] create ACD files [...] and potentially to write a few Java plug-in classes"*; *3. Talisman types* allow access to Grid applications developed using the Talisman system for rapid application development (Oinn, 2003); *4. Nested workflow types* allow child Scufl workflows to be invoked; *5. String constant types* allow a constant value to be fed into an established workflow; *5. Local processor types* allow new local functions to be used. These must be coded as classes which comply with a simple Java interface. Oinn *et al.* state that invocation mechanisms other than Web-services require *"first, creating a plug-in for the Freefluo enactor to access the resource and, second, implementing a corresponding Scufl processor type"*. In this way, Taverna provides support for a number of mechanisms, including access to BioMart (Pruess *et al.*, 2005; Durinck *et al.*, 2005), an API consumer (which can cope with a variety of Java APIs) and scripting support via the beanshell.

Taverna is an extremely powerful tool with aims which are much more wide-ranging and complex than supporting the simple desire to scan a sequence against a set of prediction servers. For our purposes, we do not require a true workflow: no data output by one tool becomes the input for another tool. While Taverna could clearly be used in this way, extending the Taverna system to access local and web-based (non Web-service) tools, either using SoapLab, or local processor types are complex procedures and require a considerable investment of time to develop the expertise required. Further, Taverna makes no attempt to enforce a common presentation of predictions for a sequence allowing the scientist to obtain a summary of all the predictions in a common format.

The ToolBus architechture (Eckart and Sobral, 2003), is another system which provides a generic, web-services based framework to deal with issues such as data and tool interoperability. ToolBus is a client-side interconnect, written in Java, which allows access to remote web-services as well local programs and files. This provides data and analysis services, and allows examination of results using a wide variety of visualization tools. In addition, ToolBus enables users to form groupings of related information and to perform comparative analysis using these data groups in order to support the discovery of interesting inter-data relationships. PathPort (Pathogen Portal) is a collection of web-services (including gene prediction and multiple sequence alignment) and visualization tools based around the ToolBus architecture.

Other attempts to integrate heterogeneous resources include ISYS and Biopipe. ISYS (Siepel *et al.*, 2001) uses a decentralized, component-based approach with a design similar to CORBA and SOAP/WSDL. It allows dynamic discovery of services via a broker. The data-model is heavily object-based and is implemented through a set of Java interfaces. Biopipe (Hoon *et al.*, 2003) is a flexible framework that aims to allow researchers to focus on designing an analysis pipeline. Analysis modules and configuration parameters are chosen and the protocol, data sources and modules are wrapped in XML. While our needs could probably have been met by either Taverna or Biopipe, our requirements were much simpler than the pipeline models which these tools provide. As mentioned above, we do not wish to transform the output of one tool into input for another tool, but simply wish to feed the same type of data (a protein sequence) into a number of separate analysis tools.

We have therefore designed a system known as APAT ('Automated Protein Annotation Tool') to achieve our aims of (i) allowing one or more sequences to be analyzed in a single run, (ii) using multiple prediction/annotation tools residing locally or over the web through normal CGI scripts or Web-services, (iii) obtaining the results in a common text format for further analysis, (iv) providing consistent visual presentation of results and (v) making the implementation of wrappers to additional tools as straightforward and language-independent as possible. An additional aim is that user-intervention is kept to a minimum. We thus assume that default parameters for the various prediction programs are

adequate, although this can easily be over-ridden. In common with Biopipe, we have decided to wrap the input and output of each tool in XML. The system then submits one or more sequences to a set of prediction tools which return their results in an XML format (APATML) which can be further analyzed, or displayed as HTML via a display program.

2 METHODS

A number of web-based and local tools were analyzed to discover the types of information which they require as input and return as output. These tools included NetPhos (Blom *et al.*, 1999), NetOGlyc (Julenius *et al.*, 2004), DAS-TM Filter (Cserzo *et al.*, 2004), TargetP (Emanuelsson *et al.*, 2000), PsiPred (McGuffin *et al.*, 2000), InterProScan (Zdobnov and Apweiler, 2001), LOCtree (Nair and Rost, 2004), Predotar (Small *et al.*, 2004), BLAST (Altschul *et al.*, 1990, 1997), FingerPrintScan (Scordis *et al.*, 1999), TMHMM (Krogh *et al.*, 2001), PATS (Zuegge *et al.*, 2001), ScanProsite (Gattiker *et al.*, 2002; Hulo *et al.*, 2004) and SMART (Schultz *et al.*, 1998).

2.1 Input data

Many of the programs have numerous options, but supply defaults for the vast majority of these. For purposes of bulk scanning of sequence data, accepting these default values should be perfectly acceptable. From the tools we examined, in addition to the sequence, the only data that must be supplied are: an identifier for the sequence, an email address and an indication of whether or not a sequence is of plant origin.

We have designed a very simple DTD which is used to encode the input sequence and any parameters required by the annotation/prediction programs. We have also provided a Perl script which converts a FASTA file into this format and accepts any additional parameters on the command line or interactively.

Some programs may have more extensive input requirements. By definition, XML is extensible so the DTD for input data can easily be extended to allow for additional input requirements of specific wrappers. These additional tags will simply be ignored by wrapper scripts which don't need them. An example XML file is shown on the web page (`http://www.bioinf.org.uk/apat/`).

2.2 Output data

We found that the annotations provided by these programs could all be described by 6 data types: text or numeric labels at the level of residues, domains or whole sequences. Table 1 shows examples of the annotations returned by different tools. In the case of residue-level annotations, a value is often provided for every residue in a sequence. Typical examples are secondary structure prediction, trans-membrane prediction, glycosylation and phosphorylation site prediction. In some cases, graphical display of such annotations in the form of both line-charts and bar-charts can be useful and it is necessary that this requirement can be flagged.

Domain-level annotations can be viewed as an extension of residue-level annotations in which discrete continuous stretches of residues are given the same label. However, the semantic meaning is somewhat different. While a residue-level annotation applies to that residue in isolation, a domain-level annotation is not meaningful in the context of a single residue. For example, domain-level annotations are generally used for the results of pattern or profile searches such as FingerPRINTScan (Scordis *et al.*, 1999), ProSite (Gattiker *et al.*, 2002; Hulo *et al.*, 2004), InterProScan (Zdobnov and Apweiler, 2001), and SMART (Schultz *et al.*, 1998). It would not be meaningful to say that a single residue matched one of the patterns which these tools recognise. From a presentational viewpoint, one generally wishes such annotations to be provided in a tabular form rather than indicated on the sequence itself. Taking FingerPRINTScan as an example, for each fingerprint matched, the server at `http://www.ebi.ac.uk/printsscan/` returns 7 numbers (the number of motifs matched; the number of motifs in the fingerprint; SumID; AveID; ProfScore; P-value; E-value) and 2 strings (the fingerprint name; an indication of which motifs within the fingerprint match). In addition, one additional string is returned for each motif matched indicating the matched residues. From this a residue range can be calculated.

Sequence-level annotations provide a value which is applicable to the whole sequence. Examples are protein localization predictions such as TargetP (Emanuelsson *et al.*, 2000), Predotar (Small *et al.*, 2004), LOCtree (Nair and Rost, 2004), PATS (Zuegge *et al.*, 2001) and the results of BLAST (Altschul *et al.*, 1990, 1997) searches. Semantically this is similar to a domain-level annotation, but the presentation requirements are rather different.

Some servers provide annotations at more than one level. For example, TMHMM (Krogh *et al.*, 2001) provides per-residue values indicating the probability that an individual residue is in a trans-membrane region. In addition, it summarizes ranges of residues predicted to form trans-membrane helices indicating their orientation together with an overall significance value. Finally it generates a number of pieces of summary data such as the number of amino acids predicted to be in trans-membrane helices and the number within the first 60 amino acids. It therefore has annotations at all three levels: per-residue, per-domain and per-sequence.

Since XML makes no distinction between numeric and character data (everything is stored as plain text), we decided to simplify this scheme further by treating the numeric and text annotations for domains and for sequences as single types. However, we retained the distinction for per-residue annotations since we may wish to generate graphs of numeric data while there will be no such requirement for character data.

We therefore have just four data types which can be used to encapsulate the annotations from all the tools likely to be encountered: per-residue numbers, per-residue strings, per-domain values and per-sequence values.

We considered the use of the DAS XML format (`http://www.biodas.org/documents/spec.html`) for our annotation requirements, but decided against it for a number of reasons. First, simplicity was a priority to allow additional service wrappers to be written easily. Being designed primarily for DNA-level annotations, DAS is unnecessarily complex for our purposes having many redundant fields. Also

Tool	Residue level		Domain level		Sequence level	
	number	text	number	text	number	text
NetPhos 2.0	√					
NetOGlyc	√					
DAS-TM	√		√	√	√	
TargetP					√	√
LOCtree					√	√
PsiPred	√	√				
Predotar					√	√
BLAST					√	√
FingerPrintScan			√	√		
TMHMM	√			√	√	
PATS					√	√
ScanProsite			√	√		
SMART			√	√		
InterProScan			√	√		

Table 1. Annotation types returned by a number of example tools.

DASGFF and DASSTYLE elements need to be combined to achieve the simple task of indicating visual annotations. Second, there is no direct way within the DASSTYLE elements to specify a requirement for a graph to be displayed. One would either have to extend the DAS XML specification or co-opt existing glyph styles to have non-standard meanings. Third, providing the semantics are easily transferable, conversion between XML formats is straightforward using XSLT, so a specific format can be chosen to ease the burden of implementing a particular system.

The actual output of many web-based servers provides visual highlighting, graphs and extensive text. We capture only the essential information from this — alternative presentation issues can be addressed in a display program. However, in addition to the pure annotation data, we do allow the storage of limited meta-data about what the annotations mean. For example, at the server level, we store the name and version number of the program, the run-time parameters and a textual description of the program's function. At the per-sequence annotation level, we can store extensive text associated with annotations and at the per-domain level we allow a description to be stored associated with a prediction. This accounts for servers such as InterProScan which potentially identify more than one region of a protein using a number of underlying databases/algorithms. We allow simple storage of the annotated residue range, together with database name (e.g. PRINTS) and annotation (e.g. SH2_DOMAIN) which is stored separately from an explanation of what a 'PRINTS SH2_DOMAIN' actually is.

In the case of numeric per-residue annotations, the DTD also allows one to indicate whether a graph (either a line chart or a bar chart) should be provided to display the data. In addition, we have provided a mechanism by which individual residues can be highlighted as 'positive' predictions. Initially we had hoped simply to provide some threshold value such that any per-residue numeric scores higher than the threshold could be flagged by the display program. However, some of the servers have much more complex threshold schemes. For example NetOGlyc makes a positive prediction if one score (the 'G-score') is > 0.5 or, for threonines, if the G-score is < 0.5, but the 'I-score' is > 0.5 and there are no other sites predicted within 10 residues. We therefore opted for the DTD to include a list of the residues considered as positive predictions together with a description of how such residues are identified. This moves the logic of indicating a positive prediction back to the service wrapper rather than the display program.

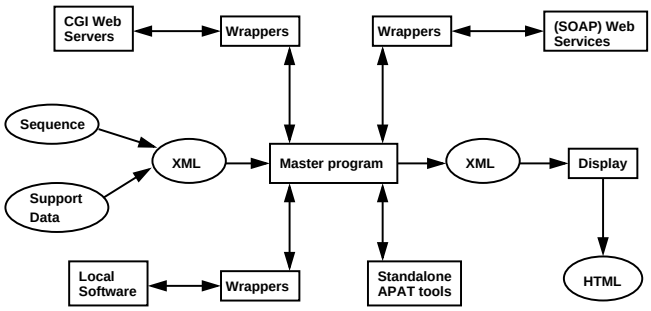

Fig. 1. Overall architecture of the APAT system.

The web site (`http://www.bioinf.org.uk/apat/`) gives simplified extracts of the APATML file which summarize the main aspects of the DTD for per-residue, per-domain and per-sequence annotations. In addition, the complete DTD and a detailed description can be downloaded.

2.3 System architecture

The overall architecture of the APAT system is shown in Figure 1. The system is implemented in Perl using the XML::DOM module for parsing XML files. The software consists of 3 major components: (i) a 'master' script which reads an XML input file containing the sequence and dispatches it to each service wrapper, (ii) wrappers for each annotation/prediction service, (iii) a display program which converts the APATML output from the annotation/prediction services to HTML for display. This display program could be replaced by any number of post-analysis scripts. In addition, since the system may be used to process a batch of

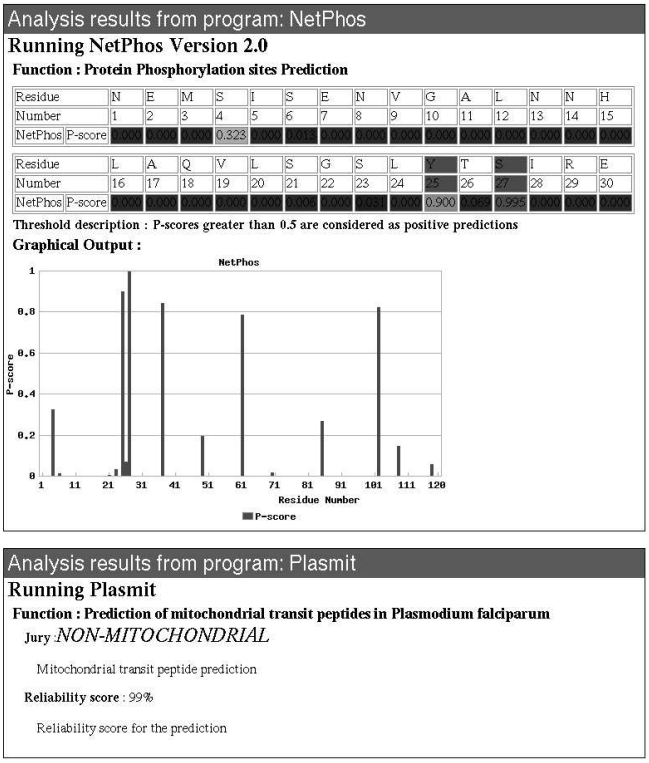

Fig. 2. Sample HTML output from the APAT system showing per-residue and per-sequence annotations. The per-residue annotation has been edited to two lines for brevity. A full example may be seen on the APAT web site.

sequences, we have implemented a short script which will run through all the input XML files in a directory and process each in turn using the 'master' script. While the system is not really designed for online use, we have implemented a simple web interface primarilly for demonstration purposes, although this could prove useful for intranet installations.

The most complex part of the system is the display program which provides a uniform display for all the annotation services. The APATML file is read using XML::DOM. Per-sequence annotations, which are applied to the whole sequence, are simply displayed as text (see Figure 2), while per-domain annotations are displayed as a simple table of results with associated descriptions following the table. Per-residue annotations are presented in the form of a table in which numeric values are coloured on a scale from blue through green to red. In addition, residues marked in the APATML as 'positive predictions' are highlighted and, where indicated by the APATML, the GD::Graph Perl module is used to provide graphical display of per-residue annotations (see Figure 2).

Each of the service wrappers is implemented as a stand-alone 'plug-in'. The master program simply identifies all the plug-ins available and runs each in turn. This design allows individual service wrappers to be implemented and debugged as stand-alone code and simply placed in a standard directory for integration into the system. Plug-ins can be implemented such that they provide a self-contained annotation service, but in practice they are generally wrappers to some other tool. Such tools may reside locally or remotely, either as Web-services or CGI-based servers on the web. Remote services may be accessed via SOAP or by 'screen-scraping' of web pages respectively. Since the only requirement of the plug-ins is that they read and write XML, they can be implemented in any programming language and integrated seamlessly into the APAT system. We have implemented a number of plug-in service wrappers in Perl for which the SOAP::Lite and LWP packages make access to Web-services and CGI-based servers straightforward.

Implementation of additional service wrappers is relatively straightforward and validation against the APATML DTD ensures that the results can be converted to HTML using our display program. The DTD was prepared with the aid of the excellent XML-to-DTD conversion utility from Hit Software available at `http://www.hitsw.com/xml_utilites/` before careful manual checking and modification.

3 RESULTS AND DISCUSSION

APAT is designed to perform a very simple but repetitive task in a straightforward manner: it allows one or more sequences to be presented to a number of differerent annotation/prediction servers, collating the results and presenting them in a consistent format for automated or visual analysis. Our approach contrasts with Taverna (Oinn *et al.*, 2004) and ToolBus (Eckart and Sobral, 2003). These are hugely capable workflow-based systems which, while clearly capable of similar things, come with an overhead of complexity requiring some considerable investment in time to learn how they can be extended. In addition, Taverna, in its current form, is not designed to highlight the key information needed by a Biologist in a simple and consistent format.

We have analyzed the output produced by a wide range of protein sequence and annotation tools and determined that all annotations can be expressed in one of four ways (character or numeric per-residue annotations or annotations at the per-domain or per-sequence level). On the basis of this analysis, we have designed an XML DTD to abstract and encode the annotations provided by any prediction server. On the basis of this DTD, we have gone on to design a display tool and wrappers to a number of annotation/prediction services running both locally and remotely.

The system is designed to be downloaded and run locally allowing the user to run many annotation/prediction services on one or more sequences without manual intervention. Results are presented in a coherent and consistent form making comparative analysis straightforward.

Users can easily choose which plug-in annotation services they wish to use and implementation of additional service wrappers is straightforward using the existing wrappers as examples. It should take an experienced Perl programmer no more than a couple of hours to implement an additional wrapper. Compliance of the resulting XML can be checked against the APATML DTD to ensure compatibility.

While the system allows multiple single sequences to be sent to prediction servers, it makes no attempt to handle servers which require multiple sequences (for example, multiple sequence alignments and phylogeny). Similarly there is no ability to deal with servers which return much more complex data such as three-dimensional models built by comparative modelling. Such additions are being considered for future versions of the system. Currently the display tool presents the results of each annotation/prediction server separately. A further possible enhancement would be to display multiple residue-level annotations on a single view of the sequence as is done in DAS.

Source code for the master and display programs and for a number of plug-in service wrappers may be downloaded from the web site together with the DTD and documentation. The download also provides the scripts for converting a sequence to the input XML format and for running all the input XML files in a specified directory through the APAT system. An installation script is provided which installs the software and, optionally, the web interface. Documentation includes detailed descriptions of the APATML format and a guide to implementing service wrappers. As a demonstration, we have also provided a web-based tool that allows a single sequence to be submitted to a small number of prokaryotic and eukaryotic prediction tools. The system may be accessed at `http://www.bioinf.org.uk/apat/`. The web site also enables users to upload service wrappers they may have written to share them with the community.

4 ACKNOWLEDGEMENTS

SVVD is funded by a Felix Scholarship held at The University of Reading.

REFERENCES

Altschul, S. F., Gish, W., Miller, W., Myers, E. W. and Lipman, D. J. (1990) Basic local alignment search tool, *J. Mol. Biol.*, **215**, 403–410.

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D. J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs, *Nuc. Ac. Res.*, **25**, 3389–3402.

Apweiler, R., Attwood, T. K., Bairoch, A., Bateman, A., Birney, E., Biswas, M., Bucher, P., Cerutti, L., Corpet, F., Croning, M. D., Durbin, R., Falquet, L., Fleischmann, W., Gouzy, J., Hermjakob, H., Hulo, N., Jonassen, I., Kahn, D., Kanapin, A., Karavidopoulou, Y., Lopez, R., Marx, B., Mulder, N. J., Oinn, T. M., Pagni, M., Servant, F., Sigrist, C. J. and Zdobnov, E. M. (2001) The InterPro database, an integrated documentation resource for protein families, domains and functional sites, *Nuc. Ac. Res.*, **29**, 37–40.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M. and Sherlock, G. (2000) Gene ontology: tool for the unification of biology. The Gene Ontology Consortium, *Nature: Genetics*, **25**, 25–29.

Bairoch, A. and Apweiler, R. (2000) The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000, *Nuc. Ac. Res.*, **28**, 45–48.

Blom, N., Gammeltoft, S. and Brunak, S. (1999) Sequence and structure-based prediction of eukaryotic protein phosphorylation sites, *J. Mol. Biol.*, **294**, 1351–1362.

Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.-C., Estreicher, A., Gasteiger, E., Martin, M. J., Michoud, K., O'Donovan, C., Phan, I., Pilbout, S. and Schneider, M. (2003) The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003, *Nuc. Ac. Res.*, **31**, 365–370.

Cserzo, M., Eisenhaber, F., Eisenhaber, B. and Simon, I. (2004) TM or not TM: transmembrane protein prediction with low false positive rate using DAS-TMfilter, *Bioinformatics*, **20**, 136–137.

Dowell, R. D., Jokerst, R. M., Day, A., Eddy, S. R. and Stein, L. (2001) The distributed annotation system, *BMC Bioinformatics*, **2**, 7–7.

Durinck, S., Moreau, Y., Kasprzyk, A., Davis, S., De Moor, B., Brazma, A. and Huber, W. (2005) BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis, *Bioinformatics*, **21**, 3439–3440.

Eckart, J. D. and Sobral, B. W. S. (2003) A life scientist's gateway to distributed data management and computing: the PathPort/ToolBus framework, *OMICS*, **7**, 79–88.

Emanuelsson, O., Nielsen, H., Brunak, S. and von Heijne, G. (2000) Predicting subcellular localization of proteins based on their N-terminal amino acid sequence, *J. Mol. Biol.*, **300**, 1005–1016.

Frishman, D., Albermann, K., Hani, J., Heumann, K., Metanomski, A., Zollner, A. and Mewes, H. W. (2001) Functional and structural genomics using PEDANT, *Bioinformatics*, **17**, 44–57.

Gattiker, A., Gasteiger, E. and Bairoch, A. (2002) ScanProsite: a reference implementation of a PROSITE scanning tool, *Appl. Bioinformatics*, **1**, 107–108.

Hoon, S., Ratnapu, K. K., Chia, J.-M., Kumarasamy, B., Juguang, X., Clamp, M., Stabenau, A., Potter, S., Clarke, L. and Stupka, E. (2003) Biopipe: a flexible framework for protocol-based bioinformatics analysis, *Genome Res*, **13**, 1904–1915.

Hubbard, T., Barker, D., Birney, E., Cameron, G., Chen, Y., Clark, L., Cox, T., Cuff, J., Curwen, V., Down, T., Durbin, R., Eyras, E., Gilbert, J., Hammond, M., Huminiecki, L., Kasprzyk, A., Lehvaslaiho, H., Lijnzaad, P., Melsopp, C., Mongin, E., Pettett, R., Pocock, M., Potter, S., Rust, A., Schmidt, E., Searle, S., Slater, G., Smith, J., Spooner, W., Stabenau, A., Stalker, J., Stupka, E., Ureta-Vidal, A., Vastrik, I. and Clamp, M. (2002) The Ensembl genome database project, *Nuc. Ac. Res.*, **30**, 38–41.

Hulo, N., Sigrist, C. J. A., Le Saux, V., Langendijk-Genevaux, P. S., Bordoli, L., Gattiker, A., De Castro, E., Bucher, P. and Bairoch, A. (2004) Recent improvements to the PROSITE database, *Nuc. Ac. Res.*, **32**, D134–D137.

Julenius, K., Mølgaard, A., Gupta, R. and Brunak, S. (2004) Prediction, conservation analysis and structural characterization of mammalian mucin-type O-glycosylation sites, *Glycobiology*.

Krogh, A., Larsson, B., von Heijne, G. and Sonnhammer, E. L. (2001) Predicting transmembrane protein topology with a Hidden Markov Model: Application to complete genomes, *J. Mol. Biol.*, **305**, 567–580.

Laskowski, R. A. (2001) PDBsum: summaries and analyses of PDB structures, *Nuc. Ac. Res.*, **29**, 221–222.

McGuffin, L. J., Bryson, K. and Jones, D. T. (2000) The PSIPRED protein structure prediction server, *Bioinformatics*, **16**, 404–405.

Milburn, D., Laskowski, R. A. and Thornton, J. M. (1998) Sequences annotated by structure: a tool to facilitate the use of structural information in sequence analysis, *Protein Eng.*, **11**, 855–859.

Nair, R. and Rost, B. (2004) LOCnet and LOCtarget: sub-cellular localization for structural genomics targets, *Nuc. Ac. Res.*, **32**, W517–W521.

Nayal, M., Hitz, B. C. and Honig, B. (1999) GRASS: a server for the graphical representation and analysis of structures, *Protein Sci.*, **8**, 676–679.

Neshich, G., Togawa, R. C., Mancini, A. L., Kuser, P. R., Yamagishi, M. E. B., Pappas, G., Torres, W. V., Fonseca e Campos, T., Ferreira, L. L., Luna, F. M., Oliveira, A. G., Miura, R. T., Inoue, M. K., Horita, L. G., de Souza, D. F., Dominiquini, F., Alvaro, A., Lima, C. S., Ogawa, F. O., Gomes, G. B., Palandrani, J. F., dos Santos, G. F., de Freitas, E. M., Mattiuz, A. R., Costa, I. C., de Almeida, C. L., Souza, S., Baudet, C. and Higa, R. H. (2003) STING Millennium: A web-based suite of programs for comprehensive and simultaneous analysis of protein

structure and sequence, *Nuc. Ac. Res.*, **31**, 3386–3392.

Oinn, T. M. (2003) Talisman–rapid application development for the grid, *Bioinformatics*, **19 Suppl 1**, i212–i214.

Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M. R., Wipat, A. and Li, P. (2004) Taverna: a tool for the composition and enactment of bioinformatics workflows, *Bioinformatics*, **20**, 3045–3054.

Pruess, M., Kersey, P. and Apweiler, R. (2005) The Integr8 project–a resource for genomic and proteomic data, *In Silico Biol.*, **5**, 179–185.

Rost, B. (1996) PHD: predicting one-dimensional protein structure by profile-based neural networks, *Meth. Enzymol.*, **266**, 525–539.

Schultz, J., Milpetz, F., Bork, P. and Ponting, C. P. (1998) SMART, a simple modular architecture research tool: Identification of signaling domains, *Proc. Natl. Acad. Sci. USA*, **95**, 5857–5864.

Scordis, P., Flower, D. R. and Attwood, T. K. (1999) FingerPRINTScan: intelligent searching of the PRINTS motif database, *Bioinformatics*, **15**, 799–806.

Senger, M., Rice, P. and Oinn, T., (2003). SoapLab — a unified sesame door to analysis tools. In *Proceedings of the UK e-Science All Hands Meeting 2003*.

Siepel, A., Farmer, A., Tolopko, A., Zhuang, M., Mendes, P., Beavis, W. and Sobral, B. (2001) ISYS: a decentralized, component-based approach to the integration of heterogeneous bioinformatics resources, *Bioinformatics*, **17**, 83–94.

Small, I., Peeters, N., Legeai, F. and Lurin, C. (2004) Predotar: A tool for rapidly screening proteomes for N-terminal targeting sequences, *Proteomics*, **4**, 1581–1590.

Zdobnov, E. M. and Apweiler, R. (2001) InterProScan–an integration platform for the signature-recognition methods in InterPro, *Bioinformatics*, **17**, 847–848.

Zuegge, J., Ralph, S., Schmuker, M., McFadden, G. I. and Schneider, G. (2001) Deciphering apicoplast targeting signals–feature extraction from nuclear-encoded precursors of *Plasmodium falciparum* apicoplast proteins, *Gene*, **280**, 19–26.