Managing Evidence from Multiple Gene Finding Resources via an XML-based Integration Architecture

Andigoni Malousi*, Vassilis Koutkias and Nicos Maglaveras

Lab. of Medical Informatics, Faculty of Medicine, PO BOX 323, 54124,
Aristotle University of Thessaloniki, Greece

Summary

While biological processes underlying gene expression are still under experimental research, computational gene prediction techniques have reached high level of sophistication with the employment of efficient intrinsic and extrinsic methods that identify protein-coding regions within query genomic sequences. Their ability though to delineate the exact exon boundaries is characterized by a trade off between sensitivity and specificity and still is prone to alternations in gene regulation during transcription and splicing and to inherent complexities introduced by the implemented methodology. Evaluation studies have shown that combinatorial approaches exhibit improved accuracy levels through the integration of evidence data from multiple resources that are further assessed in order to end up with the most probable gene assembly.

In this work, we present an integration and information handling architecture that exploits evidence derived from multiple gene finding resources, in order to generate machine-readable representations of optimal/suboptimal gene structure predictions, signal features identification and high scoring similarity matches. Unlike most combinatorial techniques, which end up with the most probable gene assembly, the objective of this architecture is to support advanced information handling mechanisms that may give more in depth insights on the underlying gene expression machinery and the alternations that may occur. Technically, XML was adopted to build and interchange structured data among the architecture's components together with relevant technologies offering graphical representations and queries formulation/execution over single/multiple information sources.

1 Introduction

Computational gene prediction constitutes a favorable way to address the genomic annotation problem in relation to the expensive and time-intensive experimental techniques. Most gene finding resources are based on descriptive models of the content-based and signal-based features that, as long as they are highly accurate, can decipher cellular processes involved in transcription, splicing and translation mechanisms. The identification of protein-coding genes by algorithmic approaches has been addressed by either extrinsic or intrinsic methods [1]. Extrinsic methods essentially rely on homologies identified by aligning a query genomic sequence against databases of annotated proteins or cDNA/ESTs. High scoring matching sequences may indicate potential evolutionary relationships with the query sequence, nevertheless, up to date extrinsic methods are used mostly to verify the existence/absence of a gene structure, since they fail to recognize new gene structures [2].

*Corresponding author: andigoni@med.auth.gr

Unlike similarity-based methods, intrinsic or ab-initio methods are beneficial in identifying statistically significant features that are not represented by homologous proteins or ESTs. Intrinsic gene prediction in eukaryotes involves the localization of non-continuous coding regions (exons) that are separated by typically long stretches of non-coding fragments (introns). Intrinsic methods take advantage of the compositional differences between coding and non-coding regions, e.g., codon bias, CG-content and the incidence of specific sequence signals characterized by consensus patterns in order to build the most probable exon assembly.

Although intrinsic gene structure finders have reached high level of sophistication in modeling transcriptional and splicing signals, they unexceptionally fail to identify alternative spicing events and atypical gene structures [3]. This issue is very important considering that alternative splicing is not an exceptional event, but occurs very frequently in form of poorly characterized consensus sequences. In human, for example, alternative splicing has been verified in 35-74% of the about 30.000 genes [4], [5] and is believed that to a great extent it presides over protein diversities [6]. Moreover, intrinsic gene structure finders fail to identify atypical transcripts, while the overall prediction accuracy is further subjected to factors related to the size and number of exons contained in the query sequence. Finally, terminal exons at both 5' and 3' regions are poorly identified since they have a single adjacent splice site, whereas in single exon genes flanking splice sites are totally absent [2]. Although alternative transcripts have not been modeled by gene structure predictors, suboptimal exons as well as less-probable gene assemblies may give indications of alternatively transcribed genes [7]. In addition, a suboptimal exonic feature that is adjacent to the optimal one may indicate an erroneous best prediction [8].

Considering the above-mentioned research studies, it becomes obvious that the association of statistically less significant features may be proved quite valuable in improving the overall prediction accuracy. Evaluation studies of various widely-known intrinsic gene structure finders have shown that although the predictive power of the underlying probabilistic models is satisfactory at nucleotide and exonic level, sensitivity and specificity at gene level are considerably lower [9], [10]. This problem is addressed more efficiently by combinatorial methods, which integrate evidence from multiple independent signal sensors and similarity-based tools. Similar to intrinsic methods, combinatorial approaches end up with the most probable exon assembly but still they do not consider alternative transcript forms or atypical gene structures [11].

Given our limited knowledge on the underlying gene expression mechanisms, it is rather simplistic to address genomic sequences annotation through a single, statistically significant gene parse. Contrariwise, a single gene under certain conditions can produce multiple protein products. Motivated by these observations, we developed an integrated approach that addresses computational gene prediction by associating a) evidence data extracted from multiple types of analysis and b) statistically less significant features that may indicate alternative gene transcripts. The objective was to seamlessly integrate evidence into machine-readable data sources that can be further processed facilitating evidence refinement. To increase the expressiveness of the data extracted and interchanged among the architecture's components, XML was used along with XML-related technologies, such as XQuery, to address information handling requirements. The types of analysis and the components of the proposed architecture are described below.

2 Resource Classification and Description

Considering the wide variety of methodologies addressing genomic sequences annotation, one of the most challenging issues is to overcome technical difficulties implied by their design and implementation constraints and perform seamless integration of the complementary evidence derived [12]. In computational gene prediction, most resources are publicly accessible through Web interfaces and offer different types of information expressed in self-defined formats. A first step towards integration of these resources is to classify them according to the type of information they extract. Accordingly, in the scope of this work, three classes of computational gene prediction resources were defined:

- ***Gene Structure Detector (GSD)***: *GSDs* exploit the intrinsic information of a query DNA sequence in order to build a gene assembly that most likely encodes the actual, biologically determined pre-mRNA. Most *GSDs* implement various statistical approaches in order to discriminate protein-coding regions from intronic fragments using both content-based and signal-based intrinsic information. High quality features are then assembled into complete gene structures with respect to specific selection rules usually through dynamic programming implementations. Currently, there are about 25 *GSDs* (some of them associate homology evidence) trained on species-specific datasets [1].

- ***Sequence Signal Detector (SSD)***: *SSDs* identify specific signals in a query DNA sequence by exploiting exclusively high quality intrinsic information that indicates potential target sites. Promoters, splice junctions, transcription start sites and polyadenylation sites are the most commonly predicted signals. Typically, *SSDs* end up with a list of probable signals identified within the query sequence along with the associated probabilities/scores by which more than one gene assembly may be composed. Thus, *SSDs* cannot be used merely to determine the exact exon boundaries and, due to alternative compositional forms that a signal may be appeared with, it is difficult to ensure the accuracy of the signal detection. *SSDs* though are quite valuable in supporting intrinsic gene structure finders giving additional evidence of the predicted exons.

- ***SImilarity-based Detector (SID)***: *SIDs* perform alignments of a query genomic sequence against databases of annotated proteins and cDNAs/ESTs, aiming to signify approximate locations of strong hits that can be further used to delineate exon boundaries. Based on the underlying alignment algorithm and the target databases, many BLAST-like algorithms have been developed performing species-specific and database-specific homology searches. The accuracy of the identified high scoring segments is estimated by an *expectation value (e-value)*, that measures the probability of a match to occur by chance.

Regardless of the identifiable functional orientations, most gene finding resources share some common technical characteristics. Specifically, the resulting information is provided through flat files of unstructured or semi-structured data formats that are intended to be human-readable rather than machine-processable. As far as the design and implementation details are concerned, most resources are heterogeneous (even belonging to the same class)in both their input/output data representation and the required configuration parameters. Moreover, most resources are highly autonomous, meaning that any modification in their design and/or implementation can be performed without prior public notification [12].

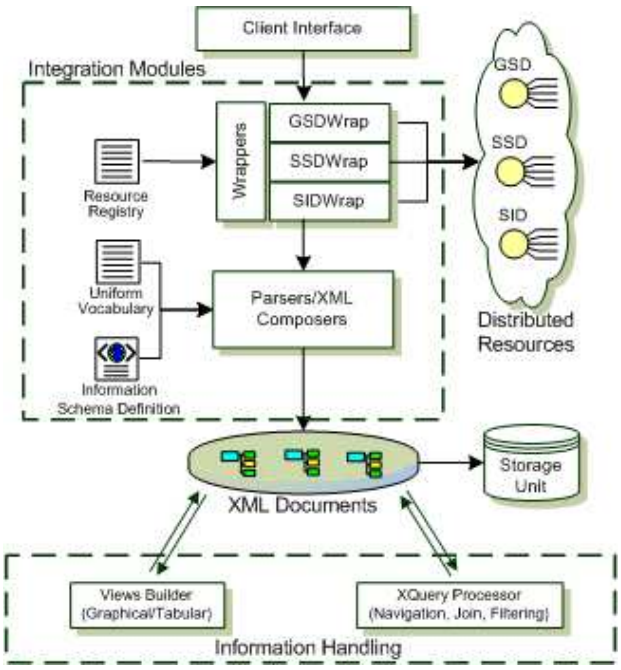

Figure 1: Schematic view of the architecture's structural and functional components.

3 Architecture Description

Figure 1 illustrates a conceptual view of the structural and functional components designed and implemented to fulfill system's requirements. The architecture serves two basic functionalities:

- seamless integration of the incorporated resources representing all three resource classes, and
- efficient information handling through the implementation of combined graphical views and a query formulation/processing mechanism.

XML (eXtensible Markup Language) is the core technology that the implementation of both functionalities is based on [13]. XML is a favorable technology that has been used to describe and exchange various types of biological data [14]. In addition, XML has been adopted as the standard data format in various bioinformatics integration frameworks, offering efficient representation and processing mechanisms [15], [16], [17]. In this work, XML-related technologies, such as XQuery [18] were also incorporated to further exploit the expressiveness and flexibility of XML, as described below.

3.1 Integration Modules

Given a set of distributed gene finding resources and a user request for gene identification, the first step in building uniform output representations is to construct appropriate wrappers that access resources and retrieve the resulting evidence. The idea of building a generalized wrapper that addresses technical requirements of all resources is rather impractical, due to inherent incompatibilities detected even among resources of the same class. Incompatibilities are expressed in form of different configuration parameters related with the underlying algorithm and

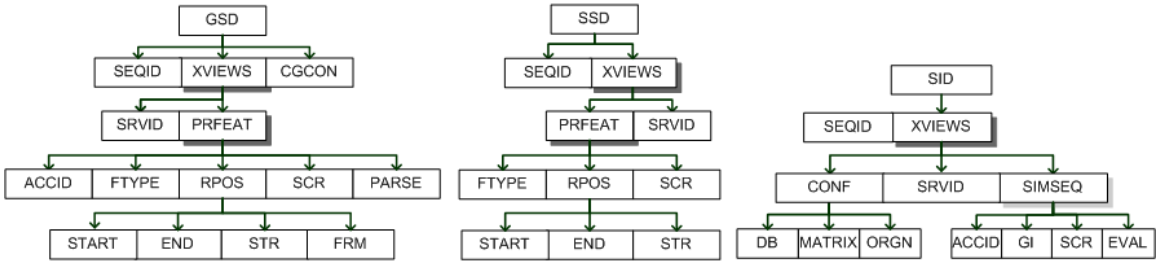

Figure 2: Tree-view of the schemas defined in the *Information Schema Definition* module. Elements in shaded boxes indicate potential recurrent appearance within the XML document.

diverse encodings used to capture input/output options. To cope with this issue, three types of wrappers were defined, namely, *GSDWrap*, *SSDWrap* and *SIDWrap*. Instructions on how to locate, access and retrieve information from each one of the incorporated resources are kept separately in a *Resource Registry*. The *Resource Registry* module is an editable data source that allows modifications/updates of the configuration parameters, as well as the incorporation of additional resources in the proposed architecture.

The transformation of the extracted outcomes into structured XML-based documents is addressed by corresponding *Parsers* and *XML Composers*. To cope with the different terminology describing same concepts, a *Uniform Vocabulary* is introduced that converts self-defined encodings into globally accepted terms. The formatting instructions of the resulting XML documents are defined in the *Information Schema Definition* module. Figure 2 illustrates a hierarchical view of the schemas defined for each type of analysis. The *SIMSEQ* element defined in the *SID* schema and the *PRFEAT* element in *GSD* and *SSD* schemas may appear more than once with respect to the amount of evidence data extracted (homologous sequences, intrinsic and extrinsic features respectively). Accordingly, the *XVIEWS* tag in compliant XML documents is repetitive in proportion to the number of tools defined in the *GSD*, *SSD* and *SID* classes and identified by the *SRVID* element. The resulting XML documents can be deposited in a *Storage Unit* and/or further processed by the *Information Handling* modules, as described below.

3.2 Information Handling

Given a set of well-formed XML documents that comply with the formatting rules of the predefined XML Schemas, the *Information Handling* modules address further processing of the resulting outcome by providing two types of combined analysis:

- build aggregated views of the statistically significant features in form of graphical/tabular expressions through the *Views Builder*, and
- formulate and execute complex queries against single or multiple types of evidence using XQuery/XPath expressions via the *XQuery Processor*.

The *Views Builder* discriminates positional and score properties among tagged elements and dynamically generates combined graphical views, illustrating the identified exonic and signal features into panels of user-defined lengths. In addition, comparative tabular views are built through XML to HTML transformations that are performed by configurable XSLTs (eXtensible Stylesheet Language Transformations). XSL transformations can be applied to an XML

Table 1: List of the analysis tools representing *GSD*, *SSD* and *SID* classes that are incorporated in the implementation of the architecture.

Resource	Class	Description	Availability	Special features
GENSCAN	GSD	Predicts single/multiple, complete/partial genes	http://genes.mit.edu/ GENSCAN.html	Suboptimal exon predictions on both strands
HMMGENE	GSD	Predicts single/multiple, complete/partial genes	http://www.cbs.dtu.dk/ services/HMMgene/	Suboptimal gene structures on both strands
FGENESH	GSD	Predicts single/multiple, complete/partial genes	http://softberry.com	Optimal gene predictions on both strands
FIRSTEF	SSD	Predicts 5' terminal exons and promoters	http://rulai.cshl.org/ tools/FirstEF/	Probabilities, both strands, cutoffs
BDGP NNPP	SSD	Identifies TATA-boxes TSS, CAAT-boxes etc., to predict promoters	http://www.fruitfly.org/ seq_tools/promoter.html	Probabilities, both strands, cutoffs
BDGP NNSSP	SSD	Two neural networks that identify acceptor donor sites	http://www.fruitfly.org/ seq_tools/splice.html	Probabilities, both strands, cutoffs
HCPOLYA	SSD	Searches for poly-A sites of 6-12 bases	http://l25.itba.mi.cnr.it/ ~webgene/ wwwHC_polya.html	Direct or reverse strand, no score estimation
BLASTX	SID	Database searches after query translation into all six frames	http://www.ncbi.nlm. nih.gov/blast/	Species-specific, multiple DBs, submatrices

document containing preview data and define the type and structure of the represented features. Unlike graphical views, which depict merely intrinsic features, tabular expressions may also associate extrinsic indications within human-readable and comprehensive HTML reports.

Except for graphical representations, the transformation of flat files into XML-formatted documents describing complementary information is further exploited by the *XQuery Processor*. The *XQuery Processor* enables the formulation/execution of various types of requests against single or multiple XML documents. Operations such as join, filtering and navigation on selected tuples are served through XPath statements and/or FLWR (For-Let-Where-Return) expressions. The latter may result in the composition of new XML documents containing matching data [19]. Examples of XPath/XQuery expressions are presented in the following.

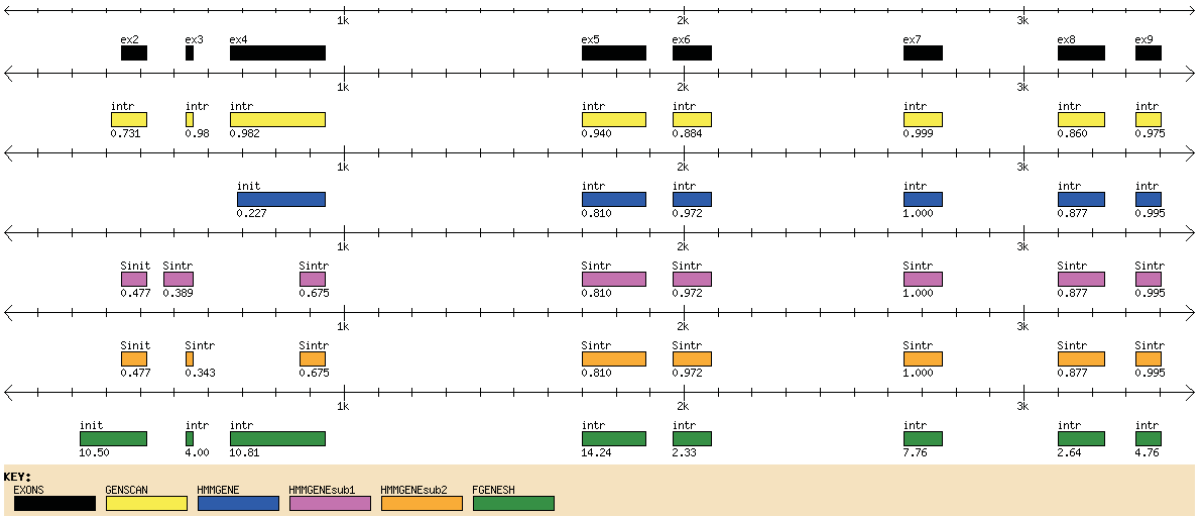

Figure 3: Graphical representation of the optimal/suboptimal, initial (*init*) and internal (*intr*) features identified within a genomic sequence containing exons 2-9 of the human *p53* tumor suppressor gene by the three *GSDs*. The upper track illustrates the actual exonic features (*exX*, *X*:number of exon). The *"S"* prefix indicates suboptimal features.

4 Implementation

The applicability of the proposed architecture was tested on a set of widely-known resources that are freely accessible through Web interfaces (Table 1). GENSCAN, HMMGENE and FGENESH were used as example tools of the *GSD* class. GENSCAN and HMMGENE can identify suboptimal exons and gene structures, respectively, by defining lower cutoffs. FGENESH is a highly accurate gene predictor that focuses more on signal information rather than content-based evidence which is more effectively identified by GENSCAN [2].

Three *SSDs*, namely, BDGP's Neural Network Promoter Predictor, BDGP's Neural Network Splice Site Predictor and HCPOLYA were incorporated in order to predict high quality splice junctions, promoters and polyadenylation sites respectively. In addition, a specialized first exon finder, FIRSTEF, was used to identify coding features at the 5' UTR, since according to Mironov et al. 80% of the alternatively spliced genes is known to contain a splicing variant at first exon's proximal region [4].

BLASTX was used to identify high scoring local alignments by translating the query nucleotide sequence into all six frames. BLASTX is very effective in detecting statistically significant sequence similarities and, in conjunction with intrinsic methods, may help refine intrinsic evidence and improve the reliability of the predicted coding regions.

Technically, the implementation of the wrapping modules was based on Java. Perl was used to parse, validate and compose XML documents that comply with the specifications of the corresponding schema definitions. Finally, bioperl's graphics library was used to visualize the intrinsic features identified by the *GSD* and *SSD* classes of tools.

4.1 Views Builder

Figure 3 illustrates an example comparative representation of the exons identified in optimal and suboptimal parses of a query genomic sequence corresponding to 2-9 exons of the human

p53 tumor suppressor gene. Setting the cutoff value to 0.10, GENSCAN did not identify any other suboptimal exonic features in this region, while HMMGENE extracted near-optimal gene structures, shown in different tracks (two suboptimal gene parses presented here). It is obvious that none of the optimal parses perfectly matched the actual features, while suboptimal features predicted by HMMGENE correctly identified exons that were missed by optimal transcripts. Similar to *GSDs*, the *Views Builder* can be configured to generate combined depictions of the intrinsic signal features extracted from the three *SSDs*.

4.2 XQuery Processing

With the formulation of specific XQuery expressions the information extracted from the incorporated resources can be filtered or combined into newly formatted XML documents. Examples of XQuery expressions are described in the following.

Example 1: Display matching *GI* elements for high scoring hits (`e-value < k`), where `k` is a user-defined cutoff value.

The XPath expression that is used to define the search path in the *SID* schema tree (Figure 2) can be written as:

```
doc(<src>)/SID/XVIEWS/SIMSEQ/GI[/SID/XVIEWS/SIMSEQ/EVAL < k]
```

Example 2: Join evidence of appropriate sources in order to obtain all terminal exons that are not included in optimal parses and are adjacent to polyadenylation sites detected in the 50 bases upstream/downstream region.

Table 2 on the left column contains the corresponding XQuery expression. The right column illustrates a fragment of the resulting XML document that complies with the tagged elements defined in the query's *return* clause and contains the matching features. Apart from combining and filtering data, XQuery expressions can be used to perform consistency checking of the positional and qualitative properties of the identified features. To its extent XQueries are particularly valuable in cross-validating the resulting information against XML descriptions of external gene annotation data deposited in distributed databanks.

5 Discussion/Conclusions

Although query processing has been efficiently addressed in biological databases integration, there is still a significant gap in integrating and processing evidence data that are dynamically generated by biological analysis tools. The described architecture addresses this issue by integrating evidence from multiple resources into machine-processable documents that can be accessed, queried and transformed into user-friendly depictions. XML was used to describe and interchange data, together with XML-related technologies that offer advanced processing capabilities over the resulting data sources. Technically, the basic obstacle towards seamless integration and information handling was the syntactic and semantic heterogeneities introduced by the design of the incorporated Web-based resources. It is believed though that with the adoption of advanced XML-based technologies, such as RDF (Resource Description Framework)

Table 2: An example FLWR expression (in the left column) resulting in new tagged elements as defined in the *return* clause. The right column contains a fragment of the matching data that were extracted from the corresponding *p53* XML-formatted gene data.

Query Expression	XML Document

<XCONF> {
for $gsd in doc(<src1>)
//PRFEAT[FTYPE=''term''],
$ssd in doc(<src2>)
//PRFEAT[FTYPE=''polyA'']
where ($gsd/PARSE=''suboptimal''
and $gsd/RPOS/STR=''+''
and $ssd/RPOS/STR=''+''
and abs($gsd/RPOS/END -
$ssd/RPOS/START)<50)
return
<FEATURES> {
$gsd/RPOS/START,$gsd/RPOS/END
<POLS>{$ssd/RPOS/START/number()}
</POLS>,
$gsd/SCR,$gsd/PARSE}
</FEATURES> }
</XCONF>
	<XCONF>
	<FEATURES>
	<START>6528</START>
	<END>6618</END>
	<POLS>6617</POLS>
	<SCR>0.133</SCR>
	<PARSE>suboptimal</PARSE>
	</FEATURES>
	</XCONF>

and Web Services, this issue will be addressed more efficiently by assigning self-description capabilities to the resources that are independent of the environment and implementation platform [20].

To our knowledge this is the first integration architecture addressing gene identification that supports also information handling capabilities over the resources outcome. Since no exhaustive evaluation of these methods is available, the proposed architecture to its extent can be used to assess the sensitivity and specificity of the underlying algorithms in an automated fashion. In addition, external links with biological databanks that offer XML views of the deposited data can help increase the reliability of the predicted genes.

References

[1] C. Mathé, M. F. Sagot, T. Schiex and P. Rouzé. Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Res.*, 30(19):4103-4117, 2002.

[2] J. Wang, S. Li, Y. Zhang, H. Zheng, Z. Xu, J. Ye, J. Yu and G. K. Wong. Vertebrate gene predictions and the problem of large genes. *Nat. Rev. Genet.*, 4(9):741-749, 2003.

[3] M. R. Brent and R. Guigo. Recent advances in gene structure prediction. *Curr. Opin. Struct. Biol.*, 14(3):264-272, 2004.

[4] A. A. Mironov, J. W. Fickett and M. S. Gelfand. Frequent alternative splicing of human genes. *Genome Res.*, 9:1288-1293, 1999.

[5] J. M. Johnson, J. Castle, P. Garrett-Engele, Z. Kan, P. M. Loerch, C. D. Armour, R. Santos, E. E. Schadt, R. Stoughton and D. D. Shoemaker. Genome-wide survey of human alternative pre-mRNA splicing with exon junction microarrays. *Science*, 302(5653):2141-2144, 2003.

[6] A. N. Ladd and T. A. Cooper. Finding signals that regulate alternative splicing in the post-genomic era. *Genome Biol.*, 3(11):reviews0008, 2002.

[7] C. Burge and S. Karlin. Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol.*, 268(1):78-94, 1997.

[8] M. Q. Zhang. Computational prediction of eukaryotic protein-coding genes. *Nat. Rev. Genet.*, 3(9):698-709, 2002.

[9] S. Rogic, A. K. Mackworth and F. B. Ouellette. Evaluation of gene-finding programs on mammalian sequences. *Genome Res.*, 11(5):817-832, 2001.

[10] R. Guigo, P. Agarwal, J. F. Abril, M. Burset and J. W. Fickett. An assessment of gene prediction accuracy in large DNA sequences. *Genome Res.*, 10(10):1631-1642, 2000.

[11] J. E. Allen, M. Pertea and S. L. Salzberg. Computational gene prediction using multiple sources of evidence. *Genome Res.*, 14(1):142-148, 2004.

[12] T. Hernandez and S. Kambhampati. Integration of biological sources: Current systems and challenges. *ACM SIGMOD Record*, 33(3):51-60, 2004.

[13] World Wide Web Consortium: Extensible Markup Language (XML) 1.0 W3C Recommendation (Third Edition) (2004) http://www.w3.org/TR/2004/REC-xml-20040204/

[14] F. Achard, G. Vaysseix and E. Barillot. XML, bioinformatics and data integration. *Bioinformatics*, 17(2):115-125, 2001.

[15] K. L. Howe, T. Chothia and R. Durbin. GAZE: A generic framework for the integration of gene-prediction data by dynamic programming. *Genome Res.*, 12(9):1418-1427, 2002.

[16] Y. Huang, T. Ni, L. Zhou and S. Su. JXP4BIGI: A generalized, Java XML-based approach for biological information gathering and integration. *Bioinformatics*, 19(18):2351-2358, 2003.

[17] S. M. Searle, J. Gilbert, V. Iyer and M. Clamp. The otter annotation system. *Genome Res.*, 14(5):963-970, 2004.

[18] World Wide Web Consortium: XQuery 1.0: An XML Query Language (W3C Working Draft) (2005) http://www.w3.org/TR/xquery/

[19] G. Gardarin, A. Mensch, T. Tuyet Dang-Ngoc and L. Smit. Integrating heterogeneous data sources with XML and XQuery. 13th International Workshop on Database and Expert Systems Applications, 839-844, 2002.

[20] L. Stein. Creating a bioinformatics nation. *Nature*, 417(6885):119-120, 2002.