BIOINFORMATICS ORIGINAL PAPER Vol. 22 no. 11 2006, pages 1383–1390
doi:10.1093/bioinformatics/btl081

Systems biology

Graph-based analysis and visualization of experimental results with ONDEX

Jacob Köhler[1,*], Jan Baumbach[2], Jan Taubert[2], Michael Specht[2], Andre Skusa[2], Alexander Rüegg[2], Chris Rawlings[1], Paul Verrier[1] and Stephan Philippi[3]

[1]Division of Biomathematics and Bioinformatics, Rothamsted Research, AL5 2JQ Harpenden, UK, [2]Faculty of Technology, Bielefeld University, Germany and [3]University of Koblenz, Germany

Received on January 4, 2006; revised on February 16, 2006; accepted on March 2, 2006
Advance Access publication March 13, 2006
Associate Editor: Martin Bishop

ABSTRACT

Motivation: Assembling the relevant information needed to interpret the output from high-throughput, genome scale, experiments such as gene expression microarrays is challenging. Analysis reveals genes that show statistically significant changes in expression levels, but more information is needed to determine their biological relevance. The challenge is to bring these genes together with biological information distributed across hundreds of databases or buried in the scientific literature (millions of articles). Software tools are needed to automate this task which at present is labor-intensive and requires considerable informatics and biological expertise.

Results: This article describes ONDEX and how it can be applied to the task of interpreting gene expression results. ONDEX is a database system that combines the features of semantic database integration and text mining with methods for graph-based analysis. An overview of the ONDEX system is presented, concentrating on recently developed features for graph-based analysis and visualization. A case study is used to show how ONDEX can help to identify causal relationships between stress response genes and metabolic pathways from gene expression data. ONDEX also discovered functional annotations for most of the genes that emerged as significant in the microarray experiment, but were previously of unknown function.

Availability: ONDEX is freely available under the GPL License and can be downloaded from SourceForge http://ondex.sourceforge.net/

Contact: Jacob.Koehler@bbsrc.ac.uk

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

Current high-throughput genomics technologies generate large quantities of high dimensional data. Microarray, NMR, mass spectrometry, protein chips, gel electrophoresis data, Yeast-Two-Hybrid, QTL mapping, gene silencing and knockout experiments are all examples of technologies that are used to capture thousands of data points, often in single experiments. Whereas bioinformatics tools exist for extracting and converting raw data from technological platforms into more readily interpretable forms, these tools often lack support for deeper scientific interpretation such as the correlation and combined analysis of experimental data coming from different technological platforms and scientific databases (Durand *et al*., 2003).

To access a wide variety of data in a consistent way, a combination of bioinformatics approaches is needed. Starting from a set of related biological data (e.g. genotype–phenotype interactions, metabolic pathways, gene regulatory networks, etc.) it is possible to generate integrated views of the data by considering them as biological networks. Such networks can then be used to analyze and visualize experimental data using graph-based methods in combination with sequence analysis methods.

Biological data such as metabolic pathways, protein interactions, etc. are best seen as a network or graph. However, biological databases are usually implemented using table centric data structures, which do not readily allow the utilization of graph analysis methods.

Several tools for graph-based visualization and analysis of biological data have been developed. In these programs, the experimental results are visualized as networks and enriched with additional information. For example Cytoscape (Shannon *et al*., 2003), MAPMAN (Thimm *et al*., 2004) and Osprey (Breitkreutz *et al*., 2003) import and visualize individual preselected biological networks. PATIKA (Demir *et al*., 2002) centers around a bespoke ontology of cellular events. All these graph-based systems can import data from different sources, but support for automated linking and mapping of data from different heterogeneous data sources is limited.

For database integration, many tools and applications exist which have been reviewed recently (Köhler, 2004; Stein, 2003). In this paper, we describe the ONDEX framework that combines large-scale database integration, sequence analysis, text mining and graph analysis. At the same time this system can be used for analysis and interpretation of experimental results. The main focus of this publication is on the graph analysis component. We also demonstrate how this integrated approach can reveal new findings that were not uncovered in the original analysis and publication by reanalyzing a recently published microarray experiment.

2 ONDEX SYSTEM OVERVIEW

2.1 Data integration and sequence analysis

The central idea behind data integration in ONDEX is to overcome technical and semantic heterogeneities between different data sources. In practice, this means converting different heterogeneous

*To whom correspondence should be addressed.

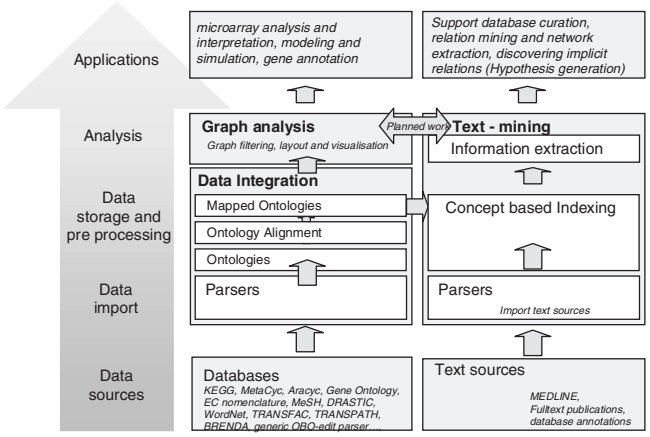

Fig. 1. Overview of the ONDEX architecture and its components.

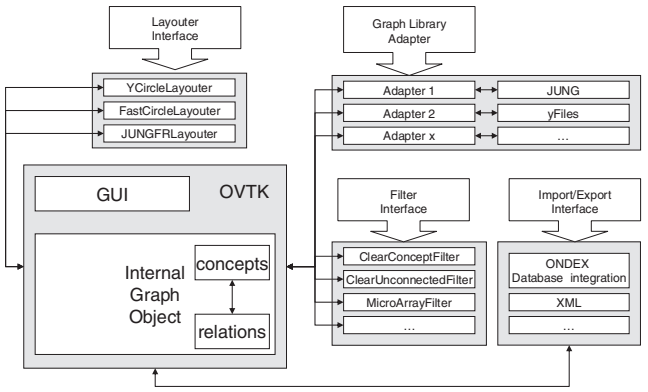

Fig. 2. System architecture of the graph analysis and visualization components of ONDEX.

data sources into a common graph-based data structure (integrated ontologies) in two fully automatic steps (Fig. 1, left hand side).

(1) *Import of databases and ontologies.* About 15 parsers for databases and ontologies have been developed. The DAG-edit and OBO-edit parsers are generic parsers that allow the import of most OBO ontologies. Users who wish to add additional databases, will have to develop a new parser. This normally requires 1–10 developer days, depending on the complexity of the data source.

(2) *Alignment of data from different sources*. ONDEX aligns/maps different data sources by generating a link between equivalent concepts rather than merging equivalent entities into a new one. ONDEX currently supports several methods for the automatic alignment of data sources. These methods use a combined approach based on comparisons of concept names, accession numbers and on structural properties of the ontology. In addition, it is possible to use sequence analysis methods for the alignment of concepts that represent proteins and enzymes. To this end, we re-implemented an improved version of the INPARANOID algorithm and methodology (Remm *et al*., 2001) in Java. The precision and recall of these methods varies. For example, simple accession-based mapping achieves a balanced precision/recall of 100% in organisms which have a good systematic naming convention like in *Arabidopsis*. Other methods that exploit concept names and structural properties of the ontology achieve a precision of ∼95% although they only find 50% of all equivalent concepts. Clearly, the quality of the data integration methods has a very direct effect on the analysis methods which operate on the integrated dataset. A detailed description and a formal evaluation of the performance of the mapping methods are, however, beyond the scope of this paper and will appear in a separate publication.

As a result of these steps, equivalent and related entities from different data sources imported into ONDEX are represented internally in a semantically consistent way using the same ontology-based data structure. The graph-based nature of this data structure is a fundamental prerequisite to the graph-based analysis and visualization of the integrated data sources.

In addition, sequence analysis methods can be used to search all genes and proteins that are integrated in ONDEX. This functionality is especially useful for high quality functional annotations of genes and genomes. More details on the ONDEX component for data integration and the methods used herein are given in Köhler *et al.* (2004).

The installation and use of the data integration methods is still command line driven and requires technical expertise to install, configure and use this component of the ONDEX system. However, no programming is required, and detailed installation guidelines are provided. The next version will be released with an improved installation procedure and it will be possible to initialize integration runs in a more user friendly way.

2.2 Text mining

Unfortunately, most of the digitally available biological knowledge is not stored in databases, but scattered over millions of scientific publications. The text mining component of ONDEX has been developed in a generic way so that it complements the database integration functionality. We have applied it, for example, to extract cell–cell interaction networks from free text, to mine for flaws in ontologies (Köhler *et al*., 2005) and for the development of a Pathogen-Host Interaction database (Winnenburg *et al*., 2006) by supporting the work of database curators.

In the next section we describe in more detail the graph analysis component, which can be used to exploit the data integrated in an ONDEX system.

3 GRAPH ANALYSIS

The complexity of biological processes and the wealth of data needed for the proper consideration of underlying interactions make data visualization and analysis a fundamental prerequisite for the exploration and interpretation of biological data. Therefore, this section describes the graph analysis and the visualization component of ONDEX, including the key requirements, the underlying data structure, as well as the overall system architecture and its implementation (Fig. 2).

Basic requirements

3.1.1 Handling of large graphs Since biological data are often very complex, necessitating the integration of many large datasets,

the back-end data integration component of ONDEX easily results in graphs with several thousand elements. Out of this comes a key requirement for the ONDEX front-end: it must be able to efficiently handle large graph structures.

3.1.2 Support for external graph libraries Since ONDEX makes use of graph-based structures for the representation of biological concepts and their relations, the visualization and analysis of such data can be based on standard graph libraries. Depending on the selected layout, off-the-shelf libraries are able to deal with graphs consisting of up to several thousands of nodes and edges. However, the databases and ontologies that can be integrated with ONDEX may be significantly bigger than this. In an initial evaluation of 23 graph visualization and analysis libraries, we observed very different levels of performance. Another motivation that made support for several libraries a requirement was that different libraries provide different layout algorithms.

3.1.3 Graph filters Since not all biological questions need all the data available from the back-end integration system, the front-end of ONDEX needs to support several levels of data filtering. First of all, it has to be possible to only import relevant subsets of data before they are analyzed in the front-end. For example, users may only be interested in certain species. Once the data are transferred to the front-end, it may need to be filtered further using graph-based methods. Such filters should make irrelevant information invisible, i.e. it should be possible to mask out irrelevant nodes and edges. As a consequence of applying various filters, the size of graphs should decrease significantly, allowing users to effectively separate important from unimportant information.

3.2 Data structures

In this section the data structure used in the ONDEX back-end (Definition 1) and front-end (Definition 2) is specified. In simple terms, the data structure used in the back-end can be seen as a graph, in which concepts are the nodes and relations are the edges. By analogy with the use of ontologies for knowledge representation in computer science, concepts are used as computational representations of real world entities. Relations are used to represent the way in which concepts are related to each other. Furthermore, concepts and relations may have additional properties and optional characteristics attached to them.

DEFINITION 1. *An integrated ontology is a* 12*-tuple O(C, R, CA, CV, CC, RT, P, ca, cv, cc, rt, id) that consists of*

- *a finite, not empty, distinct set of Concepts C(O)*
- *a finite, not empty set of Relations*: $R(O) \subseteq C(O) \times C(O)$
- *a finite set of Concept Accessions CA(O)*
- *a finite, not empty set of Controlled Vocabularies CV(O)*
- *a tree consisting of Concept Classes CC(O)*
- *a tree consisting of Relation Types RT(O)*
- *the additional properties P(O) of an ontology O' consisting of*:

  —*a finite set of Concept Names CN(O)*
  —*a finite set of Sequences SEQ(O)*
  —*a finite set of Structures STR(O)*

- *the function ca which assigns concept accessions to concepts*

  $ca: C(O) \rightarrow \{(ca_1 \times \cdots \times ca_n) | ca_j \in CA(O)\}$

- *the totally defined functions cv, cc, rt that assign CVs, concept classes and relation types to concepts or relations*

  $cv: C(O) \cup R(O) \rightarrow CV(O)$
  $cc: C(O) \rightarrow CC(O)$
  $rt: R(O) \rightarrow RT(O)$

- *the bijective function id which assigns a unique identifier to every concept and every relation with*:

  $id: C(O) \rightarrow \boldsymbol{N}$

- *and the functions def, cn, seq and str that optionally link concept names (terms), definitions, polypeptide or nucleotide sequences and protein structures to concepts*:
- $def: C(O) \rightarrow DEF(O)$
- $cn: C(O) \rightarrow \{(cn_1 \times \cdots \times cn_n) | cn_j \in CN(O)\}$
- $seq: C(O) \rightarrow \{(seq_1 \times \cdots \times seq_n) | seq_j \in SEQ(O)\}$
- $str: C(O) \rightarrow \{(str_1 \times \cdots \times str_n) | str_j \in STR(O)\}$

To layout and display an ontology in the front-end (graph analysis component), the structure given in Definition 1 has to be extended:

DEFINITION 2. *A visible graph G is a* 7*-tuple G(O, CO, colour, size, visibility, x, y) that consists of*:

- *an integrated ontology, O*
- *a finite, not empty set of Colours CO(G)*
- *the functions colour, size, visibility, x and y (coordinates) which affect the way concepts and relations are visualised*:
  —*colour*: $C(O) \cup R(O) \rightarrow CO(G)$
  —*size*: $C(O) \rightarrow \boldsymbol{R}$
  —*visibility*: $C(O) \cup R(O) \rightarrow \{true, false\}$
  —*x*: $C(O) \rightarrow \boldsymbol{R}$
  —*y*: $C(O) \rightarrow \boldsymbol{R}$

Based on the extended Definition 2, a filter and a layout can be defined in a straightforward way.

DEFINITION 3. *A filter is a function f(G)*: $G \rightarrow G'$ *that modifies colour, size and visibility of the graph G*

DEFINITION 4. *A layout is a function g(G)*: $G \rightarrow G'$ *that modifies x and y of the graph G*

Thus, different filters and layouts can be developed that modify how graph G is presented to the user in different application specific ways. The above definitions of visible graphs as well as filters and layouts are the formal basis of graph-based analysis and visualization in ONDEX. The architecture of the components built upon these definitions follows.

3.3 Architecture of the graph analysis component

A generic modular architecture was developed for the graph analysis and visualization front-end of ONDEX. This data structure is centered on a representation of graphs (Internal Graph Object) that

is independent of any graph library. This modular architecture minimizes the overhead of integrating new filter and layout algorithms, and at the same time fulfills the requirement that several graph libraries can be supported. The requirement for handling large graphs is met in the design of the Internal Graph Object, by making use of memory and CPU efficient datatypes and using efficient analysis and layout algorithms which operate efficiently on this large data structure. The Internal Graph Object is equipped with the following interfaces and adapters:

(1) The import and export interfaces are used for data exchange. Data import initializes the Internal Graph Object by transferring data from the relational back-end of ONDEX. The selection of data for import is based on criteria such as source database, concept type (class), species (taxonomy), etc. Data can be exported to XML and to the Petri Net simulator Cell Illustrator (Doi *et al*., 2003).

(2) The layout interface allows access to the layout algorithms available in different graph libraries and applies them to an Internal Graph Object. Since every graph library brings its own data structure for graph representation, implementations of this interface translate the Internal Graph Object into these graph library specific representations. After the application of a layout algorithm to a graph, the resulting coordinates for its elements are transferred back to the Internal Graph Object by the respective interface implementation. Custom layout algorithms also implement the layout interface, but usually work directly on the Internal Graph Object.

(3) The graph library adapter is used to encapsulate different graph libraries. This allows different graph libraries to be used for determining the layout (calculation of x and y coordinates) and for painting (displaying the graphs on the computer screen).

(4) The filter interface provides a common infrastructure for integrating filtering algorithms (see Definition 3 and Section 3.1)

In summary, graph analysis and visualization in the ONDEX front-end works on an Internal Graph Object which may be connected to arbitrary graph libraries as well as layout and filter algorithms by means of several interfaces and adapters. With this architecture a graph is generated from data imported from the ONDEX back-end and subsequently passed to an algorithm, independent of its origin. The results of the application of an algorithm are transferred back into the Internal Graph Object which may then be processed again by the available filter and layout algorithms. In this manner, arbitrary graph analysis and visualization processes are supported in order to provide the user with a wide range of possibilities that can be tailored to specific application scenarios.

3.4 Developing layout and filter algorithms

Specific layout and filter algorithms can be used to exploit the semantically rich information that is held in the database integration component of ONDEX (i.e. the data structure which is defined in Section 3.2). Layouts and filters are used to increase efficiency and to present information in application specific ways that will be more readily understood by users.

An example of such an efficient layout algorithm is the FastCircularLayout. This separates all visible concepts by their concept class and arranges them in discrete circles which are evenly distributed over a given circular area (Fig. 3). In contrast to the layout algorithms available in off-the-shelf graph libraries, the FastCircularLayout can, because of its linear time efficiency, also layout very large graphs consisting of several millions of elements in acceptable times on a desktop PC. This layout is especially useful for a first overview of the quantity and nature of the data i.e. the classes of data (genes, treatments, transcription factors, etc.), and how these elements are related. This layout helps users to decide on appropriate concept classes and relation types to reduce the amount of data by filtering.

An application-specific microarray filter was developed to analyze and interpret microarray data after it had been subjected to a conventional statistical significance analysis to identify the subset of genes that are differentially expressed. The first step of the microarray filter sets the size and color of concepts according to the (log transformed) expression levels given by a microarray result. The concepts to be processed are identified by a comparison between all concept accessions CA(O) and all SpotIDs in the microarray result file as well as the linkage between concepts and their accessions (function *ca*). Afterwards, the MicroarrayFilter shows only the concepts within a user specified connectivity distance (cutOff) from concepts which are contained in the microarray result by recursively expanding concepts. By not expanding relations that are of the same type and direction, the filter algorithm makes sure that not all concepts of two directly connected concept classes are fully exploded. The outcome of this filter operation is an accumulation of highlighted concepts in the context of their expression level and their surrounding neighbourhood (the analysis is stopped if two neighbourhoods are brought into contact). The details of the microarray filter are illustrated with the following description of its pseudo code.

```
mList ← list of MicroArrayResults (An element in the list
is called a spot. Each spot has an id [spot.id] and a level
of expression [spot.expressionLevel].)
cutOff ← n  // n ∈ N, cut off for neighbourhood propagation
downColour ← green  // colour for down regulated elements
upColour ← red  // colour for up regulated elements

function MicroArrayFilter(O, mList, cutOff,
downColour, upColour):
{
  for all c ∈ C(O) with visibility(c) = true do
    visibility(c) ← false

  inBoth ← new list // intersection of concepts and
     microarray
  for every spot ∈ mList do
    for every c ∈ C(O) with spot.id ∈ ca(c)
      inBoth.add(c)
      size(c) ← spot.expressionLevel
      if spot.expressionLevel < 1 do
        colour(c) ← downColour
      else
        colour(c) ← upColour
  for every c ∈ inBoth do
    visited ← new list  // already visited concepts
    propagateNeighbours(c, null, 0, cutOff)
}
```

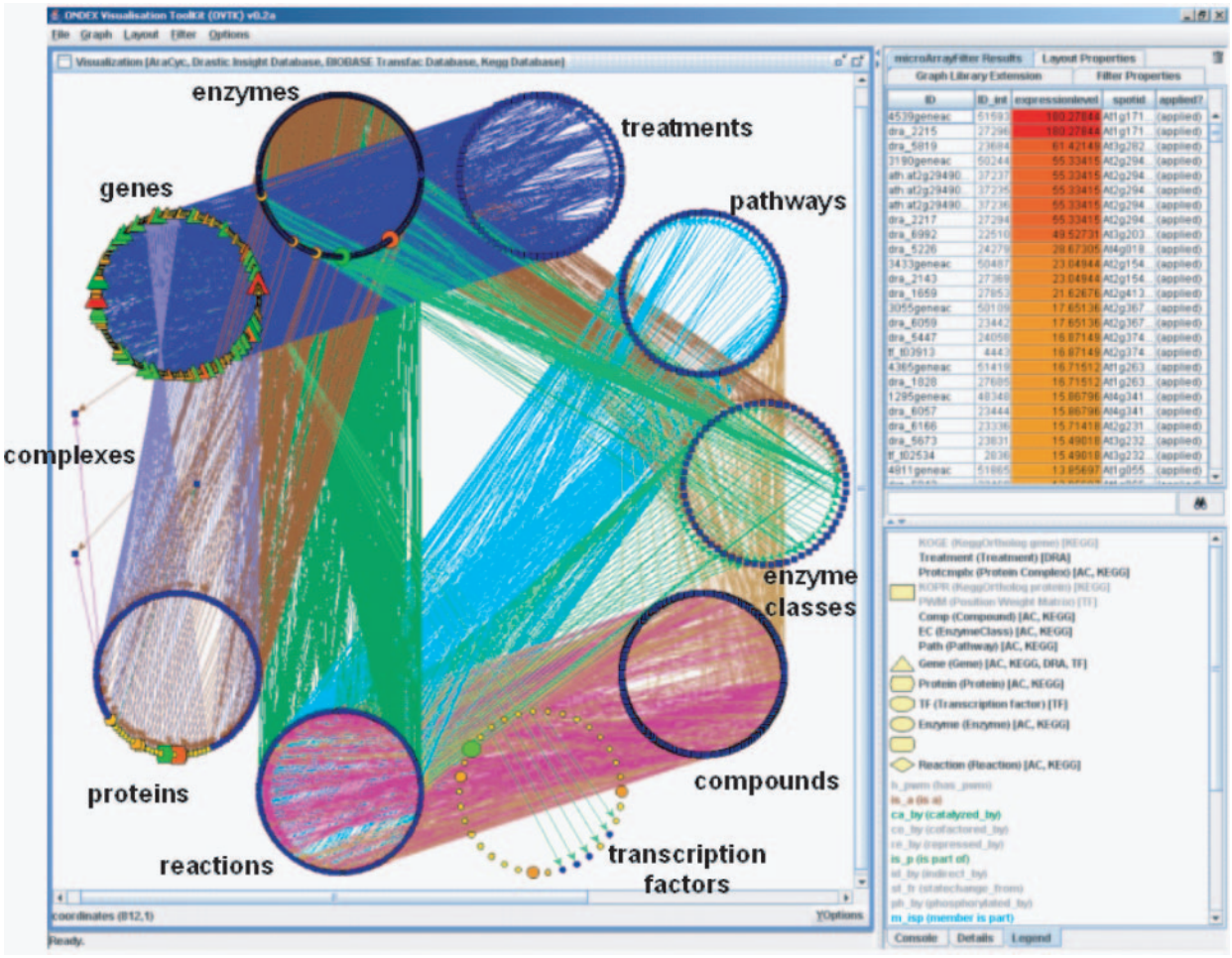

Fig 3. The ONDEX front-end can visualize, filter and analyze microarray results in the context of hundreds of thousands of concepts integrated from several heterogeneous databases (AraCyc, KEGG, Transfac, Transpath and DRASTIC). Concepts from several concept classes (transcription factors proteins and genes) are highlighted according to their expression level in the microarray experiment. The layout was generated using the FastCircularLayout algorithm (see Section 3.4). This layout is especially useful to give a first overview of the quantity and nature of the data. To extract more meaning from the data displayed in this screenshot, users can apply further filters and analysis methods.

```
//type ∈ RT(O), direction ∈ {1,0,−1}
function propagateNeighbours (c, type, direction,
cutOff):
{
  if cutOff>0 do
    visibility(c) ← true

    // all incoming relations pointing to concept c
    // and coming from concept f
    for all r ∈ R(O) with r = (f, c) do
      //do not visit the same concept twice and forbid going
      //back via the relation type and direction used to
          get here
      if c ∉ visited ∧ ¬ (rt(r) = type ∧ direction = −1) do
        visited.add(c)
        propagateNeighbours(f, rt(r), +1, cutOff--)

    // all outgoing relations from concept c
    // and going to concept t
    for all r ∈ R(O) with r = (c, t) do
      if c ∉ visited ∧ ¬ (rt(r) = type ∧ direction = +1) do
        visited.add(c)
        propagateNeighbours(t, rt(r), -1, cutOff-)
}
```

The algorithm has a worst case efficiency of O(m∗n) with m = |mList|, n = |C(O)|. The application of the filter in the context of several large integrated databases such as Aracyc, BRENDA and TRANSPATH only takes a few seconds on a standard PC.

In addition, several other layouts and filters exist which are used in the application described in the next section.

4 APPLICATION: MICROARRAY ANALYSIS WITH ONDEX

To illustrate how the interoperation between the different ONDEX components can be used in a practical data interpretation task, a gene expression experiment was selected as an example.

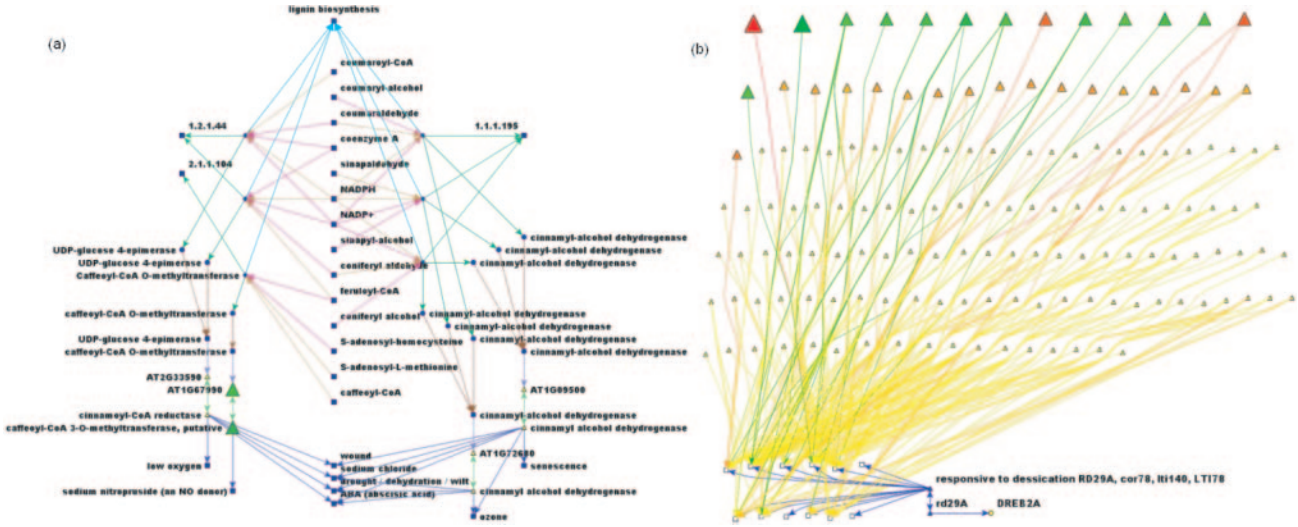

Fig. 4. (**a**) The lignin pathway displayed in the graph analysis component showing only elements where the genes are up- or down-regulated. This figure shows that the four genes At1g67990, At1g09500, At1g72680 and At2g33590 are differentially expressed. They encode several proteins and enzymes which participate in the lignin biosynthesis pathway and fall into three enzyme classes (2.1.1.104, 1.1.1.195 and 1.2.1.44). Data from the DRASTIC database show that these genes respond to different types of stress (ABA, sodium chloride, drought and wound). (**b**) Even though it was not differentially expressed on the microarray chip, according to the TRANSFAC database, rd29A (At5g52310) is known to be regulated by a transcription factor which was also differentially expressed in the microarray experiment. This gene is further annotated with 12 different stress types in the DRASTIC database (bottom left, blue rectangular boxes). These 12 stress types are also known to affect the expression of 120 other genes that were also differentially expressed in the analyzed microarray experiment.

4.1 Methods

We re-analyzed a recently published microarray experiment (Parani *et al*., 2004) in which *Arabidopsis thaliana* has been irrigated with 0.1 and 1.0 mM of sodium nitrosulphide (SNP) after first bolting. SNP is a nitrous oxide (NO) donor. The aim of the experiment was to investigate its effect on early plant development. In their original study, Parani *et al*. (2004) found by statistical analysis that in the set of 342 up-regulated and 80 down-regulated genes there were 126 'novel' genes with unknown functions. In the following, we reanalyzed the subset of statistically differentially expressed genes from the 1.0 mM NO treatment (288 genes). All steps of this analysis are fully automated and only require user interactions for selecting and configuring layouts and filters. The graph analysis can be performed by a computer literate biologist, and no programming is required to perform the analysis as described in this article.

An ONDEX database was built by importing and integrating the following data sources: AraCyc (Mueller *et al*., 2003), KEGG (Kanehisa and Goto, 2000), DRASTIC Insight (Newton *et al*., 2002) TRANSFAC, TRANSPATH (Wingender, 2004) and BRENDA (Schomburg *et al*., 2004). For aligning and mapping these data sources, the accession-based mapping method was applied. This method achieves a precision of almost 100%, by mapping concepts which have the same unambiguous accession number and which fall into the same concept class (i.e. gene, pathway, protein). A relevant *A.thaliana* specific subset of this integrated ontology was loaded into the graph analysis and visualization component (64 085 concepts and 71 210 relations) and visualized using the FastCircularLayout. Subsequently applying the microarray filter reduced the number of concepts and relations significantly and also mapped 259 out of the 288 differentially expressed genes from the 1.0 mM NO treatment to at least one element in the integrated dataset (Fig. 3).

4.2 Data analysis and visualisation

4.2.1 Key pathways In the next step, the SubtreeFilter was applied to calculate a linkage table summarizing all known metabolic pathways with at least one differentially expressed gene according to the microarray experiments. The filter scores each pathway x by dividing the number of differentially expressed genes in x by the total number of genes in x. A total of 55 pathways had at least one differentially expressed gene (parani_reanalysis.xls in Supplementary Material), the pathways cytokinin biosynthesis and glutathione metabolism showed greatest activity. Interestingly, the observation that the jasmonic acid biosynthesis is expressed following NO treatment is not mentioned in the original analysis. However, in their original interpretation Parani *et al*. (2004) also noted that there was considerable activity in the lignin pathway. In order to follow-up this information, we applied the SubtreeFilter which allowed us to narrow down to those reactions which are associated with active genes and related entities (Fig. 4a).

4.2.2 Overexpressed transcription factor but no effect on expected gene As can be seen in Figure 3, only two of the differentially expressed transcription factors are linked to other elements of the integrated dataset. This observation was followed up by filtering out all elements that are not directly or indirectly linked to these two transcription factors. Thus it was noted, that one gene (rd29A = At5g52310) which is under the transcriptional control of one of these two differentially expressed transcription factors, is also known to be associated with the types of stress observed in this experiment (Fig. 4a). Interestingly, another study

(Kreps *et al*., 2002) revealed that this specific gene (At5g52310) is the gene with the largest induction under drought, salt and cold stress. That this gene is not differentially expressed in the study of Parani *et al*. (2004) is surprising, since NO stress seems to be highly correlated with drought, salt and cold stress (see next section).

4.2.3 Key stress genes Using the SubtreeFilter statistics it was possible to identify the key stress genes which were also being seen in the microarray results (see parani_reanalysis.xls in Supplementary Material). We identified 11 treatments (from the DRASTIC database) that were linked to more than 25 genes in the analyzed microarray dataset. Of these treatments, ABA, sodium chloride and drought all relate to water shortage stress and Yariv phenylglycoside treatment relates to cell stress. This possibly indicates that NO is active both in drought response and cell wall repair, which is a novel observation not discussed in the original publication.

4.3 Gene annotation

Parani *et al*. (2004) reported a list of 126 'novel' genes with 'unknown' functions of which 87 are found in the 1.0 NO treatment. In the latest TAIR version, 42 of these genes still have no annotation, 21 are only annotated with a PFAM motif hit, and 22 genes are now at least partly annotated. These 87 'unknowns' from the original analysis were then presented to the graph analysis component. As a result of this operation, various annotations could be assigned to 69 genes. Another 2 'unknowns' and 12 of the graph analysis located genes were annotated by sequence similarity searches against all integrated data sources using PatternHunter (Ma *et al*., 2002) with stringent parameters (*E*-value 10^{-7} which typically results in hits with a sequence identity >40%). See also parani_reanalysis.xls in Supplementary Material.

5 DISCUSSION

The ONDEX framework with its components and interfaces has been described. Its use for the interpretation of gene expression results was demonstrated in a case study. According to Parani *et al*. (personal communication), the analysis results are not only sensible, but also successfully revealed novel findings that could not have been established by the conventional microarray analysis methods used in the original publication. Similar benefits could be obtained for other high-throughput 'omics datasets such as metabolomics data and these are being investigated. The key finding was that the combination of semantic database integration and data visualization established new knowledge that could not have been discovered by data integration, sequence analysis or graph analysis methods alone.

From a technical perspective, the modular and generic architecture of the ONDEX system was shown to be flexible enough to combine a number of graph libraries and filter mechanisms that proved valuable in applications. The data sources were successfully linked and integrated by the ONDEX database integration component, and the graph analysis and visualization methods successfully operated on graphs consisting of hundreds of thousands of nodes and edges.

A particular strength of the approach taken in this publication is that all data that are relevant for a given experiment are extracted from any combination of integrated databases and text sources. This means that unlike in some other related systems, no a priori knowledge and pre-selection of databases is required for the analysis. In practice, however, one normally excludes databases which are obviously irrelevant in order to minimize the computing time for the data integration steps. Another strength is that ONDEX can be used to analyze data from any organism in a species specific way. It is also possible to integrate and exploit pathway information gained from other species, including model organisms such as *Arabidopsis* and mouse. Although one has to be careful when making conclusions across species, inferring information from closely related species is often the only possibility for biologists who do not work on one of relatively well-characterized and fully sequenced model organisms.

When comparing ONDEX with dedicated microarray analysis tools like Acuity, GeneSpring and Spotfire DecisionSite for Microarray Analysis, the difference is that these tools provide comprehensive statistical support and data analysis methods. Pathway and ontology analysis is, however, normally limited to a small number of selected imported pathway maps (most commonly KEGG and GeneMAPP), or to GO-based functional categorizations. Several excellent biological datamining frameworks have been developed in companies or as commercial products that provide functionality similar to that in the ONDEX system. These include systems in VTT Finland (Gopalacharyulu *et al*., 2005), ChipInspector/ Bibliosphere Pathway Edition (from Genomatix), Phylosopher (Genedata) and ExPlain (BIOBASE) and PathwayStudio (Nikitin *et al*., 2003) (Ariadne Genomics).

What these tools have in common is that they can be used to integrate, analyze and visualize data from heterogeneous sources to assist users in interpreting experimental results, including microarray data. A detailed comparison of the functionality and the methods underpinning these tools is difficult, since in many cases, the public documentation is written in very general terms, often focussing on user interfaces. Yet, there is no doubt that many of these commercial frameworks incorporate highly advanced methods and very successfully address similar problems as described in this publication. ONDEX is distinct from these software systems in being available under the GNU Public License and therefore without cost.

We expect that such methods will continue to become more and more important in the future. The often complicated experimental techniques and the advanced statistical analysis required for the generation of 'omics data, has often become routine, and the bottleneck shifts from data generation to interpreting and making sense of the data. Since computational methods such as those described in this article largely depend on the data sources used, it will be the quality, comprehensiveness, consistency and correctness of the underlying databases, ontologies and scientific publications that will become increasingly important for the success and reliability of these methods.

ACKNOWLEDGEMENTS

All authors wish to thank Parani Madasamy, Douglas Leaman and and Stephen Goldman for their valuable comments on the microarray analysis results. A.S. thanks the 'NRW International Graduate School in Bioinformatics and Genome Research' at the Bielefeld University for financial support. A.S. and S.P. gratefully acknowledge funding from the European Science Foundation. This work was funded in part by BBSRC grant BBS/B/13640. Rothamsted Research

receives grant aided support from the Biotechnology and Biological Sciences Research Council.

Conflict of Interest: none declared

REFERENCES

Breitkreutz,B.J. *et al*. (2003) Osprey: a network visualization system. *Genome Biol*., **4**, R22.

Demir,E. *et al*. (2002) PATIKA: an integrated visual environment for collaborative construction and analysis of cellular pathways. *Bioinformatics*, **18**, 996–1003.

Doi,A. *et al*. (2003) Genomic Object Net: II. Modelling biopathways by hybrid functional Petri net with extension. *Appl. Bioinformatics*, **2**, 185–188.

Durand,P. *et al*. (2003) Integration of data and methods for genome analysis. *Curr. Opin. Drug Discov. Devel*., **6**, 346–352.

Gopalacharyulu,P.V. *et al*. (2005) Data integration and visualization system for enabling conceptual biology. *Bioinformatics*, **21** (Suppl. 1), i177–i185.

Kanehisa,M. and Goto,S. (2000) KEGG: kyoto encyclopedia of genes and genomes. *Nucleic Acids Res*., **28**, 27–30.

Köhler,J. (2004) Integration of Life Science Databases. *Drugs Discov. Today: BioSilico*, **2**, 61–69.

Köhler,J. *et al*. (2004) Linking experimental results, biological networks and sequence analysis methods using Ontologies and Generalized Data Structures. *In Silico Biol*., **5**, 33–44.

Köhler,J. *et al*. (2005) Quality Control for Terms and Definitions in Ontologies and Taxonomies. *BMC Bioinformatics*, (under review).

Kreps,J.A. *et al*. (2002) Transcriptome changes for *Arabidopsis* in response to salt, osmotic, and cold stress. *Plant Physiol*., **130**, 2129–2141.

Ma,B. *et al*. (2002) PatternHunter: faster and more sensitive homology search. *Bioinformatics*, **18**, 440–445.

Mueller,L.A. *et al*. (2003) AraCyc: a biochemical pathway database for *Arabidopsis*. *Plant Physiol*., **132**, 453–460.

Newton,A.C. *et al*. (2002) DRASTIC: a database resource for analysis of signal transduction in cells. *BSPP Newsl*., **42**, 36–37.

Nikitin,A. *et al*. (2003) Pathway studio—the analysis and navigation of molecular networks. *Bioinformatics*, **19**, 2155–2157.

Parani,M. *et al*. (2004) Microarray analysis of nitric oxide responsive transcripts in Arabidopsis. *Plant Biotechnol. J*., **2**, 359–366.

Remm,M. *et al*. (2001) Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. *J. Mol. Biol*., **314**, 1041–1052.

Schomburg,I. *et al*. (2004) BRENDA, the enzyme database: updates and major new developments. *Nucleic Acids Res*., **32**, D431–D433.

Shannon,P. *et al*. (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res*., **13**, 2498–2504.

Stein,L.D. (2003) Integrating biological databases. *Nat. Rev. Genet*, **4**, 337–345.

Thimm,O. *et al*. (2004) MAPMAN: a user-driven tool to display genomics datasets onto diagrams of metabolic pathways and other biological processes. *Plant J*., **37**, 914–-939.

Wingender,E. (2004) TRANSFAC, TRANSPATH and CYTOMER as starting points for an ontology of regulatory networks. *In Silico Biol*., **4**, 55–61.

Winnenburg,R. *et al*. (2006) PHI-base: a new database for pathogen host interactions. *Nucleic Acids Res*., **34**, D459–D464.