# GIMS – A Data Warehouse for Storage and Analysis of Genome Sequence and Functional Data

Mike Cornell, Norman W. Paton, Shengli Wu, Carole A. Goble, Crispin J. Miller, Paul Kirby
Department of Computer Science, University of Manchester, Oxford Road, Manchester, M13 9PL, UK

Karen Eilbeck, Andy Brass, Andrew Hayes and Stephen G. Oliver
School of Biological Sciences, University of Manchester, Oxford Road, Manchester M13 9PL, UK

## Abstract

*Effective analysis of genome sequences and associated functional data requires access to many different kinds of biological information. For example, when analysing gene expression data, it may be useful to have access to the sequences upstream of the genes, or to the cellular location of their protein products. Such information is currently stored in different formats at different sites in a way that does not readily allow integrated analyses. The Genome Information Management System (GIMS) is an object database that integrates genome sequence data with functional data on the transcriptome and on protein-protein interactions in a single data warehouse. We have used GIMS to store the Saccharomyces cerevisiae (yeast) genome and to demonstrate how the integrated storage of diverse kinds of genomic data can be beneficial for analysing data using context-rich queries and analyses. GIMS allows data to be stored in a way that reflects the underlying mechanisms in the organism, and permits complex questions to be asked of the data. This paper provides an overview of the GIMS system and describes some analyses that illustrate its use for analysing functional data sets for S. cerevisiae.*

## 1 Introduction

Recent and ongoing experimental developments are making available new genome-wide data-sets of both sequence and functional data. The scale and complexity of these data-sets gives rise to substantial challenges in data management and analysis. Not only do individual data-sets (e.g. on genome sequences, the transcriptome, proteome, and molecular interactions) present bioinformaticians with research issues relating to storage, comparison and presentation, it is also clear that many different analyses will need to make use of multiple, genome-level data-sets if full benefit is to be derived from the recent experimental advances.

This paper is on the development of data management and analysis techniques for use with multiple genome-wide data-sets. The hypothesis is that a full understanding of gene function and interaction requires the integration of different data-sets obtained at each level of genome-wide analysis. This hypothesis is not particularly controversial, and many researchers have carried out analyses that interrelate different kinds of genome-level data. However, although there is some experience in interrelating different genome-level data-sets, few environments have been developed specifically to support integrated analysis of multiple kinds of genomic data. For example, although many of the best known genomic information repositories, e.g. MIPS [13], KEGG [19] or YPD [9], combine many different kinds of genomic data, they tend to emphasize browsing and dissemination of the data they store, rather than analysis.

This paper gives an overview of the data storage and analysis facilities of GIMS, an object database of genome sequence and functional data for *Saccharomyces cerevisiae*. Although our experience to date has principally been with *S. cerevisiae*, work is underway to incorporate other organisms into GIMS, and the designs of the database and the analysis techniques have been carried out with a view to minimizing the genome-specific features of the system.

GIMS has two principal components: (i) a data model that describes genome sequence, transcriptome, protein-protein interaction, mutation and phenotype data, the conceptual models for which are presented in [14]; and (ii) a canned query interface, from which users can execute parameterized requests for information based on analyses conducted over the database. In essence, sequence and functional data are replicated in the GIMS database, where the different kinds of data are described using a carefully designed object model. Once so replicated, developing analyses over the data involves writing programs using the industry-standard Java binding for object databases [3], which provides a close integration of the programming language with the database. This eases the development of

| | Single Genome | Multiple Genomes | Sequence | Function |
|---|---|---|---|---|
| Browse | √ | √ | √ | |
| Visualise | √ | √ | √ | |
| Query | | | | |
| Analyse | | | | |

**Table 1. Typical broad database.**

| | Single Genome | Multiple Genomes | Sequence | Function |
|---|---|---|---|---|
| Browse | √ | | √ | √ |
| Visualise | √ | | √ | √ |
| Query | | | | |
| Analyse | | | | |

**Table 2. Typical deep database.**

| | Single Genome | Multiple Genomes | Sequence | Function |
|---|---|---|---|---|
| Browse | √ | | √ | √ |
| Visualise | | | | |
| Query | | | | |
| Analyse | √ | | √ | √ |

**Table 3. Classification of GIMS.**

analyses, and allows fast access to database data from application programs. The canned query interface runs as an application over the GIMS database. GIMS can be considered a data warehouse because it contains only information that is replicated from other sources and because the organization of the data is specifically targeted at analysis tasks.

The paper is structured as follows. Section 2 describes related work on databases containing genome level data sets. Section 3, describes the structure of the database and the data that has been loaded. Section 4 deals with the interface and methods for browsing the data. Section 5 examines the results of canned queries and the advantages of using GIMS for analysing genome level data. Section 6 presents some conclusions.

## 2 Related Work

Databases in bioinformatics can broadly be classified based on the nature of their content into two categories – *broad* and *deep*. A *broad* database stores essentially a single kind of data, but stores such data from many organisms. Examples of broad databases include Swiss-Prot (for protein sequences), PRINTS (for protein fingerprints) and WIT (for pathway data). A *deep* database focuses on one or a small number of species, but stores many different kinds of data, generally including both sequence and functional data. Examples of deep databases for *S. cerevisiae* are MIPS, SGD and YPD. GIMS is a deep database, in that it stores many different kinds of data from a single organism.

Genomic databases can also be classified based on the tasks that they support. For example, Table 1 provides a high-level classification of typical broad databases. The classification indicates which of browsing, visualising, querying and analysis are supported over sequence and functional data from single or multiple genomes. The presence of a √ indicates that a functionality is supported. Where the broad database stores function rather than sequence data, this changes the locations of the two rightmost √ symbols in the table.

In general, broad databases provide facilities for browsing through the data stored in the database, which are often represented as complex records, and facilities are often provided for interacting with visual representations of relevant data sets (e.g., PRINTS provides a viewer for multiple sequence alignments). However, such systems rarely provide ad-hoc query capabilities, and it is unusual for built-in facilities to be provided that support complex analyses.

The classification used in Table 1 is applied for typical deep databases in Table 2. As for broad databases, the focus is principally on browsing and visualising rather than querying and analysing. Although such systems typically provide simple search facilities over the data, users normally interact fairly directly with the data that is stored, rather than invoking complex computations over the database.

Table 3 shows how the classification applies to GIMS. GIMS does not specifically seek to subsume the functionality of existing databases, many of which do a good job at storing, organising and disseminating biological data. Instead, the emphasis is on the close association of analyses with the stored data, so that users interact with the database principally in terms of the analysis tasks they want to carry out, and not so much in terms of the stored data. Section 5 discusses some of the analyses that are supported as part of the GIMS system.

## 3 The GIMS Database

GIMS has been implemented using the object database POET 6.1 (`http://www.poet.com`). An advantage of object databases is that they permit direct implementation of conceptual models of biological data, thus hopefully providing more intuitive representations of stored data for browsing and programming.

The database schema is divided into three parts, representing the genome sequence, protein-protein interactions and the transcriptome, as described more fully in [14]. The portion of the schema describing genome data is shown in

Figure 4 as a screen shot of part of the GIMS interface. The figure shows the schema using the class diagram notation of the UML (Unified Modeling Language) [1]. Classes are represented by rectangles, lines show the relationships between classes, and numbers and "*"s indicate the numbers of objects that may participate in the relationship. Superclass/subclass relationships are shown using arrows that point from the subclass to the superclass.

When populating the database, the number of data sources from which data is drawn has been minimised in order to reduce the number of possible inconsistencies in the naming and numbering of sequences.

DNA sequences are stored in classes *Transcribed* and *NonTranscribed*. *Transcribed* contains all genome regions encoding RNA sequences. A *Transcribed* region consists of a collection of *TranscribedFragment*s, each of which is either a *SplicedTranscriptComponent* or an *Intron*. Alternate splicing of exons to produce different spliced transcripts can occur, so there is a many-to-many relationship between *SplicedTranscriptComponent* and *SplicedTranscript*. *SplicedTranscript* is the superclass of the mature RNA classes, *mRNA*, *tRNA*, *rRNA* and *snRNA*.

Sequences and related data for open reading frames (ORFs) have been taken from SGD [4] and MIPS [13]. Conceptually, it is incorrect to populate the Transcribed class with ORFs as they are only part of an mRNA transcript. However, the 5' and 3' untranslated regions (UTRs) have yet to be identified in the yeast genome. The database will be updated as better annotation becomes available.

Non-transcribed sequences were obtained from SGD and used to populate the *NonTranscribed* class. These include sequences not defined as ORFs, tRNA and rRNA genes, transposons or their LTRs. Non-transcribed sequences are not comprehensively annotated; only their positions and sequences are given. However, they include important sequences, such as the regulatory sequences governing gene expression and 5' and 3' UTRs of mRNAs, which can be involved in the regulation of translation and mRNA stability.

Predicted protein sequences were taken from SGD. Information regarding these proteins was obtained from MIPS and used to populate the protein attributes schema (see Figure 1). This consists of the classes *ProteinLocation*, *ProteinClassification*, *PrositeMotif*, *ProteinFunction* and *Phenotype*. Each of these classes inherit from the superclass *ProteinAttribute*. There is a many-to-many to relationship between *ProteinAttribute* and *Protein*. For example, a protein can have many prosite motifs and each prosite motif can occur in many proteins.

Transcriptome data has been obtained from the Stanford Microarray Database [15] or generated within the Yeast Group at Manchester. For the results presented in Section 5, four publicly available data sets have been used [6, 5]. The

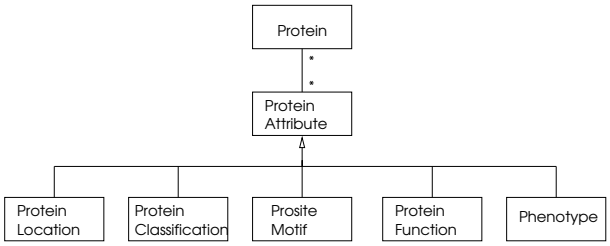


**Figure 1. The protein attributes schema.**

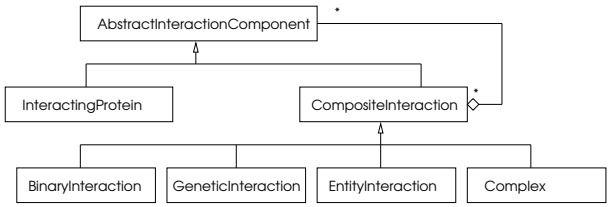


**Figure 2. A portion of the schema describing protein interactions.**

sets are from experiments involving deletion of the *TUP1* gene, over-expression of *YAP1*, sporulation, and diauxic shift. Both un-normalized and log-normalized data were loaded into the database.

Protein interaction information was obtained from public resources such as the MIPS interaction tables and published experiments [17, 11]. The protein interaction schema has been remodeled from that given in [14] to allow discrimination between different types of interaction and also to allow complexes to be assembled recursively from existing interactions. As the schema fragment in Figure 2 shows, the information is modelled using a Composite Design Pattern [8]. This pattern is often used when it is necessary to represent a hierarchy of objects, such as elements in a word-processed document, a graphical user interface, or (in the case of GIMS) the way sets of proteins are assembled via pairwise interactions to form complexes.

The most basic interactions represented by this schema are pairwise. A *BinaryInteraction* represents the result of an experiment that has established a direct contact between the two proteins. A *GeneticInteraction* describes an interaction between two proteins, which does not necessarily involve physical contact. An *EntityInteraction* involves more than two proteins and represents interactions that only occur after there has been a prior interaction. For example, the complement pathway consists of a cascade of interactions that occur sequentially. *EntityInteraction*s can be used to cleanly represent this kind of data. The final interaction is a *Complex*. A complex is represented as a set of interacting proteins.

Since GIMS is implemented using an object database

system, the model in Figure 2 can be directly represented as classes within the database, and those classes can then be made available for use by Java programs. For example, Figure 3 shows Java class definitions for *Chromosome* and *ChromFragment*. Relationships between objects are represented as attributes. For example, the set of *ChromFragment* objects associated with a particular *Chromosome* is represented by the attribute *chromFragments*. Inverses are stored for all relationships.

```
public class Chromosome
{
  private String number;
  private int size;
  public int centromerePosition;
  private Genome fromGenome;
  public ListOfObject chromFragments;
  ...
}

public class ChromFragment
{
  private String name;
  private int position;
  private int start;
  private int end;
  private int size;
  private Chromosome chromosome;
  private Gene gene;
  ...
}
```

**Figure 3. POET Java definition for *Chromosome* and *ChromosomeFragment***

## 4 The GIMS User Interface

Access to the data in GIMS can be obtained by browsing through the database or by using the canned query interface. The GIMS schemas are depicted in windows that are the entry point for browsing the data. Figure 4 shows the schema for browsing genome data. Clicking on one of the classes executes the browsing program. For example, clicking on *Chromosome* pops up the window shown in Figure 5.

The scalar attributes size and number are listed. Further information can be obtained by clicking the *genome* or *chromFragments* buttons, which pop up new windows. In addition, clicking on the *Next*, *Previous*, *First*, or *Last* buttons allows browsing through the other objects in the class.

GIMS can also be accessed using canned queries. A canned query is a parameterised analysis task in which a form is used to obtain input from the user that is of relevance to an analysis program that can be run over the database. The currently supported list of transcriptome canned queries is shown in Figure 6.

Clicking on a query brings up a data entry form that re-

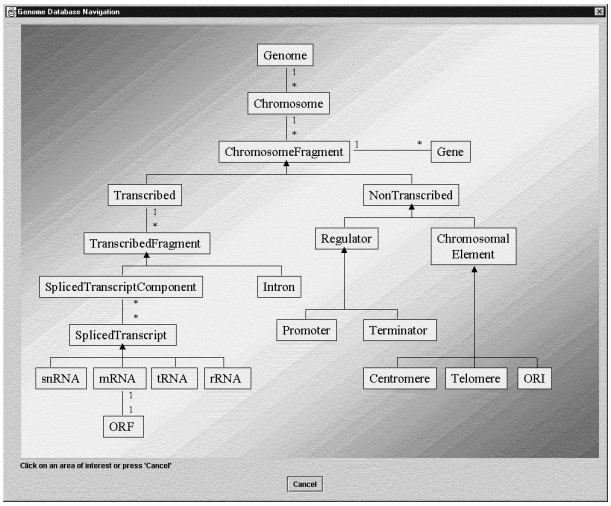


**Figure 4. The class-based browser showing the genome schema.**

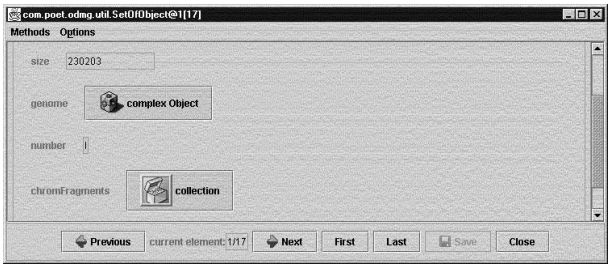


**Figure 5. The form-based representation of an instance of *Chromosome*.**

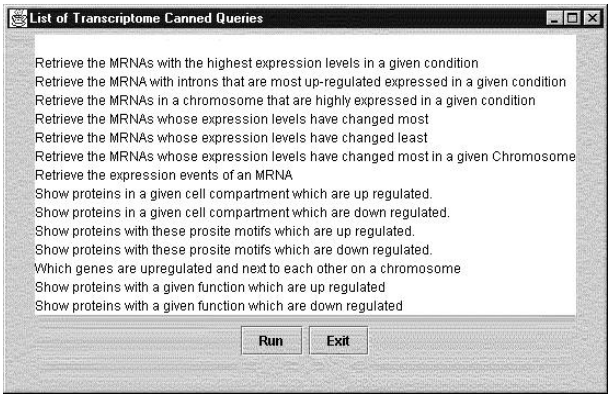


**Figure 6. Transcriptome canned queries.**

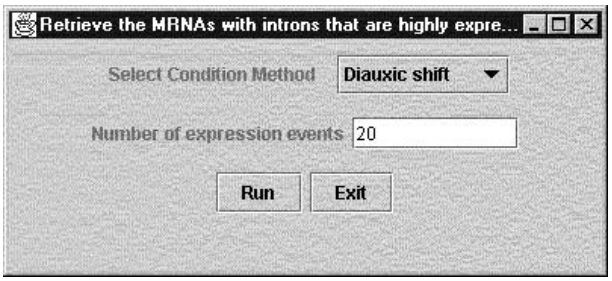


**Figure 7. Example canned query interface.**

quests values for parameters of the query. For example, executing the query "Retrieve the MRNAs with introns that are most up-regulated in a given condition" pops up the window shown in Figure 7, allowing the input of parameters that direct the search.

Inputting 20 into the number of expression events will cause GIMS to return the 20 *mRNA* objects, encoded by intron-containing genes, that show the highest normalized ratio in the diauxic shift experiments.

The visual interface program has been implemented using Java over POET. A three-tier infrastructure has been adopted, with the visual interface running at the client site, a Java RMI (remote message invocation) server in the middle, and the POET database server as the back end. As a result of the three-tier structure, the interface can be run either as an applet or a stand-alone program.

## 5 Analysis

GIMS contains data describing many different attributes of protein and DNA sequences, and associated functional data. Properties relating to sequences include their size, location, motifs and functions. Canned queries can be written that consider any combination of the sequence-based and functional data types in the database.

This allows questions to be asked in a context-rich way. For example, rather than analysing transcriptome data in a spreadsheet and asking which genes are up or down-regulated, it is possible to ask which mRNAs encoding membrane-associated proteins are up-regulated. It is then possible to browse the relevant ratio reading objects and investigate the mRNA, the protein it encodes, and other attributes of the protein. This section presents some illustrative examples of analyses that have been carried out using GIMS, each of which combines different kinds of biological data.

### 5.1 Relating Gene Expression to Gene Structure

The canned query discussed earlier ("What intron-containing genes are up-regulated the most during diauxic shift?"), analyses gene expression in terms of gene structure. The results are shown in Table 4. Introns are comparatively rare in yeast; only around 4% of genes contain them. However, although scarce, this small subset of genes is responsible for over 25% of yeast mRNAs [12]. Many of the yeast introns are in genes encoding ribosomal proteins, and these have very abundant transcripts.

Diauxic shift is the switch from fermentative to respiration metabolism. *COX5b* is involved in the aerobic metabolism of glucose, but it is a hypoxic gene only expressed at low oxygen concentrations [2]. Therefore, as the concentration of glucose in the medium has decreased, the yeast are switching to aerobic metabolism in order to maximise the energy they can obtain from the glucose (respiratory glucose metabolism produces more energy than fermentation). On switching from fermentation to respiration, genes like *COX5b* encoding components of the electron-transport chain are up-regulated.

The upregulation of genes involved in protein degradation and sporulation can also been seen as a stress response.

### 5.2 Relating Gene Expression to Cellular Location

Transcriptome data can be related to the location of gene products within the cell, for example, to identify genes specifying membrane-bound proteins that are up-regulated in *tup1*$\Delta$ cells. *tup1*$\Delta$ is a mutant in which the *TUP1* gene has been deleted.

The protein encoded by *TUP1* is involved in the transcriptional repression of genes when there is glucose in the growth medium [18]. When glucose is present in the growth medium, the expression of genes involved in the uptake and metabolism of other sugars is repressed. GIMS contains 145 genes that specify proteins identified as being located in the plasma membrane, and therefore potentially involved in the transport of sugars and other metabolites. Thirty three of these genes have normalized log ratios of greater than 0.6 (equivalent to being 50% up-regulated) in *tup1*$\Delta$ cells relative to the wild-type (see Table 5). Of these, sixteen encode proteins involved in sugar transport, including hexose transporters, and galactose and maltose permeases.

As well as its involvement in the metabolism of sugars, *TUP1* is involved in other processes, such as the uptake of nitrogen. Three genes encoding proteins that are involved in the uptake of N-compounds, the urea transporter Dur3p (YHL016C), an allantoin permease Dal4p (YIR028W), and Mep2p (YNL142W), are also up-regulated. So too are three genes encoding proteins involved in iron uptake, *SIT1* (YEL065W) which encodes a ferrioxamine B permease [20], *FET3* (YMR058W) which specifies a multicopper oxidase that oxidizes extracellular iron, and a permease gene *FTR1* (YER145C) whose product transports the oxidized

| mRNA | Gene Name | Time Point | Normalized Ratio | Description |
|---|---|---|---|---|
| YIL111W | COX5b | 6 | 2.657956 | Cytochrome-c oxidase chain Vb |
| YEL012W | UBC8 | 7 | 2.295273 | ubiquitin-conjugating enzyme; ubiquitin-protein ligase |
| YBR230C | | 7 | 1.712532 | function not yet known |
| YDL079C | MRK1 | 6 | 1.649205 | MDS1 related protein kinase |
| YDR059C | UBC5 | 7 | 1.486016 | ubiquitin-conjugating enzyme |
| YIL111W | COX5b | 5 | 1.463881 | Cytochrome-c oxidase chain Vb |
| YBL050W | SEC17 | 6 | 1.426066 | peripheral membrane protein required for vesicular transport between ER and Golgi |
| YHR016C | YSC84 | 6 | 1.358348 | SH3 domain in C-terminus |
| YMR133W | FOL3 | 7 | 1.150697 | mRNA is induced early in sporulation |
| YJR079W | | 6 | 1.138584 | function not yet known |
| YMR133W | FOL3 | 7 | 1.150697 | mRNA is induced early in sporulation |

**Table 4. Genes containing introns showing greatest up-regulation during diauxic shift. Descriptions are from SGD. Some genes occur twice because ORFs can be spotted more than once on a chip.**

| ORF Name | Gene Name | Normalized Ratio | Description |
|---|---|---|---|
| YJR158W | HXT16 | 2.506443 | Hexose permease |
| YDL245C | HXT15 | 2.493622 | Hexose transporter |
| YNR072W | HXT17 | 2.419545 | Putative hexose transporter |
| YHR092C | HXT4 | 2.188174 | High-affinity glucose transporter |
| YEL069C | HXT13 | 2.099174 | High-affinity hexose transporter |
| YEL065W | SIT1 | 2.000219 | Ferrioxamine B permease |
| YNL270C | RRN9 | 1.720273 | Basic amino acid permease |
| YDR343C | HXT6 | 1.610511 | Hexose transporter |
| YFL026W | STE2 | 1.521907 | alpha-factor pheromone receptor; seven-transmembrane domain protein |
| YDR342C | HXT7 | 1.511043 | Hexose transporter |
| YMR058W | FET3 | 1.473987 | multicopper oxidase |
| YMR011W | HXT2 | 1.393443 | high affinity hexose transporter-2 |
| YNL192W | CHS1 | 1.216208 | chitin synthase 1 |
| YJL214W | HXT8 | 1.196706 | hexose permease |
| YJL219W | HXT9 | 1.179095 | hexose permease |
| YBR298C | MAL31 | 1.16762 | Maltose permease |
| YIL013C | PDR11 | 1.153586 | Putative member of the ABC family of membrane transporters |
| YNL173C | MDG1 | 1.14507 | Involved in G-protein mediated signal transduction |
| YCL027W | FTR1 | 1.05819 | serinethreonine-rich membrane protein |
| YHL016C | DUR3 | 0.996901 | Urea transporter |
| YER145C | FTR1 | 0.995203 | Iron permease |
| YOR153W | PDR5 | 0.972659 | multidrug resistance transporter |
| YDR345C | HXT3 | 0.964494 | Low-affinity glucose transporter |
| YOL156W | HXT11 | 0.932532 | Glucose permease |
| YGR241C | YAP1802 | 0.86669 | Member of clathrin assembly polypeptide AP180 family |
| YNL142W | MEP2 | 0.818359 | Ammonia transport protein |
| YLR081W | GAL2 | 0.8076 | galactose permease |
| YHR096C | HXT5 | 0.775579 | hexose transporter |
| YLR120C | YPS1 | 0.766352 | GPI-anchored aspartic protease |
| YGR032W | GSC2 | 0.629318 | catalytic component of 1,3-beta-D-glucan synthase |
| YIR028W | DAL4 | 0.523504 | allantoin permease |
| YNR044W | AGA1 | 0.512524 | anchorage subunit of a-agglutinin |
| YHL036W | MUP3 | 0.503629 | very low affinity methionine permease |

**Table 5. Genes encoding membrane-bound proteins up-regulated in tup$\Delta$ cells**

iron into the cell [7].

Tup1p has also been linked to repression of mating [16]. Our search identified *FUS1* (YCL027W) as being up-regulated in *tup1*$\Delta$ cells. *FUS1* encodes a protein involved in cell fusion during mating, has low expression levels in vegetative cells and is up-regulated during mating. Its expression can be induced by exposure to mating pheromone [10].

| ORF name | Gene name | Normalized Ratio | Description |
|---|---|---|---|
| YBR297W | MAL33 | 1.32596 | MAL-activator |
| YBR298C | MAL31 | 1.16762 | Maltose permease |
| YBR299W | MAL32 | 2.40663 | Maltase |

**Table 6. ORFs situated next to each other and up-regulated in tup1$\Delta$ cells.**

### 5.3 Relating Gene Expression to Chromosome Position

One approach is to relate transcription to the relative positions of ORFs. For example, it is possible to identify genes that are located next to each other on a chromosome and that are up-regulated in *tup1*$\Delta$ cells. This question results in 163 mRNAs among which were three neighbouring ORFs on chromosome II: Mal33p (YBR297W) is a maltose fermentation regulatory protein, Mal31p (YBR298C) is a maltose permease and Mal32p (YBR299W) is maltase (see Table 6). All three ORFs encode proteins involved in maltose metabolism, and their close proximity to each other could mean that they share regulatory elements.

### 5.4 Relating Regulatory Sequences to Protein-Protein Interactions

Recently, large-scale yeast two-hybrid screens (e.g. [17, 11]) have identified many pairs of interacting proteins. The interactions can be plotted to form protein interaction graphs, with the proteins as nodes and the interactions between them represented as arcs. However when a screen has identified thousands of interactions, the resulting maps contain complex structures that are difficult to interpret. Because GIMS contains so much background information about proteins and the genes that encode them, it is possible to introduce context to the graphs.

For example, Figure 8 shows part of a protein interaction graph constructed using data from [11]. The nodes can be coloured according to which transcription factors are known to control the expression of genes encoding these proteins. Using the control panel on the left of the figure, the nodes regulated by a particular transcription factor can be set to another colour. In Figure 8, proteins regulated by the transcription factor Hap2p have been coloured so that they appear lighter. The graph shows that many proteins interact with Srp1p (importin). Of the five for which we have data, four are regulated by the same transcription factor Hap2p.

Two proteins which directly interact, Sno1p and Snz1p are also regulated by the same transcription factor Gcn4p, as illustrated at the top of the figure. In this instance, this result is not too surprising, since their open reading-frames are next to each other and they share an upstream region (Sno1p is encoded by YMR095C and Snz1p by YMR096W). Two other pairs of genes in the SNZ/SNO families share this relationship (SNO stands for SNZ proximal ORF).

## 6 Conclusions

The availability of fully sequenced genomes and the development of functional genomics have lead to the generation of genome-level bioinformatics. The vast quantities of complex data that are being generated create great challenges both in the storage and analysis of these data. GIMS represents an alternative to repositories such as MIPS, KEGG and YPD. While each combines different kinds of genomic data, the focus in GIMS is on supporting efficient and effective analysis of data. This is done by replicating sequence and functional data in an object database, making full use of the rich modeling and tightly integrated programming facilities provided. These in turn provide an environment in which it is straightforward to construct canned queries and analyses that are difficult to express using other systems. The change in emphasis from browsing to analysis is also reflected in the interface to the system, which focuses on parameterised analysis tasks implemented as canned queries.

This paper has introduced the GIMS system, and illustrated its use with multiple data sets, including those relating to the genome, transcriptome and protein interactions. The analyses presented show that obtaining insights into the consequences of specific functional data sets is often made easier with reference to other categories of data. The principal aim of the GIMS system is to provide an effective environment within which such analyses can be conducted.

**Acknowledgements** This work has been supported by the BBSRC/EPSRC Bioinformatics Programme. Further development of the GIMS database forms part of COGEME (Consortium for the Functional Genomics of Microbial Eukaryotes), which is supported by the BBSRC's Investigating Gene Function Initiative. Crispin Miller is supported by an MRC Fellowship.

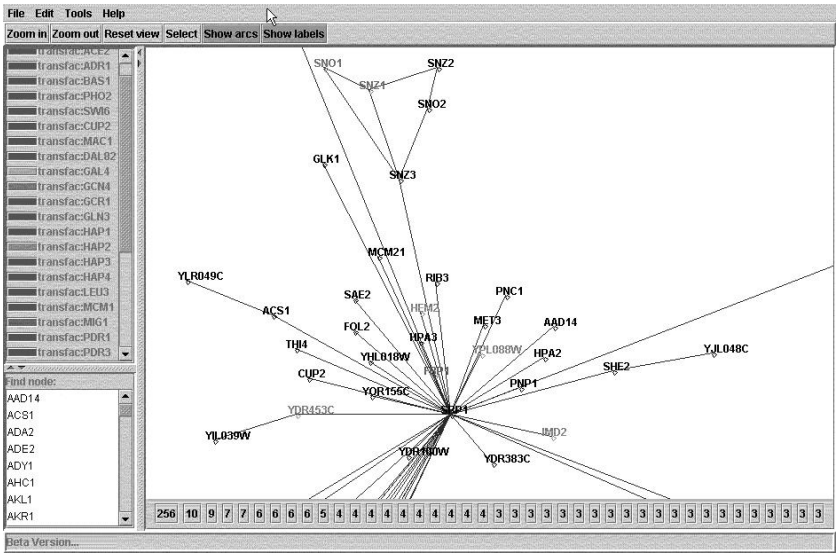


**Figure 8. Screen shot of protein-protein interaction browser.**

## References


[1] G. Booch, J. Rumbaugh, and I. Jacobson, editors. *The Unified Modelling Language User Guide*. Addison-Wesley, 1999.

[2] P. Burke, D. Raitt, L. Allen, L. Kellogg, and R. Poyton. Effects of oxygen concentration on the expression of cytochrome c and cytochrome c oxidase genes in yeast. *J. Biol. Chem.*, 272(23):14705–14712, 1997.

[3] R. Cattell and D. Barry. *The Object Database Standard: ODMG 3.0*. Morgan Kaufmann, 2000.

[4] J. Cherry et al. SGD: Saccharomyces Genome Database. *Nucleic Acids Research*, 26(1):73–79, 1998.

[5] S. Chu, J. DeRisi, M. Eisen, J. Mulholland, D. Botstein, P. Brown, and I. Herskowitz. The transcriptional program of sporulation in budding yeast. *Science*, 282(5389):699–705, 1998.

[6] J. DeRisi, V. Iyer, and P. Brown. Exploring the metabolic and genetic control of gene expression on a genome scale. *Science*, 278:680–686, 1997.

[7] M. di Patti, S. Pascarella, D. Catalucci, L., and Calabrese. Homology modeling of the multicopper oxidase Fet3 gives new insights in the mechanism of iron transport in yeast. *Protein Eng.*, 12(11):895–897, 1999.

[8] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, editors. *Design Patterns*. Addison-Wesley, 1994.

[9] J. Garrels. YPD-A database for the proteins of Saccharomyces cerevisiae. *Nucleic Acids Research*, 24(1):46–99, 1996.

[10] D. Hagen, G. McCaffrey, and G. Sprague. Pheromone response elements are necessary and sufficient for basal and pheromone-induced transcription of the FUS1 gene of Saccharomyces cerevisiae. *Mol Cell Biol.*, 11(6):2952–2961, 1991.

[11] T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, and Y. Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc. Natl. Acad. Sci.*, 98(8):4569–4574, 2001.

[12] M. A. Jr., L. Grate, and M. Pauling. A handful of intron-containing genes produces the lion's share of yeast mRNA. *RNA*, 5(9):1138–1139, 1999.

[13] H. Mewes et al. MIPS: a database for genomes and protein sequences. *Nucleic Acids Research*, 28(1):37–40, 2000.

[14] N. Paton, S. Khan, A. Hayes, F. Moussouni, A. Brass, K. Eilbeck, C. Goble, S. Hubbard, and S. Oliver. Conceptual Modelling of Genomic Information. *Bioinformatics*, 16(6):548–557, 2000.

[15] G. Sherlock et al. The Stanford Microarray Database. *Nucleic Acids Research*, 29(1):152–155, 2001.

[16] D. Tzamarias and K. Struhl. The Stanford Microarray Database. *Functional dissection of the yeast Cyc8-TUP1 transcriptional co-repressor complex*, 369(6483):758–761, 1994.

[17] P. Uetz et al. A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae. *Nature*, 403(6770):623–627, 2000.

[18] F. Williams, U. Varanasi, and R. Trumbly. The CYC8 and TUP1 proteins involved in glucose repression in Saccharomyces cerevisiae are associated in a protein complex. *Mol Cell Biol.*, 11(6):3307–3316, 1991.

[19] J. Wixon and D. Kell. The Kyoto encyclopedia of genes and genomes–KEGG. *Yeast*, 17(1):48–55, 2000.

[20] C. Yun et al. Desferrioxamine-mediated iron uptake in Saccharomyces cerevisiae. Evidence for two pathways of iron uptake. *J Biol Chem*, 275(14):10709–10715, 2000.