Published in final edited form as:

Methods Mol Biol. 2013 ; 939: 183–200. doi:10.1007/978-1-62703-107-3_12.

Data Mining in the MetaCyc Family of Pathway Databases

Peter D. Karp, Suzanne Paley, and Tomer Altman

Bioinformatics Research Group, SRI International, 333 Ravenswood Ave, EK207, Menlo Park, CA 94025

Peter D. Karp: pkarp@ai.sri.com

Abstract

Pathway databases collect the bioreactions and molecular interactions that define the processes of life. The MetaCyc family of pathway databases consists of thousands of databases that were derived through computational inference of metabolic pathways from the MetaCyc Pathway/Genome Database (PGDB). In some cases these DBs underwent subsequent manual curation. Curated pathway DBs are now available for most of the major model organisms. Databases in the MetaCyc family are managed using the Pathway Tools software. This chapter presents methods for performing data mining on the MetaCyc family of pathway DBs. We discuss the major data access mechanisms for the family, which include data files in multiple formats; application programming interfaces (APIs) for the Lisp, Java, and Perl languages; and web services. We present an overview of the Pathway Tools schema, an understanding of which is needed to query the DBs. The chapter also presents several interactive data mining tools within Pathway Tools for performing omics data analysis.

Keywords

Metabolic pathways; pathway databases; systems biology

1 Introduction

Pathway DBs describe the functional landscape of the cell. They collect the bioreactions and molecular interactions that comprise the processes of life. This chapter presents methods for performing data mining on the MetaCyc family of pathway DBs. Most DBs in the MetaCyc family contain metabolic pathway data, although some contain signaling pathway and regulatory information. Databases in the MetaCyc family are created, updated, and queried using the Pathway Tools software [1].

The MetaCyc family consists of thousands of Pathway/Genome Databases (PGDBs) whose contents were derived through computational inference of metabolic pathways from the MetaCyc PGDB [2], followed in some cases by additional manual curation. Curated pathway DBs are now available for a number of model organisms including *Escherichia coli*, *Saccharomyces cerevisiae*, *Arabidopsis thaliana*, *Mus musculus*, and *Homo sapiens*. Table 1 lists the locations of these PGDBs.

The methodology we present is as follows.

- Choose the DB or DBs to be used for your data mining project. The MetaCyc family contains thousands of DBs created by many different researchers.
- Choose among the many data access mechanisms that exist for these DBs:
 - Files in BioPAX format

- Files in attribute-value format
 - Perl API
 - Java API
 - Lisp API
 - Web services API
- Study the Pathway Tools schema to understand what data fields to query.
 - Use interactive data mining tools provided by the Pathway Tools software to analyze data of interest.

Pathway Tools has many additional capabilities that are not presented here including tools for identification of dead-end metabolites and metabolic choke points, and tools for inferring pathway hole fillers and operons. For more details, see [1].

2 Materials

For web access to PGDBs, we recommend use of FireFox, Safari, or Chrome. Versions of Internet Explorer released prior to 2011 are not recommended because of their slow speed and other problems.

BioCyc PGDBs are freely and openly available to all users, and are available in a variety of file formats listed at [3].

The Pathway Tools software is available for the Linux, Windows, and Macintosh platforms. It is freely available to academic users, and is available to commercial users for a fee. BioCyc and Pathway Tools may be obtained from [4].

3 Methods

3.1 Choose the Database(s) for Your Data Mining Project

The following factors should be considered when choosing a PGDB for a data mining project.

Does a PGDB exist for the organism of interest? The websites listed in Table 1 contain thousands of PGDBs that overlap to some degree. If a PGDB does not exist for an organism of interest, a researcher can create a new PGDB using Pathway Tools.

Does a curated PGDB exist for the organism of interest? Curated DBs are generally significantly more accurate than uncurated DBs (note that most pathway DBs outside the MetaCyc family are uncurated). Curated DBs contain minireviews that summarize the biology of a gene or pathway. They contain citations to the literature, and they contain curated data that cannot be predicted computationally (such as enzyme inhibitors and cofactors). The curation level of a PGDB can be assessed by invoking the command *Tools* → *Comparative Analysis*, and then selecting the Organisms report for the PGDB of interest. The resulting report identifies the number of publications cited in the PGDB, and the number of genes and pathways with experimental evidence. In some cases, a curated PGDB contains a large and unique collection of data. For example, EcoCyc contains extensive curated information on the regulation of *E. coli* genes by a variety of mechanisms including transcriptional regulation, regulation by attenuation, and regulation by small RNAs.

Depending upon how you plan to access the PGDB for data mining, a local copy of the PGDB may be required for use within a local copy of Pathway Tools. Accessing PGDB data

using web services does not require having a local copy. Accessing PGDB data using the Java, Perl, and Lisp APIs does require having a local copy, because those APIs are accessed from within the Pathway Tools software using a local Unix socket. Pathway Tools must load the PGDB into its virtual memory.

A local copy of a PGDB can be downloaded via the PGDB Registry [5], which is a peer-to-peer DB server. The Pathway Tools command *Tools* → *Browse PGDB Registry* will list all PGDBs defined within the registry. The user can click on a PGDB to download it locally for use within Pathway Tools. If a known PGDB of interest is not within the registry, contact its authors to request that they deposit it there.

3.2 Choose a Data Access Mechanism

Three different approaches can be used to obtain and operate on PGDB data: 1) download the full data files from the BioCyc website and operate on them locally, 2) use the web services interface to retrieve selected data, or 3) install the Pathway Tools software on a local machine and use one of the programmer APIs.

3.2.1 Data Files—Data files can be downloaded from the BioCyc website by following the instructions at [4]. For PGDBs hosted elsewhere, contact the administrators of that site directly. The downloadable archive for a given PGDB contains all relevant files for that PGDB, including BioPAX files, Attribute-Value files, sequence files, and various tabular files. If the Pathway Tools software is installed locally, these files can also be generated using the **File**→**Export** command.

BioPAX Files: BioPAX [6] is an XML-based standard format that facilitates the exchange of pathway-related information. A number of software tools can import or use data in BioPAX format. Two versions of the BioPAX format file are provided for each PGDB: a Level 2 file, which conforms to an older BioPAX version, and a Level 3 file, which uses the most recent BioPAX version (the two versions of BioPAX are incompatible). A single file contains data for all pathways and metabolic and transport reactions in the PGDB — including all metabolites, enzymes, and regulators — relevant to those pathways and reactions. It does not contain information about genes, transcription units, or transcriptional or translational regulation. To obtain BioPAX data for just a single pathway rather than for all pathways, use the web services interface instead.

Attribute-Value Files: Attribute-value files contain data in a format that corresponds closely to the Pathway Tools schema. A different file is provided for each class of data object, such as genes, proteins, or reactions. For each Pathway Tools object (such as a single gene), the file will list its unique identifier and the values of its attributes, with each attribute on a different line. More details can be found at [7]. Users may have to parse and integrate data from multiple files to extract the information they want. For example, to determine the reaction catalyzed by an enzyme, search the *proteins.dat* file for the value of the *CATALYZES* attribute of the enzyme (which will give the id for the catalysis object, known as an *enzymatic-reaction*), search for that value in the *enzrxns.dat* file to extract the *REACTION* attribute, and then search for that reaction in the *reactions.dat* file to retrieve properties of the reaction, such as its reactants and products.

Other Files: The file archive for a PGDB also contains other miscellaneous files. FASTA files contain the sequence for each gene or protein in the PGDB in FASTA format. Tabular data files summarize selected commonly used relationships between objects, such as genes of a pathway or reactions of an enzyme, in tab-delimited table format. This format can be more convenient than having to extract and integrate information from multiple attribute-

value files if those are the relationships you are interested in retrieving. The full list of files can be found at [7].

3.2.2 Web Services—Data can be retrieved from the BioCyc website (or from any other website that runs version 14.5 or higher of the Pathway Tools software) in XML format using the web services API. Simple queries can return the data for a single Pathway Tools object (such as a gene or a reaction) given its unique id. More complex queries can be constructed using the BioVelo Query Language to retrieve data for one or more objects based on their properties. For a detailed description of the web services API, including sample queries, see [8].

Pathway data for a single pathway can be optionally retrieved in BioPAX format. All other requests through the web services API generate data in a format known as **ptools-xml**. This format is closely related to the underlying Pathway Tools object representation, so an understanding of the Pathway Tools Schema is necessary in order to interpret the data. The ptools-xml format is further described at [9].

To retrieve data for a single object (such as a gene or reaction), you must supply both the PGDB identifier (a short string that uniquely identifies the organism DB) and the object identifier. Objects can be retrieved in either low or full detail. Low detail includes only selected attributes and relationships. For example, low detail for a protein or RNA includes only its name and synonyms, and links to its gene, its components (if a complex) or complexes, and any reactions it catalyzes. Full detail (the default) includes all information associated with that object in the DB, including but not limited to textual summaries, citations to the literature, links to other DBs, reactions in which the object participates, and regulation information.

The URL to retrieve an object from the BioCyc website in ptools-xml format is [http://websvc.biocyc.org/getxml?\[ORGID\]:\[OBJECT-ID\]](http://websvc.biocyc.org/getxml?[ORGID]:[OBJECT-ID]) or [http://websvc.biocyc.org/getxml?id=\[ORGID\]:\[OBJECT-ID\]&detail=\[low/full\]](http://websvc.biocyc.org/getxml?id=[ORGID]:[OBJECT-ID]&detail=[low/full]), where [ORGID] and [OBJECT-ID] are the PGDB identifier and the object identifier, respectively.

The BioVelo query language [10] allows you to write precise queries to extract a set of objects from one or more PGDBs that satisfy specific criteria. Some example searches might be for all proteins whose name includes some search string, all heteromultimeric complexes in the PGDB, all genes associated with a particular pathway, or all compounds in a particular molecular weight range. Query results can be returned at either low detail (the default), full detail, or no detail. In the no detail case, the query returns unique ids only, without any additional attributes, for objects that satisfy the query.

The URL to issue a BioVelo query to the BioCyc website that returns a list of objects in ptools-xml format is [http://websvc.biocyc.org/xmlquery?\[QUERY\]](http://websvc.biocyc.org/xmlquery?[QUERY]) or [http://websvc.biocyc.org/xmlquery?query=\[QUERY\]&detail=\[none|low|ful\]](http://websvc.biocyc.org/xmlquery?query=[QUERY]&detail=[none|low|ful]), where [QUERY] is a properly escaped BioVelo query string that returns a single list of Pathway Tools objects. For example, the request [http://websvc.biocyc.org/xmlquery?\[x:x<-ecoli^^pathways\]](http://websvc.biocyc.org/xmlquery?[x:x<-ecoli^^pathways]) retrieves data for all pathways in EcoCyc. The request [http://websvc.biocyc.org/xmlquery?\[x:x<-mtbrv^^Genes,x^left-end-position>100000,x^right-end-position<200000,%23\[pp:pp<-x^product,pp^molecular-weight-kd>=50,pp^molecular-weight-kd<=60\]>0\]](http://websvc.biocyc.org/xmlquery?[x:x<-mtbrv^^Genes,x^left-end-position>100000,x^right-end-position<200000,%23[pp:pp<-x^product,pp^molecular-weight-kd>=50,pp^molecular-weight-kd<=60]>0]) retrieves data for all genes in the *M. tuberculosis* H37Rv PGDB whose map position on the chromosome is between 100,000 and 200,000, and that have a product whose molecular weight is between 50 and 60 kdaltons.

BioVelo can be a tricky language to master, but several example queries are provided at [8]. In addition, BioVelo is the engine underlying a number of the search mechanisms available from the Search menu on the BioCyc website, including the Compound, Genes/Proteins/RNAs, Reactions, Pathways, and Advanced search forms. Each search conducted using these forms also returns the BioVelo query that was evaluated, so some experimentation with these can provide guidance when generating your own BioVelo queries (but bear in mind that these query forms often generate multi column tables, whereas the web services API is more limited and supports only queries that return a single column).

3.2.3 Lisp, Perl and Java APIs—If the Pathway Tools software has been installed on a local machine, programs can be written that query the data directly through the Lisp, Perl or Java APIs. A wealth of information and relevant links can be found at [11], including the list of API functions, example queries, download links for PerlCyc and JavaCyc, and general information about Lisp and the Lisp debugger. It is strongly recommended to explore these resources before beginning to use any of the APIs, so this discussion will provide just a general overview of the APIs.

There are two layers of API functions. The bottom level is the basic object access protocol, known as the Generic Frame Protocol (GFP) [12], which includes operations for enumerating all objects in a class and accessing the attributes (slot values) of an object. These operations require precise knowledge of the Pathway Tools schema in order to utilize them effectively. On top of that are a number of higher-level functions, known collectively as the Pathway Tools API, that encapsulate some of the biological knowledge and relationships between objects, such as how to query the pathways containing a gene, the genes regulated by a given gene, or whether a protein is an enzyme or a transporter. Your programs will probably make use of functions from both layers.

Lisp API: The Pathway Tools software is written in the Common Lisp language, so when writing a new program starting from scratch (as opposed to interfacing with an existing Perl or Java program), the Lisp API is the most convenient API to use. There is no separate package to install, and no interprocess communication to worry about. Learning enough Lisp to write simple queries is quite straightforward (some good resources for learning Lisp are [13] and [14]), and will greatly increase your productivity when working with the data.

To use the Lisp API, start Pathway Tools in Lisp mode. On a Linux or Macintosh machine, this is done by supplying the `-lisp` command-line argument when starting Pathway Tools. On a Windows machine, a separate desktop icon is provided to start in Lisp mode. In either case, the normal Pathway/Genome Navigator interface will not appear. Instead, you will see a lisp prompt in the console window, which indicates that the software is ready to accept commands. Commands and code snippets can be either typed or pasted directly to the lisp prompt, or they may be loaded from a separate file. Commands and code are interpreted as they are entered and, in case of errors, can be debugged interactively. Some example queries can be found at [15].

PerlCyc and JavaCyc: The Perl and Java APIs are known as PerlCyc [16] and JavaCyc [17], respectively, and must be downloaded as separate modules. Unix file sockets are used for communication between the process running the PerlCyc or JavaCyc program and the Pathway Tools server, which leads to several important limitations: (1) the Java and Perl APIs work only with a Unix installation of Pathway Tools (Linux or MacOS), not with Windows, (2) the machine running the Java or Perl program must be the same as, or share, a file system with the machine running Pathway Tools, and (3) only one external process may interact with a Pathway Tools process at a time.

To use PerlCyc or JavaCyc, Pathway Tools must have been invoked with the `-api` command line argument. When invoked in this fashion, no change is noticeable in the operation of the Pathway Tools program — the Pathway/Genome Navigator appears as usual, and users can interact with it in the normal fashion. The only difference is that simultaneously the program is alert to connections from an external process.

The external Perl or Java program must load the PerlCyc or JavaCyc module, and then create a new `perlcyc` or `javacyc` object to connect to the desired PGDB. Perl functions and Java methods are available that are analogous to most of the corresponding GFP and Pathway Tools API Lisp functions. When one of these functions is invoked on the PerlCyc or JavaCyc object, the function is converted to a Lisp query and submitted to the Pathway Tools process. The subsequent response is then converted back into the appropriate datatype in Perl or Java. Examples showing exactly how to write JavaCyc or PerlCyc queries are included in the appropriate module documentation and in Figure 1. Note that the JavaCyc interface does not support passing of actual objects between Java and Pathway Tools, merely object identifiers. An additional JavaCyc query would be necessary to retrieve some attribute of a returned object.

3.3 Study the Pathway Tools Schema

PGDB data is stored in an object management system called Ocelot [18]. Objects are organized into a class/instance hierarchy. Each class can have its own set of attributes, known as slots. A slot can have one or multiple values, and either describes an attribute of the object (such as the name or molecular weight of a protein) or defines a relationship between that object and some other object in the DB (such as the gene for a protein, or the complexes of which a protein is a component). Some of the major classes in the Pathway Tools Schema are listed in Table 2.

3.4 Use Interactive Data Mining Tools

Pathway Tools includes powerful interactive tools for analyzing data, including enrichment/depletion analysis, visualization tools for high-throughput experimental data, and comparative analysis.

3.4.1 Enrichment Analysis—Consider the analysis of a gene expression experiment in which 200 genes are found to be significantly up or down regulated. Biologists frequently want to ask whether those 200 genes contain significant numbers of genes involved in one or more biological processes (such as cell division), or in one or more biological pathways. That is, are genes from one or more processes statistically overrepresented in that set of genes? Put another way, is the given set of genes enriched for genes from one or more processes or pathways? Similarly, in analysis of metabolomics data one might ask whether a set of metabolites observed to have changed between two experiments is enriched with respect to the metabolites in one or more metabolic pathways.

Enrichment analysis is a statistical analysis tool that is able to answer this type of question. Gene or metabolite lists such as those in our examples are usually generated as a result of a high-throughput experiment. High-throughput experiments are noisy and genes and compounds can participate in multiple biological processes or pathways. So in the context of the above example it is a mistake to take all the pathways in which at least one gene from a list of 200 genes is involved and assume that they all participate in the phenomenon studied in the gene expression experiment. Enrichment analysis enables us to statistically distinguish the pathways explaining the phenomenon that underlies the expression experiment from those that contain genes from the list by accident.

Enrichment analysis was initially described [19, 20, 21] for lists of genes obtained using microarray experiments and for Gene Ontology (GO) terms. We have developed a general framework for enrichment analysis. It is tied in to our Object Groups functionality to allow facile creation of groups of genes, compounds, or other types of objects, and then evaluating whether their enrichment or depletion of GO terms, pathways, or other sets is statistically significant.

3.4.2 Omics Viewers—Pathway Tools Omics Viewers use the Cellular, Regulatory and Genome Overviews to illustrate the results of high-throughput experiments in a global metabolic and genomic context.

Omics Viewers can show absolute data values (such as the concentration of a metabolite or protein, or the absolute expression level of a gene) and can be used to compare two sets of experimental data by computing a ratio and mapping the ratios onto a color spectrum. Multiple sets of experimental data can be superimposed on the same overview diagram so that users can, for example, combine gene expression and metabolomics in the same figure, or view the results of two different microarray experiments together. When combining multiple datasets, users should be careful to assign color schemes that avoid ambiguity. For example, you might want to use “warm colors” like yellow and red for one dataset, and “cool colors” like blue and purple for a different dataset, to allow them to be seen side by side.

Superposition of multiple sets of experimental data on the Overviews can also be animated to show, for example, how gene expression levels of enzymes change with time over the course of an experiment. The animation can be exported to HTML so that it can be published online.

After displaying Omics data on one of the Overviews, navigating to any pathway display will show the Omics data superimposed on the individual pathway. If a particular reaction step has multiple isozymes, then rather than just choosing one value as is done on the Cellular Overview, all values are shown.

Cellular Overview Omics Viewer: The Cellular Overview diagram is a representation of all metabolic pathways and reactions, signaling pathways, membrane proteins, and transporters defined for the current organism. In this diagram, each icon (e.g., circle, square, ellipse) represents a single metabolite. The shape of the icon encodes the chemical class of the metabolite. Each thick line in the Overview diagram represents a single reaction. Neither the icons nor the lines are unique in the sense that a given metabolite or a given reaction may occur in more than one position in the diagram. If there are any thin gray reaction lines, these represent reactions for which no enzymes have been identified in the PGDB — in other words, pathway holes.

The Cellular Overview Omics Viewer can be used to illustrate an even wider range of high-throughput experimental results in a global metabolic pathway context (Figure 3). Genes (in the case of a gene expression experiment) and proteins (in the case of a proteomics experiment) that are involved in metabolism are mapped to reaction steps in the Cellular Overview, and the range of data values in a given experimental dataset is mapped to a spectrum of colors. Reaction steps in the Cellular Overview are colored according to the corresponding data value. Similarly, for metabolomics experiments, compound nodes are colored according to the data value for the corresponding compound. This facility enables the user to see instantly which pathways are active or inactive under some set of experimental conditions.

The Cellular Omics Viewer can be used for:

Microarray Gene Expression Data: Reaction lines (and protein icons, where present) are color-coded according to the relative or absolute expression level of the gene that codes for the enzyme that catalyzes that reaction step. The Cellular Omics Viewer allows a scientist to interpret the results of gene-expression experiments in a pathway context.

Proteomics Data: Reaction lines (and protein icons, where present) are color-coded according to the concentration of the enzyme that catalyzes that reaction step.

Metabolomics Data: Compound icons are color-coded according to the concentration of the compound.

Reaction Flux Data: Reaction lines are color-coded according to reaction flux values.

Other Experimental Data: Any experiment, high-throughput or otherwise, in which data values are assigned to genes, proteins, reactions or metabolites, can be viewed in a pathway context using the Omics Viewer.

Genome Overview Omics Viewer: The Genome Overview Omics Viewer can map any dataset that focuses on genes (such as a gene expression studies) onto the full genome of the organism, using a spectrum of colors to display the numerical values associated with each gene (Figure 4).

The Genome Overview shows in one screen all the genes in an organism's genome, as well as additional information about their transcription units and products. The Genome Overview has several key differences from the Genome Browser. Unlike the Genome Browser, the Overview is not to scale, nor does it reflect spacing between genes. Conversely, the Genome Overview shows the full genome of an organism at once, even if that genome contains multiple chromosomes or plasmids. Each individual replicon (chromosome, plasmid) is displayed on the page with an appropriate identifying label.

The Genome Overview uses iconography similar to that of the Genome Browser, showing the direction of gene transcription with a sloping line on the top (protein-coding gene) or on the bottom (RNA-coding gene) of the gene. Lines underneath genes indicate the extent of transcription units, and are particularly useful for identifying multiple promoters. Transcription units are also indicated by shared gene color, although this coloring is replaced if you choose to map expression data onto the Overview using the Genome Overview Omics Viewer function.

You can identify a gene in the overview by clicking on it to go to its gene page, or by mousing over it to display its gene name, product, and distance from neighboring genes at the bottom of the screen.

The Genome Overview Omics Viewer can be used for

Microarray Gene Expression Data: Genes are color coded according to the relative or absolute expression level of the gene.

Other Experimental Data: Any experiment, high-throughput or otherwise, in which data values are assigned to genes, can be viewed using the Genome Omics Viewer. One such possible use is the mapping of a set of ESTs that have been assigned to genes onto a sequenced genome, thus offering a view of how much of, and which parts of, the genome are covered by that EST set.

Regulatory Overview Omics Viewer: The Regulatory Overview Omics Viewer, like the Genome Overview Omics Viewer, can display gene- or protein-oriented data. It utilizes the Regulatory Overview to display experimental data in the context of a PGDB's regulatory network.

The Regulatory Overview enables the user to visually analyze the regulatory relationships between genes for a specific organism. These relationships are based on the regulatory data available in the PGDB of the organism. Currently, the relationships are based on transcriptional regulatory data (future versions may cover other types of regulation). The Regulatory Overview is represented as a network with nodes and arrows (i.e., arcs). Each node represents a gene of a specific organism. An arrow extends from gene A to gene B if and only if A regulates B.

The Regulatory Overview initially shows the transcriptional regulatory network for the currently selected organism, in one window, without any arrow relationships shown. Otherwise, their great number would clutter the overview. These arrows can be selectively added by using highlighting commands. Each node icon in the diagram, such as a plus sign or circle, depicts one gene. Not all genes in the genome are shown in the diagram. The genes shown are regulators (transcription factors and sigma factors) and all other genes for which the PGDB encodes regulatory information for the gene.

Two network layouts are available: three nested ellipses and top to bottom rows. Three nested ellipses is the default layout when displaying the entire overview for the first time. Top to bottom rows is the default layout when redisplaying only the highlighted genes.

For the three nested ellipses layout the genes are partitioned into three groups, each group being laid out on a separate ellipse. The two inner ellipses contain all the genes that regulate at least one gene. The innermost ellipse contains the genes that regulate the most. Typically, about 15% of the regulators are in the innermost ellipse. The outermost ellipse contains genes that are regulated but that do not regulate.

The outermost ellipse is further partitioned into groups of genes such that all genes within one group are regulated by the same set of genes. We call these groups *multi-regulons*. Some groups are shown as a line or as a line leading to a triangle, perpendicular to the outermost ellipse. Note that although all genes within a multi-regulon respond to the same set of regulator genes, different genes in the group may be controlled in different ways. For example, consider a multi-regulon, comprising genes A and B, that are regulated by genes X and Y. Genes X and Y might both activate the transcription of A, but they might both inhibit transcription of B.

It is possible to display a regulatory subnetwork of a specific organism by doing a series of highlighting operations, and then using the command Redisplay Highlighted Genes Only. This command will create a new, smaller layout of the regulatory network containing only the genes that are highlighted. Genes that do not regulate, or are not regulated by any highlighted genes, are not included in the subnetwork. Further operations can be done on this subnetwork as for the complete overview.

The Regulatory Overview Omics Viewer illustrates the results of high-throughput experiments in the context of gene regulation. Genes that are involved in regulation are mapped to their gene icon in the Regulatory Overview diagram, and the range of data values in a given experimental dataset is mapped to a spectrum of colors. This facility enables the user to see instantly which genes are active or inactive under some set of experimental conditions.

The Regulatory Overview Omics Viewer is very similar to the Cellular Overview Omics Viewer. The data file can utilize the names or DB identifiers of genes, proteins, reactions, EC numbers, or compounds in the first column. The data file supports having a mix of objects specified on separate lines (for example, you can specify metabolomic data in one set of lines in the data file, and microarray data in a different set of lines in the same file).

Omics Graphing: When an experimental dataset includes multiple individual experiments, such as in a time-series experiment, the Omics Viewer animation offers one view of the data. However, it can also be useful to visualize the data from all time points simultaneously. Thus, for a given object (such as a gene or a metabolite) or set of objects (such as all the genes or metabolites in a pathway), the software can generate a popup overlay depicting the omics data for all time points (Figure 5). The data can be displayed either as a heat map, a bar graph, or a plot. Any number of popups can be displayed simultaneously, and the user can drag them with the mouse to reposition them as desired.

Acknowledgments

The projects described were supported by award numbers GM077678, GM080746, and GM75742 from the National Institute of General Medical Sciences, respectively. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institute of General Medical Sciences, or the National Institutes of Health.

References

1. Karp PD, Paley SM, Krummenacker M, Latendresse M, Dale JM, Lee T, Kaipa P, Gilham F, Spaulding A, Popescu L, Altman T, Paulsen I, Keseler IM, Caspi R. Pathway Tools version 13.0: Integrated software for pathway/genome informatics and systems biology. *Briefings in Bioinformatics*. 2010; 11:40–79. 10.1093/bib/bbp043 [PubMed: 19955237]
2. Caspi R, Altman T, Dale JM, Dreher K, Fulcher CA, Gilham F, Kaipa P, Karthikeyan AS, Kothari A, Krummenacker M, Latendresse M, Mueller LA, Paley S, Popescu L, Pujar A, Shearer AG, Zhang P, Karp PD. The MetaCyc database of metabolic pathways and enzymes and the BioCyc collection of pathway/genome databases. *Nuc Acids Res*. 2010; 38:D473–9. advanced access. 10.1093/nar/gkp875
3. Pathway Tools File Formats. <http://bioinformatics.ai.sri.com/ptools/flatfile-format.html>
4. BioCyc Downloads. <http://biocyc.org/download.shtml>
5. Pathway Tools PGDB Registry. <http://BioCyc.org/registry.html>
6. Demir E, et al. The BioPAX community standard for pathway data sharing. *Nat Biotechnol*. 2010; 28(12):935–942. [PubMed: 20829833]
7. Pathway Tools Data-File Formats. <http://bioinformatics.ai.sri.com/ptools/flatfile-format.html>
8. Pathway Tools Web Services. <http://biocyc.org/PToolsWebsiteHowto.shtml#webServices>
9. Guide to ptools-xml. <http://biocyc.org/ptools-xml-guide.shtml>
10. The BioVelo Query Language. <http://biocyc.org/bioveloLanguage.html>
11. Querying Pathway/Genome Databases. <http://brg.ai.sri.com/ptools/ptools-resources.html>
12. Karp, PD.; Myers, K.; Grube, T. The Generic Frame Protocol. *Proceedings of the 1995 International Joint Conference on Artificial Intelligence*; 1995. p. 768-774.
13. Seibe, P. *Practical Common Lisp*. APress; 2005. 1
14. Graham, P. *ANSI Common Lisp*. Prentice Hall; 1995.
15. Example Pathway Tools queries in Common Lisp. <http://bioinformatics.ai.sri.com/ptools/examples.lisp>
16. PerlCyc API to Pathway Tools. <http://solgenomics.net/downloads/perlcyc.pl>
17. JavaCyc API to Pathway Tools. <http://solgenomics.net/downloads/index.pl>
18. Karp PD, Chaudhri Vinay K, Paley Suzanne M. A collaborative environment for authoring large knowledge bases. *J Intelligent Information Systems*. 1999; 13:155–94.

19. Grossmann, Steffen; Bauer, Sebastian; Robinson, Peter N.; Vingron, Martin. Improved detection of overrepresentation of gene-ontology annotations with parent child analysis. *Bioinformatics*. 2007; 23(22):3024–3031. [PubMed: 17848398]
20. Khatri, Purvesh; Draghici, Sorin; Charles Ostermeier, G.; Krawetz, Stephen A. Profiling gene expression using onto-express. *Genomics*. 2002; 79(2):266–270. [PubMed: 11829497]
21. Rivals, Isabelle; Personnaz, Leon; Taing, Lieng; Potier, Marie-Claude. Enrichment or depletion of a GO category within a class of genes: which test? *Bioinformatics*. 2006:633.
22. Guide to the Pathway Tools Schema. <http://biocyc.org/schema.shtml>
23. Karp PD. An ontology for biological function based on molecular interactions. *Bioinformatics*. 2000; 16(3):269–285. [PubMed: 10869020]
24. Karp, PD.; Paley, SM. Representations of metabolic knowledge: Pathways. In: Altman, R.; Brutlag, D.; Karp, P.; Lathrop, R.; Searls, D., editors. *Proc Second International Conference on Intelligent Systems for Molecular Biology*. Menlo Park, CA: AAAI Press; 1994. p. 203-211.
25. Karp, PD.; Riley, M. Representations of metabolic knowledge. In: Hunter, L.; Searls, D.; Shavlik, J., editors. *Proc First International Conference on Intelligent Systems for Molecular Biology*. Menlo Park, CA: AAAI Press; 1993. p. 207-215.

```

a) Lisp (select-organism :org-id 'ECOLI)
(loop for x in (get-class-all-instances '|Regulation-of-Enzyme-Activity|)
  ;; Does the regulator slot contain ATP, and is the mode of regulation
  ;; negative (representing inhibition)?
  when (and (member-slot-value-p x 'regulator 'atp)
    (member-slot-value-p x 'mode "-"))
  ;; The value of the regulated-entity slot is an Enzymatic-Reaction
  ;; object, so we need to access its enzyme slot to retrieve the enzyme.
  ;; The collected values are returned as a list.
  collect (get-slot-value (get-slot-value x 'regulated-entity) 'enzyme))

b) Perl use perlcyc;
my $cyc = perlcyc->new("ECOLI");
my @regs = $cyc->get_class_all_instances("|Regulation-of-Enzyme-Activity|");
foreach my $reg (@regs) {
  my $bool1 = $cyc->member_slot_value_p($reg, "regulator", "ATP");
  my $bool2 = $cyc->member_slot_value_p($reg, "mode", "-");
  if ($bool1 && $bool2) {
    my $enzrxn = $cyc->get_slot_value($reg, "regulated-entity");
    my $enz = $cyc->get_slot_value($enzrxn, "enzyme");
    ##Print the results on the terminal
    print STDOUT "$enz\n";
  }
}

c) Java Import java.util.*;

public class JavaCycSample {
  public static void main(String[] args) {
    Javacyc cyc = new Javacyc("ECOLI");
    ArrayList regs =
      cyc.getClassAllInstances("|Regulation-of-Enzyme-Activity|");
    for (int i=0; i<regs.size(); i++) {
      String reg = (String)regs.get(i);
      boolean bool1 = cyc.memberSlotValueP(reg, "regulator", "ATP");
      boolean bool2 = cyc.memberSlotValueP(reg, "mode", "-");
      if (bool1 && bool2) {
        String enzrxn = cyc.getSlotValue(reg, "regulated-entity");
        String enzyme = cyc.getSlotValue (enzrxn, "enzyme");
        System.out.println(enz);
      }
    }
  }
}

```

Figure 1.

Sample code showing how to find all enzymes inhibited by ATP in EcoCyc using the a) Lisp, b) Perl, and c) Java APIs.

Figure 2 illustrates some of the relationships between different classes of objects. More detailed information about classes and their slots can be found at [22, 23, 24, 25] or in the Pathway Tools User's Guide.

Figure 2. Some of the major relationships between classes of objects in the Pathway Tools schema. For example, the arrow labeled reaction-list going from Pathways to Reactions indicates that the Pathways class has a slot reaction-list, whose values are members of the Reactions class.

Figure 3.
The Cellular Omics Viewer displays many kinds of high-throughput data.

Figure 5.
A pathway display showing the three different styles of omics popups.

Table 1

Other Pathway Tools-based websites.

Website	Contents	URL
ApiCyc	Apicomplexan genomes	http://apicyc.apidb.org/PLASMO/server.html
AraCyc	<i>Arabidopsis thaliana</i>	http://www.arabidopsis.org/biocyc/index.jsp
BeoCyc	33 BioEnergy genomes	http://cricket.ornl.gov/cgi-bin/beocychome.cgi
BioCyc	1,004 microbial Genomes	http://biocyc.org/
CalbiCyc	<i>Candida albicans</i>	http://pathway.candidagenome.org/
ChlamyCyc	<i>Chlamydomonas reinhardtii</i>	http://chlamycyc.mpimp-golm.mpg.de
EcoCyc	<i>Escherichia coli</i>	http://ecocyc.org/
FungiCyc	Multiple fungal genomes	http://fungicyc.broadinstitute.org:1555
DictyCyc	<i>Dictyostelium discoideum</i>	http://dictybase.org/DictyInfo/dictycycinfo.html
LeishCyc	<i>Leishmania major</i>	http://leishcyc.bio21.unimelb.edu.au/
MicroScope	542 Genomes	http://www.genoscope.cns.fr/agg/microscope/
MouseCyc	<i>Mus musculus</i>	http://mousecyc.jax.org:8000/
PseudoCyc	<i>Pseudomonas aeruginosa</i>	http://v2.pseudomonas.com:1555/
RiceCyc	Multiple plant genomes	http://www.gramene.org/pathway/
Scocyc	Streptomyces genomes	http://scocyc.streptomyces.org.uk:14980/SCO/server.html
ShewCyc	18 Shewanella genomes	http://spruce.ornl.gov:1555/SHEON/organism-summary?object=SHEON
SolCyc	Solanacea genomes	http://solcyc.sgn.cornell.edu/
SoyCyc	Soybean	http://soybase.org
TBestDB	Taxonomically Broad EST DB	http://pepdbpub.bcm.umontreal.ca/pathway/
TBCyc	<i>Mycobacterium tuberculosis</i>	http://tbcyc.tbdb.org/
SGD	<i>Saccharomyces cerevisiae</i>	http://www.yeastgenome.org/

Table 2

Some of the major classes and subclasses in the Pathway Tools schema

Compounds-And-Elements	Compounds Elements
Proteins	Polypeptides Protein-Complexes
RNAs	tRNAs rRNAs Regulatory-RNAs
Genetic-Elements	Chromosomes Plasmids Contigs
DNA-Segments	Genes DNA-Binding-Sites Promoters Terminators Transcription-Units
Protein-Features	Modified-Residues Active-Sites Signal-Sequences
Reactions	Chemical-Reactions Transport-Reactions Binding-Reactions Redox-Half-Reactions
Pathways	
Enzymatic-Reactions	
Regulation	Regulation-of-Enzyme-Activity Regulation-of-Transcription-Initiation Transcriptional-Attenuation Regulation-of-Translation
Organisms	
Publications	
Evidence	
CCO (Cell Component Ontology)	
Gene-Ontology-Terms	

Compounds-And-Elements	Compounds
	Elements
Databases	