AN EVIDENCE ONTOLOGY FOR USE IN PATHWAY/GENOME DATABASES

Appears in Pacific Symposium on Biocomputing 2004
Pages 190-201, World Scientific Publishers, Singapore

P.D. KARP, S. PALEY, C.J. KRIEGER
SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025 USA
{*pkarp,paley,krieger*}*@ai.sri.com*

P. ZHANG
Carnegie Institution of Washington, Department of Plant Biology
260 Panama Street, Stanford, California 94305
peifenz@acoma.stanford.edu

Abstract. An important emerging need in Model Organism Databases (MODs) and other bioinformatics databases (DBs) is that of capturing the scientific evidence that supports the information within a DB. This need has become particularly acute as more DB content consists of computationally predicted information, such as predicted gene functions, operons, metabolic pathways, and protein properties. This paper presents an ontology for encoding the type of support and the degree of support for DB assertions, and for encoding the literature source in which that support is reported. The ontology includes a hierarchy of 35 evidence codes for modeling different types of wet-lab and computational evidence for the existence of operons and metabolic pathways, and for gene functions. We also describe an implementation of the ontology within the Pathway Tools software environment, which is used to query and update Pathway/Genome DBs such as EcoCyc, MetaCyc, and HumanCyc.

1 Introduction

An important emerging need in Model Organism Databases (MODs) and other bioinformatics databases (DBs) is that of capturing the scientific evidence that supports the information within a DB. This need has become particularly acute as more DB content consists of computationally predicted information, such as predicted gene functions, operons, metabolic pathways, and protein properties. DB users want to know the *type* of evidence that supports assertions within a DB, and they want to know the *strength* of that evidence. Strength and type are in general independent parameters, although they are often related; for example, computationally generated predictions are generally held to be less reliable than are wet-lab experiments, but there are certainly unreliable types of wet-lab methods.

This paper reports on an ontology for encoding the type of support and the degree of support for DB assertions, and for encoding the literature source

in which that support is reported. We also describe an implementation of the ontology within the Pathway Tools software environment,[1] which is used to query and update Pathway/Genome DBs (PGDBs) such as EcoCyc, [2] MetaCyc, [3] and HumanCyc (see URL http://HumanCyc.org/).

2 Motivations for an Evidence Ontology

The evidence ontology is designed to encode information about *why* we believe certain assertions in a PGDB, the *sources* of those assertions, and the *degree of confidence* scientists hold in those assertions.

An assertion could be the existence of a biological object described in a PGDB. For example, we would like to be able to encode the evidence supporting the existence of a gene, an operon, or a pathway that is described within a PGDB. Has the operon been predicted using a computational operon finder? Or is it supported by wet-lab experiments? If the latter, what types of experimental methods were used? It should be possible to capture multiple types of evidence: if the existence of a metabolic pathway is supported by both a computational algorithm and by two different types of wet-lab experiments, our evidence ontology should be able to capture that information, and also to capture the literature citations that are the source of that information.

We also want to be able to capture two types of confidence information in the evidence ontology. If the probability of correctness of an individual piece of evidence is known, we want to capture that. For example, if we have measured the overall accuracy of an operon predictor and found that accuracy to be 80%, then computational operon predictions made by that program should be recorded to have an individual confidence of 0.8. Similarly, if we know that a wet-lab method has a probability of correctness of 0.7, we should be able to capture that information in conjunction with our evidence codes. But in addition to capturing the confidence in individual pieces of evidence, we want to capture the overall confidence in an assertion that results from synthesizing across multiple pieces of evidence. Consider a case where the existence of a metabolic pathway is supported by a computational prediction and by two wet-lab experiments. We would like a curator to be able to record his or her overall confidence level in that pathway that results from integrating those three pieces of evidence.

Object existence is one class of PGDB assertion. But object properties and relationships form another important class of assertions. We should be able to encode evidence not just about object existence, but also regarding slot values stored in a PGDB, such as the type of evidence supporting the molecular weight of a protein, or the assertion that pyruvate inhibits an enzyme, or the assertion that a protein catalyzes a given reaction.

2.1 Related Work

Gene Ontology provides an evidence ontology that satisfies some of the preceding criteria, and that formed the starting point for our work.[4] The GO evidence codes are described at URL http://geneontology.org/doc/GO.evidence.html. Wherever possible we have adopted the GO evidence codes, or small variations of them, to facilitate translation between the systems. But in many cases we significantly extended or reworked the GO system because it was not designed to satisfy, and cannot satisfy, the requirements listed earlier in this section. For example: (a) The GO system does not encode specific classes of experimental methods, that is, the GO code "IDA" (Inferred from Direct Assay) has no subclasses to define subtypes of experimental assays (such as assays that provide evidence for the activity of an enzyme or for the presence of a promoter). (b) Strictly speaking, the GO evidence system is intended to be used to annotate the support for attachment of a GO term to a gene. It is not specifically designed for use to record evidence for the existence of a biological object, or for other types of assertions such as slot values. (c) The GO evidence system does not provide a way to associate confidence values with assertions.

We are unaware of directly related work in the Artificial Intelligence community. AI work on truth maintenance systems (TMSs) is not relevant because TMSs are concerned with capturing relationships between propositions inferred by an automated reasoner, and the propositions on which those inferences depend [5]. That is a different problem than trying to capture general classes of evidence that support some proposition.

3 Overview of Pathway Tools

The Pathway Tools software is a reusable package for creating, querying, visualizing, and analyzing MODs. Its components include the following.

PathoLogic — This is a module for computationally creating a new PGDB for an organism from its annotated genome. PathoLogic includes a metabolic pathway predictor [6] and an operon predictor. Given a properly formatted Genbank entry for an annotated genome, PathoLogic can create a new PGDB within a day. Additional manual processing that is typically required before a PGDB is ready for release takes 2–3 weeks for a bacterial genome.[7]

Pathway/Genome Editors — This module includes a graphical interactive editing tool for every datatype managed by Pathway Tools, including genes, proteins, biochemical reactions and pathways, small molecule metabolites, and operons.

Pathway/Genome Navigator — This module allows users to query a PGDB and to visualize the results of a query. Visualization tools supported include visualization of chromosomes (genome browser), genes, proteins (with

specialized displays for enzymes, for transporters, and for transcription factors — the latter displays all operons controlled by the transcription factor), pathways, and transcription units (operons). A **metabolic overview** diagram is a drawing of all known metabolic pathways of an organism. Expression data for a given organism can be painted onto the metabolic overview to place expression data in a pathway context and to allow the user to discern the coordinated expression of entire pathways, or of important steps within a pathway.

SRI has used Pathway Tools to develop 15 PGDBs, which are available through the BioCyc Web site at URL http://BioCyc.org/. In addition, Pathway Tools has been licensed by 31 groups in academia and industry. PGDBs available from those groups are also listed at URL http://BioCyc.org/.

Recent enhancements to Pathway Tools include (1) ontology, visualization, and editing support for introns, exons, and alternative splicing; (2) tools for exporting PGDB information to flat files, and for importing information from flat files into PGDBs; (3) implementation of Perl and Java APIs for querying and updating PGDBs, called PerlCyc and JavaCyc, respectively (see URL http://bioinformatics.ai.sri.com/ptools/ptools-resources.html).

4 Pathway Tools Implementation of the Evidence Ontology

This section describes how the Pathway Tools evidence ontology is presented to the user in Pathway Tools displays to give the reader an understanding of how the ontology is used. We extended Pathway Tools so that the pathway and operon predictors within PathoLogic decorate the pathway and operon PGDB objects that they create with evidence-code information to indicate computationally predicted objects as such. We extended the Editors to include functionality that allows users to interactively enter and modify evidence codes. We extended the Navigator to display evidence information.

For example, the Navigator window shown in Figure 1 displays information about the transcription unit[a] called cbl. The flask icon at the top right of the diagram indicates that evidence from wet-lab experiments supports the existence of the transcription unit. The lower flask and computer icons adjacent to the "Promoter: cbl" line indicate that both wet-lab experiments and computational predictions support the existence of this promoter. Finally, the flask at the bottom right of the window indicates that experimental evidence supports the information about the activity of the transcription factor.

Although our evidence system provides for more precise distinctions than simply "wet-lab" versus "computational," we felt it best to keep our graphical

[a] Transcription units are essentially the same as operons, although transcription units can contain single genes, whereas by definition operons must contain multiple genes.

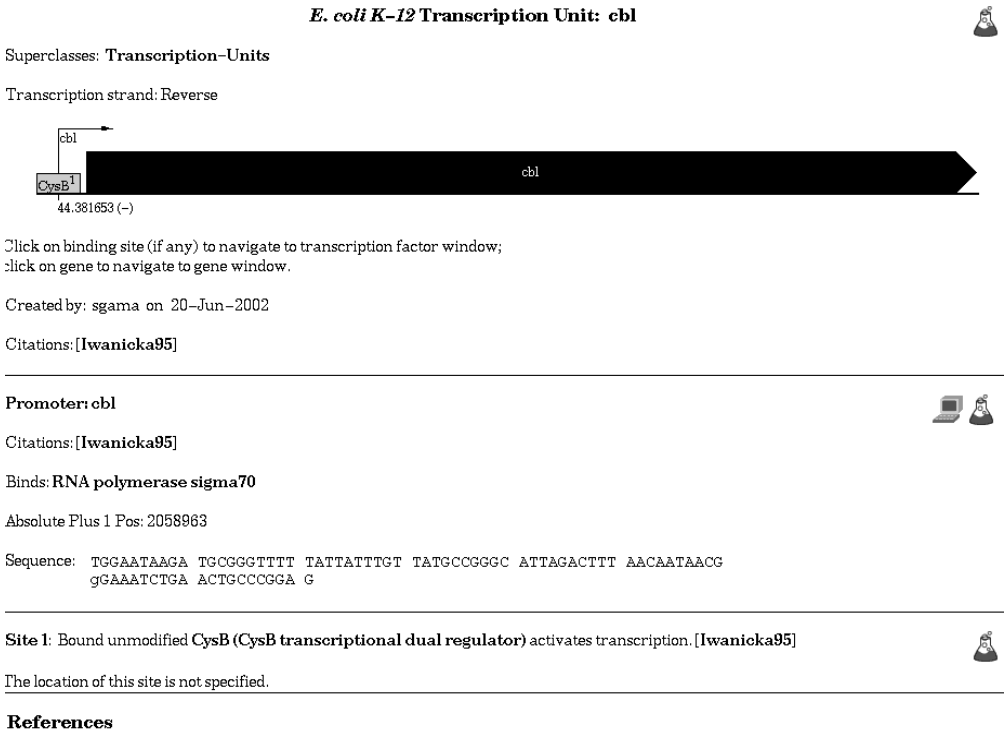

Figure 1: Pathway Tools display of the EcoCyc transcription unit cbl.

interface simple by displaying only a few different icons, since expecting users to learn icons for each of our 35 evidence codes would be unreasonable. Therefore, object display windows such as the transcription-unit display in Figure 1 show icons that only differentiate evidence codes at the highest level of our hierarchy, such as distinguishing experimental evidence from computational evidence. The user can click on these icons to view another screen that shows the detailed evidence codes that support the existence of an object (see Figure 2), and from what literature sources that evidence was reported.

5 The Evidence Ontology

Each piece of evidence about object existence in PGDBs is recorded as a structured *evidence tuple*, as a value within the Citations slot of PGDB objects such

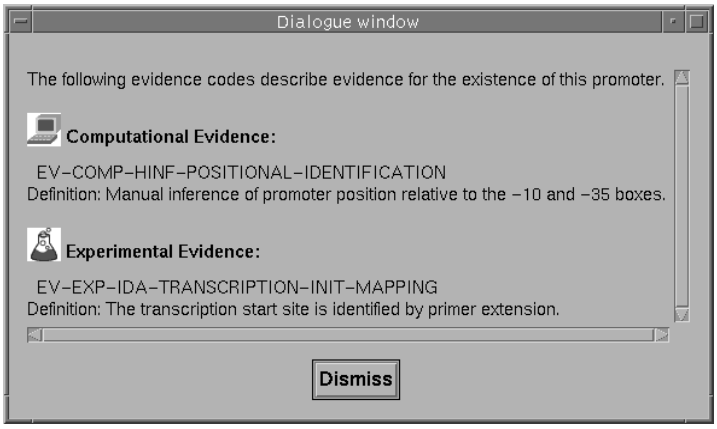

Figure 2: Detailed evidence report for existence of the cbl promoter.

as pathways and transcription units. An evidence tuple allows us to associate several types of information within one piece of evidence. Each *evidence tuple* is of the form:

`Evidence-code : Citation : Curator : Timestamp : Probability`

where

- `Evidence-code` is a unique ID for the type of evidence, as provided in Table 1.
- `Citation` is an optional citation identifier such as a PubMed ID that indicates the source of the evidence. For computational evidence, the citation refers to an article describing either the general properties of the algorithm used, or its application in this case.
- `Curator` is the username of the curator who created this evidence tuple.
- `Timestamp` is an optional integer representing the time and date on which this evidence tuple was created.
- `Probability` is an optional real number that indicates the probability that the assertion supported by this evidence is correct, such as a probability provided by an algorithm. We expect that the probability portion of the evidence tuple will be used much more frequently for computational evidence than for wet-lab evidence because the accuracies of

computational techniques tend to be better known in general than for experimental methods.

The notion of what it means for a biological object to exist varies somewhat by object type, and is difficult to define precisely. For example, what does it mean for a gene to exist? Does the existence of a gene depend only on whether some gene product is produced from a region of DNA, or on whether the exact boundaries of the gene are defined precisely? In the case of a gene, the probability of existence should not reflect whether the exact nucleotide start and stop positions of the gene are correct, but should depend only on whether a gene product is produced by an approximate region of a chromosome, due to the accuracy limits of current gene finders. In contrast, the notion of probability of existence of a transcription unit should indeed depend on where the gene boundary of the transcription unit lies, since *every* bacterial gene lies within *some* transcription unit, and the fundamental problem of predicting transcription units is defining their gene boundaries.

Finally, we defined a slot Confidence that allows a PGDB to record an overall integrated probability for the existence of an object. Whereas probabilities within evidence tuples encode the probability associated with an *individual* piece of evidence, the Confidence slot is intended to hold the *net* probability that results from integrating across the potentially multiple pieces of evidence available. This integration process will in most cases be a manual process performed by a curator, that will therefore be subjective and will vary among individuals, and we are in the process of developing guidelines for that integration process. Note our implementation performs no manipulation of probability values other than permitting their entry and display.

5.1 The Hierarchy of Evidence Codes

The `Evidence-code` components of evidence tuples denote different types of evidence. Those evidence types are arranged in a generalization–specialization hierarchy as shown in Table 1. The root of that hierarchy, the node Evidence, has four direct children that define the four main evidence types:

- EV–Comp: Inferred from Computation. The evidence for an assertion comes from a computational analysis. The assertion itself might have been made by a person or by a computer, that is, EV–Comp does not specify whether manual interpretation of the computation occurred.

- EV–Exp: Inferred from Experiment. The evidence for an assertion comes from a wet-lab experiment of some type.

EV–Comp	Inferred from computational analysis.
EV–Comp–HInf	Inferred by a human based on computational evidence.
EV–Comp–HInf–Positional–Identification	Human inference of promoter position.
EV–Comp–HInf–Similar–To–Consensus	Human inf. based on similarity to consensus sequences.
EV–Comp–HInf–Fn–From–Seq	Human inf. of function from sequence.
EV–Comp–AInf	Inferred computationally without human oversight — automated inference.
EV–Comp–AInf–Positional–Identification	Automated inf. of promoter position.
EV–Comp–AInf–Similar–To–Consensus	Automated inf. based on similarity to consensus sequences.
EV–Comp–AInf–Fn–From–Seq	Automated inf. of function from sequence.
EV–Comp–AInf–Single–Directon	Automated inf. that a single-gene directon is a transcription unit.
EV–Exp	Inferred from experiment.
EV–Exp–IPI	Inferred from physical interaction.
EV–Exp–IMP	Inferred from mutant phenotype.
EV–Exp–IMP–Site–Mutation	Site mutation.
EV–Exp–IMP–Polar–Mutation	Polar mutation.
EV–Exp–IMP–Reaction–Blocked	Reaction blocked in mutant.
EV–Exp–IMP–Reaction–Enhanced	Reaction enhanced in mutant.
EV–Exp–IGI	Inferred from genetic interaction.
EV–Exp–IGI–Func–Complementation	Inferred by functional complementation.
EV–Exp–IEP	Inferred from expression pattern.
EV–Exp–IEP–Gene–Expression–Analysis	Gene expression analysis.
EV–Exp–IDA	Inferred from direct assay.
EV–Exp–IDA–Binding–Of–Cellular–Extracts	Binding of cellular extracts.
EV–Exp–IDA–Binding–Of–Purified–Proteins	Binding of purified proteins.
EV–Exp–IDA–RNA–Polymerase–Footprinting	RNA polymerase footprinting.
EV–Exp–IDA–Transcription–Init–Mapping	Transcription initiation mapping.
EV–Exp–IDA–Boundaries–Defined	Boundaries of transcription experimentally identified.
EV–Exp–IDA–Transcript–Len–Determination	Length of transcript experimentally determined.
EV–Exp–IDA–Unpurified–Protein	Assay of unpurified protein.
EV–Exp–IDA–Purified–Protein–Multspecies	Assay of protein purified from mixed culture.
EV–Exp–IDA–Purified–Protein	Assay of purified protein.
EV–IC	Inferred by curator.
EV–AS	Author statement.
EV–AS–TAS	Traceable author statement.
EV–AS–NAS	Non-traceable author statement.

Table 1: The taxonomy of evidence types. Each row of this table defines one evidence type, giving its code and description. Indentation indicates ordering in the taxonomy, for example, EV–AS–TAS (fourth row) is a child of EV–AS.

- EV–IC: Inferred by Curator. An assertion was inferred by a curator from relevant information such as other assertions in a database.

- EV–AS: Author Statement. The evidence for an assertion comes from an author statement in a publication, where that publication does not provide direct experimental support for the assertion. (Ordinarily, this code will not be used directly — generally one of its child codes, EV–AS–TAS or EV–AS–NAS, will be used instead.)

We expect the most commonly used codes will be EV–Comp and EV–Exp, and their sub-codes.

An HTML version of the entire evidence ontology, including detailed comments describing each evidence code, is available at URL http://bioinformatics.ai.sri.com/ptools/evidence-ontology.html.

There are several reasons why we feel this evidence system is best structured as a hierarchical ontology rather than as a flat list of controlled terms. First, we expect that the hierarchy will facilitate understanding of the evidence system by new curators because terms are grouped into logically related clusters. This aspect will be particularly important as the size and complexity of the evidence ontology grows to model additional detailed evidence types. Second, we expect the hierarchy will facilitate the curation process itself by allowing faster retrieval of relevant terms from editor menus than if retrieval was from a flat list. Third, non-leaf nodes in the evidence ontology will themselves be used in curation in cases where leaf evidence nodes do not match the evidence that is actually available, or where a publication is not specific about the type of evidence that supports some assertion, thus requiring a more general evidence code.

The lower levels of the hierarchy have thus far been designed primarily for encoding the evidence for protein function, and evidence related to mechanisms of regulation of transcription initiation. There are many types of experiments and computational techniques, but our curators have made efforts to divide and group them into the categories they judge to be most meaningful. These evidence codes are not comprehensive with respect to other types of biological information. In the future, if we decide to apply evidence codes to different types of objects, we expect to extend the existing set of evidence codes to cover the types of experiments and analyses applicable to the new object types.

In our implementation of the evidence codes, each code is defined as a class within the Ocelot object DBMS whose object ID is the evidence code. Slots defined for each evidence code include its name (such as "Inferred by Computational Analysis"), a comment describing the code, and a slot Pertains–To that lists the classes of PGDB objects to which the code can be applied. The Pathway Tools editors query this slot to determine what evidence codes

are applicable to a given type of objects, such as a metabolic pathway, when generating choose-lists of evidence codes for the curator.

The EV–AS (Author Statement) category has two subtypes: author statements that are traceable to a publication that contained direct evidence for an assertion, and statements that are not traceable in that manner.

The EV–Comp (Inferred by Computational Analysis) category is also divided into two subtypes: computational inferences that were made in a purely automated fashion, and inferences in which a human was involved, under the assumption that the condition of whether or not a person was involved in arbitrating among computational evidence is a factor that a scientist interpreting the evidence would consider important.

An examination of the subtypes of EV–Comp reveals that some of these evidence types apply only to certain PGDB object types. For example, the code EV–Comp–AInf–Single–Directon applies only to computational inference of operons (it indicates that an operon was inferred by the existence of a gene G for which the adjacent genes on both sides of G are both transcribed in the opposite direction from G, implying that those genes cannot be in the same operon as G). This property of being relevant only to specific object types applies to other evidence types in our system.

5.2 Attaching Evidence Codes to Individual Slot Values

As well as using evidence tuples to record evidence about object existence, we can attach evidence tuples to individual values within a PGDB to record evidence for finer-grained assertions. For example, we could record the evidence that supports the strength of a promoter stored in a PGDB, or that supports the assertion that a given metabolite inhibits the activity of an enzyme.

5.3 Object and Relational Implementations of Evidence Tuples

We implement the association of evidence tuples with individual slot values using an Ocelot mechanism called *annotations*. An annotation is a five-tuple of the form

```
Frame : Slot : Value : Label : Datum
```

that allows a labeled datum to be associated with a value within a slot of a frame. In this case, the label is the string "Evidence" and the datum is the evidence tuple itself. In Ocelot, annotation tuples are physically stored within the frame that they are associated with.

We envision that our evidence system could be implemented in a relational DBMS by creating a table whose columns are `Table-ID, Key, Column, Value, Evidence-Code, Citation, Curator, Timestamp,` and `Probability`. `Table-ID, Key,` and `Column` are analogous to `Frame` and `Slot` in the Ocelot

representation — they identify an "object" within a relational DBMS. `Column` and `Value` identify a specific relational column and value within that column to which the evidence tuple applies. When the evidence tuple applies to the entire object, `Column` and `Value` would be null. In both the Ocelot and the relational implementations, it is straightforward to compute across evidence information, such as to query all pathways with experimental evidence.

6 Use of the Evidence Ontology within EcoCyc and MetaCyc

The EcoCyc DB currently records evidence codes for the existence of 341 of its 810 transcription units, for 531 of its 878 promoters, and for 952 of its 1071 DNA binding sites. The preceding evidence codes were assigned over the past few years using an earlier, more primitive evidence system. We translated codes from that system into the system described here, which was feasible because the new system was explicitly designed to be a superset of the old system. Assignment of evidence codes for pathways and protein functions by curators of the EcoCyc and MetaCyc DBs is just beginning.

As an example, the anaerobic toluene degradation pathway, described in MetaCyc as it appears in *Thauera aromatica,* has been established by biochemical assay of the enzymes, pathway intermediates, and products. Thus, we would assign to it the evidence code EV–Exp–IDA (Inferred by Direct Assay), along with one or more citations to the literature. If a user were to use PathoLogic to create a PGDB for some other organism, X, and PathoLogic inferred that the anaerobic toluene degradation pathway were present in X, it would be assigned an evidence code of EV-AInf (automated inference) in the new PGDB (this assignment is done automatically by PathoLogic).

A protein that is multifunctional may have the same or different evidence codes assigned to each function. In addition to containing objects for each protein and for each reaction, the Pathway Tools schema defines an object, called an enzymatic-reaction, that describes the pairing of an enzyme to a reaction. We assign the evidence code for each function of a protein to the corresponding enzymatic-reaction object rather than to the protein object so there is no ambiguity about which functional assignment each evidence code pertains to. For example, the product of the *E. coli ndh* gene has NADH dehydrogenase and NADH cupric reductase activities. The NADH dehydrogenase activity was established by direct assay of the purified protein. The NADH cupric reductase activity was established by observing that the reaction is blocked or enhanced in mutants with the gene missing or over-expressed, respectively. Thus, we assign the evidence code EV–Exp–IDA–Purified–Protein (along with relevant literature citations) to the enzymatic-reaction that links the *ndh* gene product to the NADH de-

hydrogenase reaction, and the codes EV-IMP–Reaction–Enhanced and EV–Exp–IMP–Reaction–Blocked to the enzymatic-reaction that links the protein to the NADH cupric reductase reaction. (Note that this is not intended to be a complete description of the *ndh* system — additional codes may apply to one or both activities.) We are not restricted to using the lowest-level codes in the evidence ontology. If the enzyme was characterized by some experiment that did not precisely match one of our lowest-level codes, or the literature did not provide enough information to distinguish between them, we might assign a code of EV–Exp–IMP even EV–Exp in place of one of the more specific codes.

7 Software Availability

Pathway Tools for SUN/Solaris, Intel/Windows, and Intel/Linux is freely available to academics; a license fee applies to commercial use. See http://BioCyc.org/download.shtml for more details.

Acknowledgments

Julio Collado-Vides contributed to development of evidence codes related to transcriptional regulation. This work was supported by grant R01-HG02729-01 from the NIH National Human Genome Research Institute. The contents of this article are solely the responsibility of the authors and do not necessarily represent the official views of the National Institutes of Health.

1. P.D. Karp, S. Paley, and P. Romero. The Pathway Tools Software. *Bioinformatics*, 18:S225–S232, 2002.
2. P.D. Karp, M. Riley, M. Saier, I.T. Paulsen, S. Paley, and A. Pellegrini-Toole. The EcoCyc database. *Nuc. Acids Res.*, 30(1):56–8, 2002.
3. P.D. Karp, M. Riley, S. Paley, and A. Pellegrini-Toole. The MetaCyc database. *Nuc. Acids Res.*, 30(1):59–61, 2002.
4. M. Ashburner, C.A. Ball, J.A. Blake, D. Botstein, H. Butler, J.M. Cherry, A.P. Davis, K. Dolinski, S.S. Dwight, J.T. Eppig, M.A. Harris, D.P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J.C. Matese, J.E. Richardson, M. Ringwald, G.M. Rubin, and G. Sherlock. Gene Ontology: Tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.
5. J. Doyle. Truth maintenance systems. *Artificial Intelligence*, 12(3):231–72, 1979.
6. S. Paley and P.D. Karp. Evaluation of computational metabolic-pathway predictions for *H. pylori*. *Bioinformatics*, 18(5):705–714, 2002.
7. P. Romero and P.D. Karp. PseudoCyc, a Pathway/Genome Database for *Pseudomonas aeruginosa*. *J. Mol. Microbiol. Biotech.*, 5(4):230–9, 2003.