Using Term Lists and Inverted Files to Improve Search Speed for Metabolic Pathway Databases

Greeshma Neglur[1], Robert L. Grossman[2], Natalia Maltsev[3], and Clement Yu[4]

[1]Laboratory for Advanced Computing,
University of Illinois at Chicago, Chicago, IL 60607, USA
neglur@lac.uic.edu
[2]Laboratory for Advanced Computing,
University of Illinois at Chicago, Chicago, IL 60607, USA
grossman@uic.edu
[3]Math and Computer Science Division,
Argonne National Laboratory, Argonne, IL 60439, USA
maltsev@mcs.anl.gov
[4]Department of Computer Science,
University of Illinois at Chicago, Chicago, IL 60607, USA
yu@cs.uic.edu

This is a draft of the article: Greeshma Neglur, Robert L. Grossman, Natalia Maltsev, and Clement Yu, Using Term Lists and Inverted Files to Improve Search Speed for Metabolic Pathway Databases, 3rd International Workshop on Data Integration in the Life Sciences 2006 (DILS'06), Lecture Notes in Bioinformatics, Volume 4075, Springer-Verlag, Berlin, 2006, pages 168-184.

Abstract. This paper describes a technique for efficiently searching metabolic pathways similar to a given query pathway, from a pathway database. Metabolic pathways can be converted into labeled directed graphs where the nodes represent chemical compounds. Similarity between two graphs can be computed using a metric based on Maximal Common Subgraph (MCS). By maintaining an inverted file that indexes all pathways in a database on their edges, our algorithm finds and ranks all pathways similar to the user input query pathway in time, which is linear in the total number of occurrences of the edges in common with the query in the entire database.

1 Introduction

Understanding of the complex architecture of metabolic networks provides insights into the fundamental design principles underlying the structure and function of living organisms. Common ancestry leads to the similarity of many molecular functions observed in all domains of life (Eukaryotes, Prokaryotes and Archaeabacteria). However, differences in organisms' physiology and lifestyle result in divergent evolution and emergence of variants of metabolic organization and phenotypic features. Large amount of metabolic and proteomic data available in public databases now allows for systematic exploration of adaptive mechanisms that led to the diversification of biological systems and the emergence of metabolic pathways characteristic of particular taxonomic or phenotypic groups of organisms. Such evolutionary and comparative analysis of metabolic pathways represents one of the

essential problems in life sciences and is essential for progress in medicine, biotechnology and bioremediation. Metabolic pathways corresponding to various metabolic processes in an organism may be represented as a labeled directed graph.

The basic elements of metabolic pathways are chemical reactions that include compounds (e.g., substrates, products) and enzymes. Hence, computing similarity between pathways involves matching the constituent reactions (that include substrates and enzymes) and the connectivity between them. Brute force methods of matching the structures of substrates, enzymes and the pathway itself involve graph isomorphism tests at three levels and turn out to be computationally very expensive. The problem is further complicated if we are trying to query a database consisting of thousands of pathways with an average pathway size of 20+ nodes. Hence, efficient techniques for querying pathway databases are essential.

Our technique is able to search and retrieve pathways from a database similar to a query pathway in *time linear in the total number of occurrences of the pathway edges that are in common with the query* in the entire database. We do this by employing a simple indexing technique that uses terms defined from pathway edges and inverted files containing these terms.

2 Related Work

Many metabolic pathway similarity computing algorithms [8, 9] are based on abstracting pathways as enzyme graphs, i.e., directed labeled graphs where nodes are labeled with enzyme EC [12] (Enzyme Commission) numbers and a directed edge from one node to another implies that the product of the former node is the substrate of the latter. EC numbers provide a hierarchical classification of enzymes based on the reactions they catalyze. The classification tree consists of 4 levels with a root. Each enzyme is assigned a string consisting of 4 numbers, each of which corresponds to a level, for example: 1.2.3.4. In [8] the algorithm takes as input a sequence of EC numbers representing the enzyme graphs, aligns one sequence to another and attempts to find all EC numbers with the same 4-level hierarchical numbers, scores the similarities and cuts the sequences by removing the identical EC numbers and each pair of sub-sequences is initialized to begin a new round of 3-level hierarchical EC number match and so on. Another polynomial time algorithm described in [9] uses Approximate Labeled Subgraph Homeomorphism. A disadvantage of the enzyme graph representation is that it does not incorporate the similarity verification (i.e., in terms of structural similarity or chemical formula/sequence similarity) of substrates and products[1] in the pathway graphs.

Another technique [11] overcomes this disadvantage by combining sequence information of substrates and enzymes with graph topology of the underlying pathway. Several algorithms that efficiently perform pairwise pathway comparison are known [8, 9, 11, 17, 19]. One of the popular techniques outlined in [18] is called PathBLAST, which performs pairwise protein-protein interaction network alignment to detect linear paths and clusters [19] that are conserved between different species. This approach incorporates a refined probabilistic model for protein interaction data and also includes an automatic system for laying out and visualizing the resulting

[1] Products in a pathway are chemical compounds

conserved subnetworks. This method is useful in evolutionary analysis by comparing the same pathway from different organisms, but may not scale efficiently to search large databases as it performs pair wise comparison of the pathways and does not describe any graph indexing techniques.

The problem of graph indexing is a critical problem in the field of computational biology. Several efficient graph indexing techniques [20, 21, 22, 23, and 24] for semi structured/XML databases and complex graph databases have been proposed. DataGuide [20] describes efficient index techniques for path expressions and Apex [21] considers the adaptivity of index structure to fit the query load. Another popular XML indexing technique called HOPI [22] provides support for path expression search with wildcards. However, these techniques are optimal and more suitable for path expressions and tree-structured data than for arbitrary graph queries.

There has been a variety of prior work that has used paths to index graphs, including Shasha et al. [23] and the Daylight System [25]. More recently, Han et. al. [24] have used labels attached to frequent subgraphs to index graphs.

3 Background

Definition. A metabolic pathway is a *series of 2 or more interconnected enzyme-mediated (or spontaneous) chemical reactions* that take place in a cell. A chemical reaction consists of one or more substrates (chemical compounds) transforming into one or more products (chemical compounds) via an enzyme (represented with an Enzyme Commission Number or ECN). The basic structure of a bio-chemical pathway is shown in Figure 1.

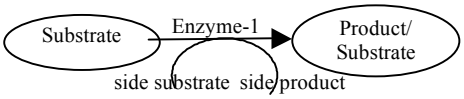

Fig. 1. Basic structure of metabolic pathway

Based on their functionality, pathways are classified into the following:

1. *Metabolic pathways*: consist of a series of chemical reactions occurring in an organism for energy production, synthesis of carbohydrates, etc. For example, photosynthesis.
2. *Signaling pathways*: consists of chemical reactions for information transmission and processing.
3. *Protein Interaction networks*: used to record pairs of proteins, which are experimentally observed to interact with each other.
4. *Gene regulatory networks*: orchestrate the level of expression for each gene in the genome by controlling whether and how vigorously that gene will be transcribed into RNA.

We will be chiefly dealing with metabolic pathways as most metabolic pathways in different organisms have been identified and a large collection of them can be found in various pathway databases. Also, the concepts developed here can be extended to

index protein interaction networks, signaling pathways, gene regulatory networks and other directed or undirected graphs.

Metabolic Pathway Databases. There are several metabolic pathway databases [1], each of which stores thousands of pathways for different organisms. Differences between the databases are in their source of information, classification of pathways for organisms, level of detail, graph representations, etc. Metabolic pathway databases include: KEGG [2], ENZYME [3], BRENDA [4] and EcoCyc/BioCyc/MetaCyc [5].

In this paper, we primarily focus on MetaCyc. The MetaCyc database consists of pathways, reactions, enzymes, substrates and citation to source. It also contains Super-pathways: i.e., groups of pathways linked by common substrates. Pathways are represented as directed graphs with nodes for each enzyme and substrate; graph edges connect substrates. The BioCyc database also has a web-interface to retrieve pathways, given a single chemical compound name, Enzyme name or EC number or pathway name.

4 Terms and Definitions

4.1 Uniquely Labeled Graph

By a uniquely labeled directed graph, we mean a directed graph in which there is a unique label attached to each node in the graph. Figure 2 is a uniquely labeled directed graph. As we will see below in Section 4.5.2, there are a number of reasons that it is difficult to associate uniquely labeled graphs with pathways.

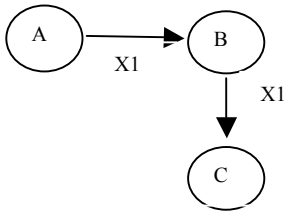

Fig. 2. Uniquely labeled directed graph

4.2 From Directed to Undirected Graphs

Graphs representing metabolic pathways are generally directed. To compute the common subgraph between the query and the pathways in the database, we first enumerate all the directed edges in common between the query and each of the indexed pathways. Next we form the corresponding undirected graph from the set of common edges for each, as in Figure 3. Hence, we are not losing the directionality information, it is already being considered in the index.

A directed graph is said to be weakly connected if its undirected version is connected, i.e., there is a path from each vertex to any other vertex in its undirected version.

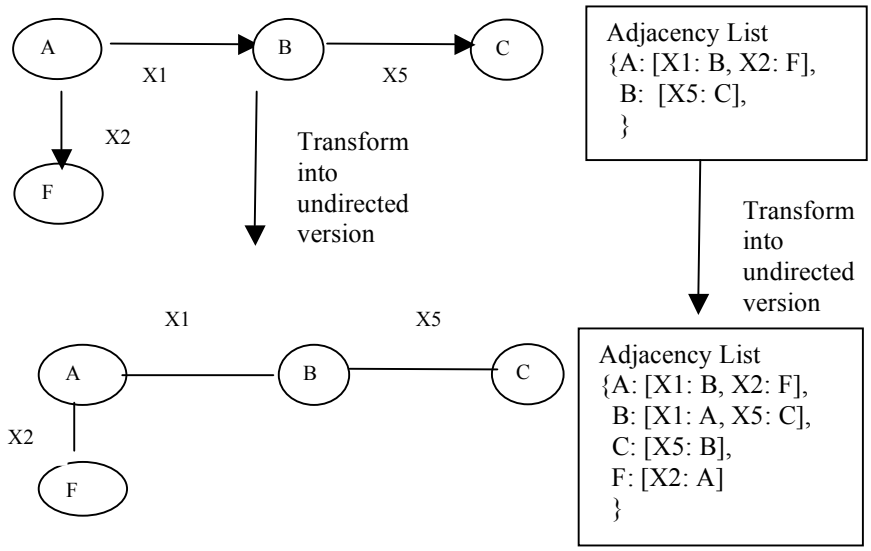

Fig. 3. Weakly connected directed graph

4.3 Common Subgraph

A common subgraph between two directed graphs Q and P represents a subgraph in common between Q and P. For example, the common subgraphs between P and Q in Figure 4 are {A:X1:B, A:X2:F} and {C:X3:D}.

We compute the common subgraphs between two pathways by first enumerating all directed edges in common between them. Second, to obtain the component subgraphs in common from this set of edges, we ignore the directions of these edges (as the directions have already been considered in the previous step) and compute the connected components formed. Each of these connected components is a common subgraph between the two pathways. For example: The set of edges (A:X1:B, A:X2:F, C:X3:D) in common between P and Q form two common subgraphs {A:X1:B, A:X2:F} and {C:X3:D}.

4.4 Maximal Common Subgraph

A common subgraph that has the maximum number of edges is called a Maximal Common Subgraph. In Figure 4, out of the two subgraphs in common {A:X1:B, A:X2:F} is maximal.

4.5 Metabolic Pathways as uniquely labeled directed graphs.

4.5.1 Motivation. Metabolic pathways are representations of reactions. In general, there is no unique way to describe a reaction. For example, different researchers may expand or contract the definition of a reaction by incorporating more compounds or fewer compounds respectively. For this reason, graphs that abstract reactions are not unique. In contrast, graphs that represent chemical structures are unique. Even so, by coding pathways as uniquely labeled graphs, we can speed up many common pathway queries, which is the point of view we take in this paper.

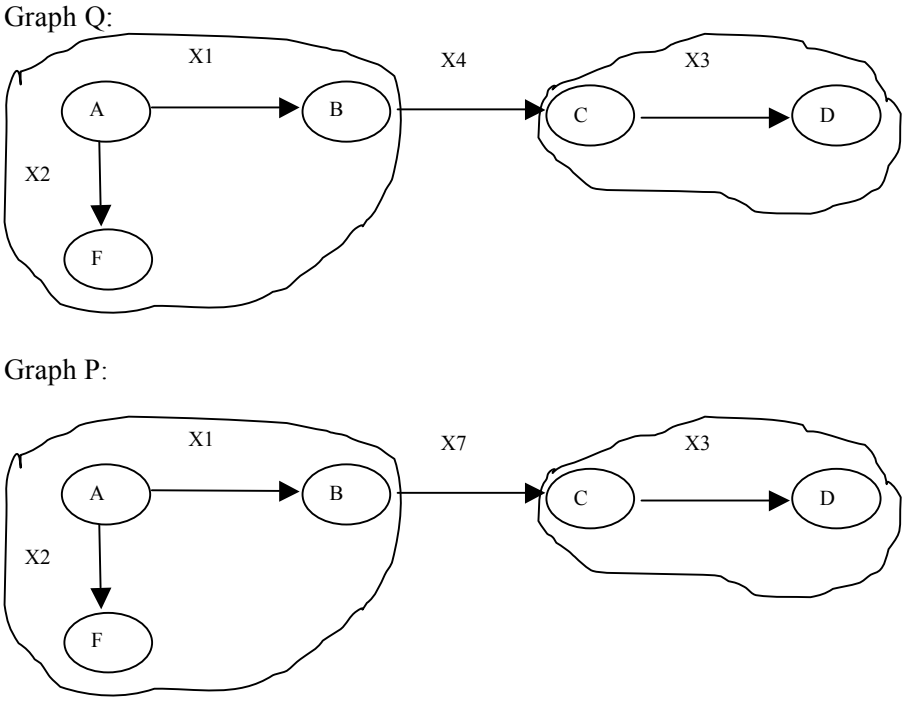

Fig. 4. The common subgraphs between graph P and graph Q are {A:X1:B, A:X2:F} and {C:X3:D}

4.5.2 The labeled graph associated with a pathway. We associate a labeled graph to a pathway as follows:

1) The nodes of the graph are the chemical compounds in the pathway.
2) Two nodes are connected by an edge when there is a chemical reaction in the pathway transforming one node into the other.
3) The edge is labeled with the enzyme using the EC number of the enzyme.

For the study described in this paper, we perform the following two additional steps:

4) We label the nodes with Canonical SMILES string of the chemical compound associated with the node. The Canonical SMILES string can be

obtained from the PubChem database [6].

5) We identify all nodes whose labels are the same. That is, if 1) – 4) above define the graph G, then the graph associated with the pathway is the graph $G' = G / \sim$, where $\sim$ is the equivalence relation defined as follows: $u \sim v$ in case the nodes u and v in G have the same label defined in Step 4.

Figure 5 illustrates Steps 1) – 4) above. The names of the compounds in the nodes have been skipped for brevity. Instead, just letters A, B, C are used to denote different compound names.

Some remarks about this construction:

1. Although Canonical SMILES are not unique [16], for the purposes of the work here, they are adequate.
2. This construction ignores side substrates and side products, although these are easy to include by simply incorporating them into the label associated with an edge.
3. Working with the quotient graph defined in Step 5) is not necessary for the algorithm we describe below, but simplifies the algorithm and the code a bit and was used in this study. The quotient graph is uniquely labeled as defined above in the sense that the labels of the graph are all unique. We emphasize though that the algorithm and approach do not require the use of uniquely labeled graphs.
4. Some pathways in MetaCyc could not be converted into a labeled graph using these steps since either the Canonical SMILES strings for some compounds were not available from PubChem or the complete 4-digit EC numbers were not available from MetaCyc. However there are relatively few pathways like this. We note that the methods in [15] could be used to assign unique labels for chemical compounds.

4.6 Concept of a term for a pathway: A term is an ordered-triplet consisting of a substrate, enzyme and product, which we denote as follows:

substrate:enzyme:product. **(term)**

Note that a term represents an edge in the uniquely labeled graph of the pathway, or one or more edges in the labeled graph associated with a pathway. Because of this, we use "term" and "edge" synonymously below. For example: C=C1CN:2.4.8.1:C2C3=SC is a term (refer to the above pathway in Figure 5). Terms will be used below to build an index structure, which is an inverted file.

4.7 Pathway Vector: Given a labeled graph and an ordering of terms (for example, a lexigraphical ordering), one can associate a vector with the pathway. The component of the vector associated with a term is simply the number of times the term occurs in the graph. Note that if the graph is uniquely labeled, then the pathway vector is a binary vector in the sense that each component is either 0 or 1.

As we will see below, treating pathways as vectors of terms and using inverted files provides a natural index. For the pathways in Figure 4 the vectors are as below, the top row represents the terms of the pathway graphs.

A:X1:B, A:X2:F , B:X4:C, B:X7:C, C:X3:D
Q(1 , 1 , 1 , 0 , 1)
A:X1:B, A:X2:F , B:X4:C, B:X7:C, C:X3:D
P(1 , 1 , 0 , 1 , 1)

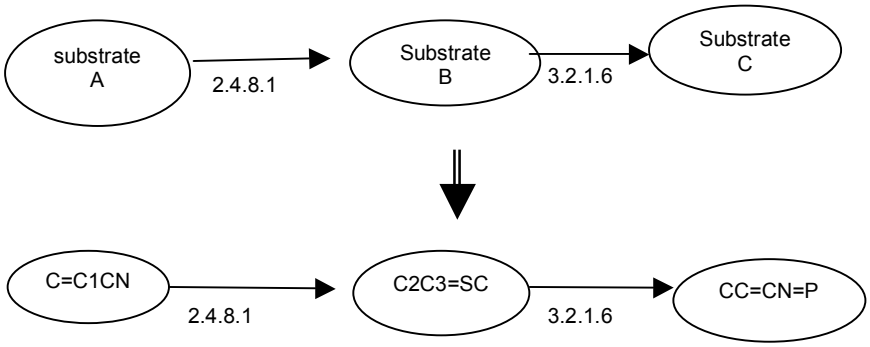

Fig. 5. Transformation of a pathway into uniquely labeled directed graph representation

4.8 Pathway Adjacency List: To capture the structure of the pathway, we can represent it using an adjacency list, which for each node 'p' in an undirected graph, stores a list consisting of nodes that are incident at 'p', together with the label of the corresponding edges. For the undirected pathway in Figure 3 the adjacency list is to the left in Figure 6. The adjacency list for the node with label 'C' consists of a single adjacent node 'B' via an edge with label 'X5'. The adjacency list representation can be converted into a vector and vice versa. Similarly, for a directed graph the adjacency list for each node 'p' is a set of nodes that are pointed to by 'p'. For example, the adjacency list for the directed pathway in Figure 3 is to the right in Figure 6.

Adjacency List
{A : [X1:B, X2:F],
B : [X1:A, X5:C],
C : [X5:B],
F : [X2:A]
}

Adjacency List
{A : [X1:B, X2:F],
B : [X5:C],
}

Fig. 6. Left: Adjacency list for undirected graph in Fig 3. Right: Adjacency list for the directed graph in Fig 3.

4.9 Similarity functions

4.9.1 Cosine Similarity function: Given two pathway vectors 'Q' and 'G' the measure of cosine similarity between them can be computed as in Equation 1. The cosine similarity is a measure of the number of terms/edges in common between the pathway vector 'Q' and vector 'G' [Salton and McGrill 1983].

$$F(Q,G) = \frac{\sum_{i}^{n} q_i G_i}{\sqrt{\sum q_i^2}\sqrt{\sum G_i^2}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

where Q = (q_1, q_2,……,q_n) and G = (G_1, G_2,……,G_n)

4.9.2 Global Similarity measure based on MCS: Given two pathway graphs 'Q' and 'G' the measure of similarity between them based on MCS can be computed as in Equation 2. This is based on the distance metric described in [7], which states that the distance metric d(Q, G) = 1 - Sim(Q, G) is a metric, i.e., for any graphs G_1, G_2 and G_3 the following properties hold true:

1. $0 \leq d(G_1, G_2) \leq 1$,
2. $d(G_1, G_2) = 0 \Leftrightarrow G_1$ and G_2 are isomorphic to each other,
3. $d(G_1, G_2) = d(G_2, G_1)$,
4. $d(G_1, G_3) \leq d(G_1, G_2) + d(G_2, G_3)$.

$$Sim(Q, G) = \frac{|mcs(Q, G)|}{Max(|Q|, |G|)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2)$$

Where mcs(Q, G) is the Maximal Common Subgraph between Q and G and |G| is the size of the graph in terms of number of edges (E) in the graph.

5 Algorithms

The following sections describe in detail the algorithms employed for preprocessing the database, building an index structure for the database, and for searching and displaying pathways similar to the user input query pathway in descending order of similarity.

5.1 Database preprocessing: The database preprocessing consists of the following two steps:

- Transforming the pathways stored in the database to a labeled graph.
- Building of the index structure for the database, which indexes each term or edge consisting of a triplet of the form substrate:enzyme:product

5.2 Database Index structure.

We construct an indexed structure for the database that indexes pathways based on terms/edges present in them. Hence for each term/edge in the index structure, there will be a list of pathways that contain that term/edge, i.e., a list of pathways that contain the corresponding edge. Each pathway has a unique identifier that is stored in the index structure. The size of the pathway is computed and stored as well. An example is given in Figure 7.

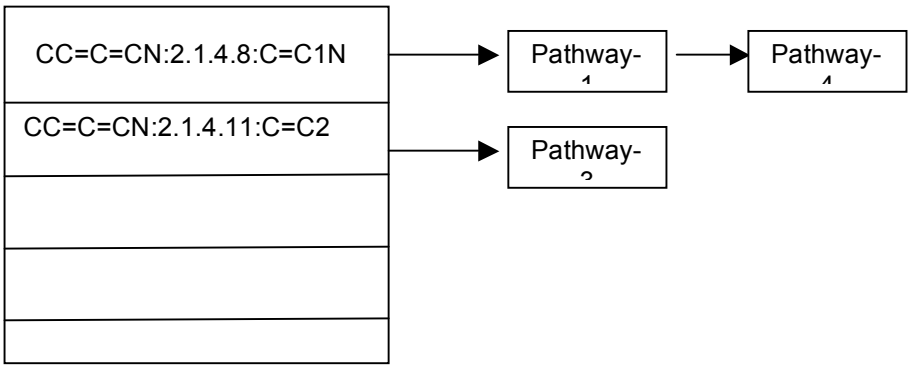

Fig. 7. Example index structure

5.3 Single pass algorithm for indexing the database

Here is the pseudo-code of the single pass algorithm for generating an index structure for a database of pathways to facilitate pathway queries:

```
//database index structure indexes terms/edges and stores all
//the pathways that have the term as a list pointed to by the
//term.
Define Database_Index_Structure as a hash table

//temporary data structure which records the unique canonical
//string for a given compound name obtained from the PubChem
//database. This facilitates a quick local check to see if a
//unique string for the compound was obtained from
//PubChem previously and reduces the number of calls to the
//PubChem database.
Define Compound_Names as a hash table

For each pathway in the DB do
{
   Pathway_Vector = ( )
   Begin a Breadth First Traversal
   For each edge(substrate:enzyme:product) encountered in the
   pathway traversal do
   {
     Term = ''
     For each substrate in the edge do
     If compound does not have unique string in
          Compound_Names Then {
                  Connect to PubChem, obtain unique string
                  Insert entry into Compound_Names
     }
     Append substrate unique string to the Term
     Append the enzyme EC number to the Term
     Append product unique string to the Term
     Insert term into Database_Index_Structure along
               with Pathway id
     Append term to Pathway_Vector
  }
 }
 Return Database_Index_Structure and Compound_Names
```

5.4 Search Engine Functionality.

5.4.1 Given a query pathway; for each of the substrates/products in the query retrieve the SMILES string from the PubChem database.

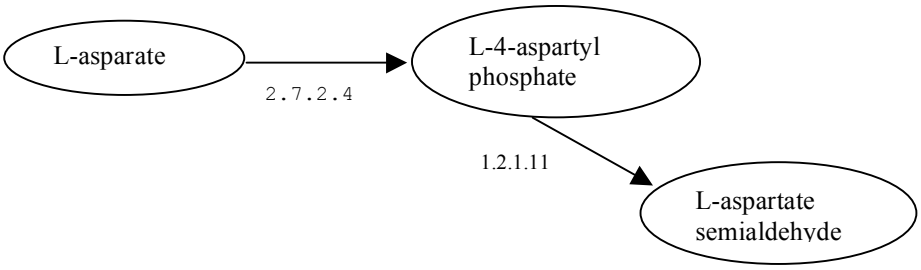

Fig. 8. Example user query to be retrieved from MetaCyc database

For the query in Figure 8: We submit the names L-asparate, L-4-aspartyl phosphate and L-asparate semialdehyde to PubChem to obtain strings:

(1) L-asparate – C(C(C(=O)O)N)C(=O)O
(2) L-4-aspartyl phosphate – C(C(C(=O)O)N)C(=O)OP(=O)(O)O
(3) L-asparate semialdehyde - C(C=O)C(C(=O)O)N

5.4.2 Submit the new query graph represented in the form of a labeled graph (as shown in Figure 9 for the query graph) to the search engine, which will retrieve all the similar pathway graphs from the given database, e.g., MetaCyc, and display them in descending order of the similarity measure calculated using a similarity metric chosen by the user, i.e., either based on maximal common subgraphs or cosine similarity. Details of the algorithm are given in the next section.

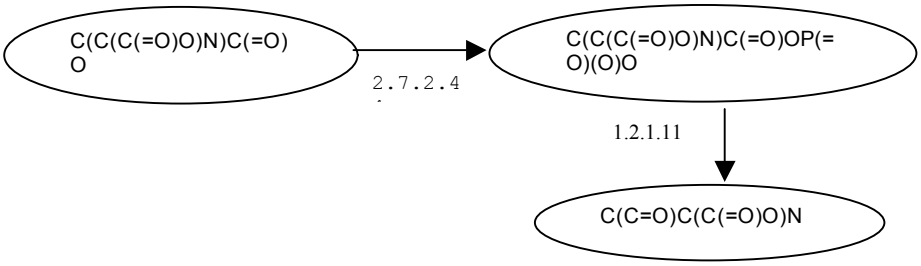

Fig. 9. Labeled graph representation for graph in Fig. 8

Q = { A:5.3.1.9:B, B:10.10.10.10:C, C:2.7.2.3:D, D:5.4.2.1:E, E:4.2.1.11:F, F:2.7.1.40:G }

P1 = { A:5.3.1.9:B, B:2.7.1.11:H, H:4.1.2.13:I, H:4.1.2.13:J, J:5.3.1.1:I, I:1.2.1.12:C, C:2.7.2.3:D, D:5.4.2.1:E, E:4.2.1.11:F , F:2.7.1.40:G }
P2 = { K:2.7.1.147:A, A:5.3.1.9:B, B:2.7.1.146:H, H:4.1.2.13:I, H:4.1.2.13:J, I:X:D, J:5.3.1.1:I, G:2.6.1.2:L, G:1.2.7.1:M , M:6.2.1.13:N, D:5.4.2.1:E, E:4.2.1.11:F, F:2.7.1.40:G }

Fig. 10. Example query graph 'Q' and example database with two pathways 'P1' and 'P2'.

5.5 Algorithm for computing similarity based on MCS and Cosine Similarity: The words edges and terms are used synonymously.

Overview. The important tasks performed by the algorithm are as follows:

Step 1.1 For each edge given in the query pathway; find all the database pathways that have the edge.

Step 1.2 For each pathway obtained in Step 1.1; find all the common edges between the pathway and the query graph.

Step 1.3 Represent the common edges in the form of adjacency lists (refer to section 4.8 for details) for the undirected graph (this is required to obtain the maximal common subgraph defined as a connected component). The common edges for the graphs 'P1' and 'P2' of Figure 10 are shown in Figure 11.

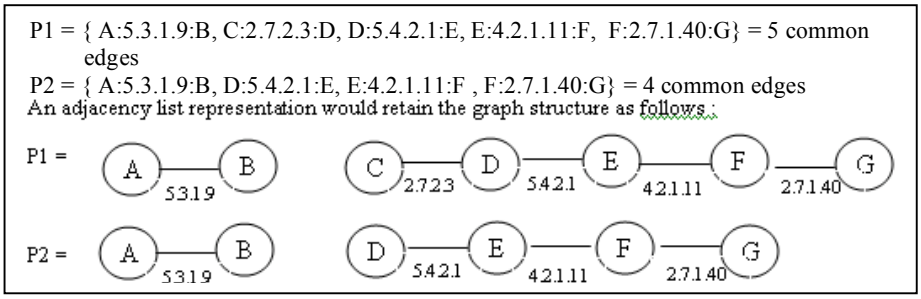

Fig. 11. Common edges after Step 1 for the example in Fig 10.

Step 2. For each pathway with common edges found above, perform a simple Depth First Traversal (DFT) on the undirected graph obtained in Step 1. The connected components (trees) obtained in the Depth First Traversal forest will represent the common subgraphs between Q and the pathway. The common subgraphs/connected components obtained from Figure 11 are shown in Figure 12.

Step 3. From the above computed common subgraphs, find a maximal subgraph and use it to compute the similarity measure based on Eq 2 (MCS similarity). Use the set of common edges to compute the similarity measure based on Eq 1 (Cosine Similarity). Rank the pathways in descending order of similarity based on the similarity measure chosen by the user.

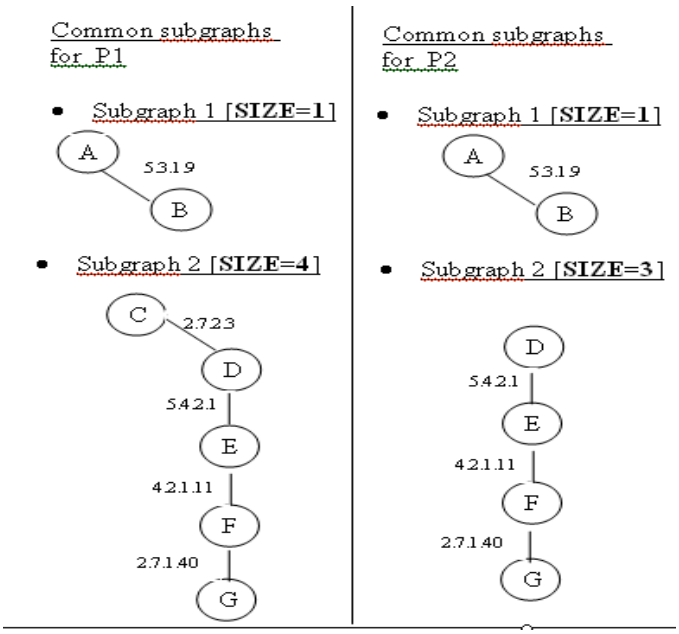

Fig. 12. Common subgraphs found for 'P1' and 'P2'

Time Complexity Analysis for the steps described above:

Step 1.1: For the i'th edge in the query graph, let n_i be the number of pathways that have the edge. Using the index associated with this edge, all pathways having this edge can be obtained. Thus, all edges in common between the database pathways and the query can be obtained in time O((sum over all edges in the query) n_i) = O(n), where n is the number of such common edges.

Step 1.2: Since the index associated with each edge has the IDs of the pathways having the edge, all edges the pathway has in common with the query can be assembled in linear time. Thus, this sub-step also takes O(n) time.

Step 1.3: This is the same as Step 1.2, except that the representation of the edges is in the form of adjacency lists. This takes O(n) time.

Step 2: The depth-first algorithm to find all connected components based on adjacency lists can be obtained in linear time, i.e., O(n) [14].

Step 3: During the computation of the connected components in Step 2, we keep track of the sizes of the connected components for each database pathway that has common edges with the query. Every time that an edge is added to a connected component, its size is increased by 1. Thus, the time to compute the sizes of the connected components is bounded by O(n).

The sizes of the database pathways are pre-computed in Algorithm 4.3 and stored. Thus, the time to compute the two similarity measures is bounded by O(n + Q) = O(n), assuming that each edge in the query Q occurs in some database pathway.

Hence, the search time/retrieval time given a query pathway graph is linear in the total number of edges (n) in common with the query in the entire database.

6 Experimental Results

We performed some preliminary experimental studies and verified that the performance is approximately linear. We have implemented the algorithms in python and have provided a PHP [13] web interface to the search engine that can be accessed at [10]. We ported the search engine onto a web server on an Intel® Xeon™ CPU 2.4GHz and 1GB RAM. We analyzed pathway query graphs of different sizes, and plotted the algorithm's performance as shown in Figure 13 for the MetaCyc database.

The Metacyc dataset we used has 547 pathways, 4955 enzymatic reactions, 1940 enzymes and 3551 chemical compounds. The total number of unique edges in the database that are computed and stored in the index is 2294. The total number of edges in the entire database is 41561 and the average pathway size is 78 edges. The maximum length of the list of pathways corresponding to an edge in the index is 41.

The time to compute the index was 11.77 seconds. Since this was a very small database the queries of varying sizes were selected at random to target different sets of pathways in the database. Also, queries consisting of most occurring edges and least occurring edges were formed.

We are working on extending our algorithm to larger databases such as KEGG [2] and EMP[26] to test its scalability and performance. We are also currently working on performing a comparative analysis of our algorithm with other popular graph indexing techniques like GraphGrep [23] or GIndex [24].

No. of input edges	Total No. of common edges in the database (**X axis**)	No. of output pathways	Retrieval time in secs (**Y axis**)
1	2	2	0.00075
1	6	6	0.00088
1	13	13	0.00124
3	28	16	0.00181
3	40	17	0.00241
6	55	21	0.00332
7	61	26	0.00372
8	66	28	0.0041
9	72	34	0.0046
10	84	27	0.0057

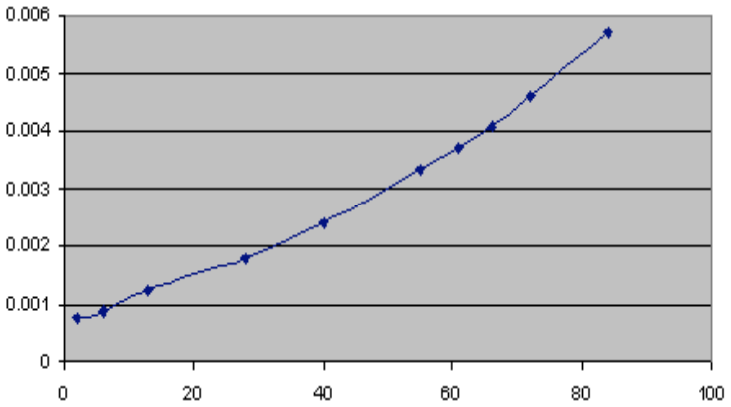

Fig. 13. Bottom: X-axis: total no. of edges in common with the query in the entire database, Y-axis: retrieval time in secs. Top: tabular description of performance.

7 Summary and Conclusion

Finding similar pathways has important applications in drug discovery and in the study of evolution. For this reason, it is important to develop efficient techniques to compute pathway similarity. In this paper, we have introduced an algorithm for retrieving similar pathways that uses an inverted file for the pathway database and indexes all the pathways in the database based on their edges. In this way, we are able to find and rank all the pathways similar to the user input query pathway in linear time.

We have implemented this algorithm using data from the MetaCyc database and provided a web interface. We have also performed a preliminary experimental analysis showing that the queries are indeed linear as expected.

The study in this paper assumed that the graph associated with a pathway was uniquely labeled. This is easily accomplished by working with the quotient graph $G,'$ defined above, instead of the graph G. Essentially the same algorithm described in Section 4 works for the graph G – the only difference is that the components of the pathway vector are not restricted to 0 or 1, but can be any positive integer. In practice, working with the quotient graph G' is not an important restriction and speeds up the computations. We are currently studying the trade-offs involved when using the graph G instead of the graph G'.

We are also currently integrating additional pathway databases so that similarity searching can be done across multiple distributed pathway databases. We are also studying the impact of other graph properties such as edge to node ratios, number of cliques, etc. on the query performance.

References

1. Bader GD, Cary MP, Sander C. Pathguide: a pathway resource list. Nucleic Acids Res. 2006 Jan 1;34(Database issue):D504-6.
2. KEGG - Kanehisa, M., Goto, S., Hattori, M.,Aoki-Kinoshita, K.F., Itoh, M., Kawashima, S.,Katayama, T., Araki, M., and Hirakawa, M.; From genomics to chemical genomics: new developments in KEGG. Nucleic Acids Res. 34, D354-357 (2006).
3. Bairoch A. The ENZYME database in 2000 Nucleic Acids Res 28:304-305(2000).

4. BRENDA, enzyme data and metabolic information Schomburg, I., Chang, A., Schomburg, D. Nucleic Acids Res. (2002) 30, 7-9
5. MetaCyc - Cynthia J. Krieger, Peifen Zhang, Lukas A. Mueller, Alfred Wang, Suzanne Paley, Martha Arnaud, John Pick, Seung Y. Rhee, and Peter D. Karp (2004) MetaCyc: A Multiorganism Database of Metabolic Pathways and Enzymes Nucleic Acids Research, 32(1):D438-42.
6. PubChem database : http://pubchem.ncbi.nlm.nih.gov/
7. Horst Bunke , Kim Shearer, A graph distance metric based on the maximal common subgraph, Pattern Recognition Letters, v.19 n.3-4, p.255-259, March 1998
8. Ming Chen, Ralf Hofestaedt, PathAligner: Metabolic Pathway Retrieval and Alignment, *Applied Bioinformatics*, 2004, 3(4): 241-252.
9. Ron Pinter et al. - Tree-based Comparison of Metabolic Pathways
10. Metabolic Pathway Search Engine - http://data.dataspaceweb.net/pathways/Search.php
11. Forst CV, Schulten K. Evolution of metabolisms: a new method for the comparison of metabolic pathways using genomics information.. J Comput Biol. 1999 Fall-Winter;6(3-4):343-60
12. EC-Published in Enzyme Nomenclature 1992 [Academic Press, San Diego, California, ISBN 0-12-227164-5 (hardback), 0-12-227165-3 (paperback)] with Supplement 1 (1993), Supplement 2 (1994), Supplement 3 (1995), Supplement 4 (1997) and Supplement 5 (in Eur. J.Biochem. 1994, 223, 1-5; Eur. J. Biochem. 1995, 232, 1-6; Eur. J. Biochem. 1996, 237, 1-5; Eur. J. Biochem. 1997, 250; 1-6, and Eur. J. Biochem. 1999, 264, 610-650; respectively) [Copyright IUBMB].
13. Rasmus Lerdorf, Kevin Tatroe. Programming PHP. Published: 05/04/2002 ISBN: 1565926102
14. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. Introduction to Algorithms, Second Edition. Section 22.3 Depth First Search.
15. Robert L. Grossman, Pavan Kasturi, Donald Hamelberg, Bing Liu, An Empirical Study of the Universal Chemical Key Algorithm for Assigning Unique Keys to Chemical Compounds, Journal of Bioinformatics and Computational Biology, 2004, Volume 2, Number 1, 2004, pages 155-171.
16. Greeshma Neglur, Robert L. Grossman, Bing Liu: Assigning Unique Keys to Chemical Compounds for Data Integration: Some Interesting Counter Examples. DILS 2005: 145-157.
17. Kelley, B. P., Sharan, R., Karp, R., Sittler, E. T., Root, D. E., Stockwell, B. R., and Ideker, T. Conserved pathways within bacteria and yeast as revealed by global protein network alignment. Proc Natl Acad Sci U S A 100, 11394-9 (2003).
18. Kelley, B. P., Yuan, B., Lewitter, F., Sharan, R., Stockwell, B. R., and Ideker T. PathBLAST: a tool for alignment of protein interaction networks. Nucleic Acids Res.32(Web Server issue):W83-8. 2004.
19. Sharan, R., Suthram, S., Kelley, R. M., Kuhn, T., McCuine, S., Uetz, P., Sittler, T., Karp, R. M., and Ideker, T. Conserved patterns of protein interaction in multiple species. Proc Natl Acad Sci U S A. 8:102(6) 1974-79 (2005).
20. R. Goldman and J. Widom. Dataguides:enabling query formulation and optimization in semistructured databases. In Proceedings of VLDB, pages 436--445, 1997.
21. C.-W. Chung, J.-K. Min, and K. Shim. Apex: an adaptive path index for XML data. In SIGMOD, 121--132, 2002.
22. Ralf Schenkel, Anja Theobald, Gerhard Weikum, "Efficient Creation and Incremental Maintenance of the HOPI Index for Complex XML Document Collections," icde, pp. 360-371, 21st International Conference on Data Engineering (ICDE'05), 2005.
23. D. Shasha, J. T. L. Wang, and R. Giugno. Algorithmics and applications of tree and graph searching. In Symposium on Principles of Database Systems, pages 39--52, 2002.
24. X. Yan, P. S. Yu, and J. Han. Graph indexing: A frequent structure based approach. In Proceedings of SIGMOD 2004.
25. C.A. James, D. Weininger, and J. Delany. Daylight theory manual daylight version 4.82. Daylight Chemical Information Systems, Inc, 2003.

26. E Selkov, S Basmanova, T Gaasterland, I Goryanin, Y Gretchkin, N Maltsev, V Nenashev, R Overbeek, E Panyushkina, L Pronevitch, E Selkov, Jr, and I Yunus. The metabolic pathway collection from EMP: the enzymes and metabolic pathways database: Nucleic Acids Res. 1996 January 1; 24(1): 26–28.