Systems biology

Simple and fast alignment of metabolic pathways by exploiting local diversity

Sebastian Wernicke and Florian Rasche*

Institut für Informatik, Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2, 07743 Jena, Germany

Received on February 17, 2007; revised on April 28, 2007; accepted on May 17, 2007
Advance Access publication May 31, 2007
Associate Editor: Jonathan Wren

ABSTRACT

Motivation: An important tool for analyzing biological networks is the ability to perform homology searches, i.e. given a pattern network one would like to be able to search for occurrences of similar (sub)networks within a set of host networks. In the context of metabolic pathways, Pinter et al. [Bioinformatics, 2005] proposed to solve this computationally hard problem by restricting it to the case where both the pattern and host networks are trees. This restriction, however, severely limits the applicability of their algorithm.

Results: We propose a very fast and simple algorithm for the alignment of metabolic pathways that does not restrict the topology of the host or pattern network in any way; instead, our algorithm exploits a natural property of metabolic networks that we call ‘local diversity property’. Experiments on a test bed of metabolic pathways from the BioCyc database indicate that our algorithm is much faster than the restricted algorithm of Pinter et al.—the metabolic pathways of two organisms can be aligned in mere seconds—and yet has a wider range of applicability and yields new biological insights. Our ideas can likely be extended to work for the alignment of various types of biological networks other than metabolic pathways.

Availability: Our algorithm has been implemented in C++ as a user-friendly metabolic pathway alignment tool called METAPAT. The tool runs under Linux or Windows and can be downloaded at http://theinf1.informatik.uni-jena.de/metapat/

Contact: florian.rasche@uni-jena.de

Supplementary information: Supplementary data are available at bioinformatics online.

1 INTRODUCTION

Studies of biological sequences are fruitful in many areas of bioinformatics, ranging from the analysis and prediction of molecular function to large-scale evolutionary (e.g. Durbin et al., 1999; Mount, 2004). Shifting attention from linear data to more complex functions and interactions, recent years have seen a surge in the availability of non-sequential biological network data.

1.1 Motivation

Analogous to sequences, an important tool for analyzing biological networks data is being able to perform comparative studies, i.e. to be able to search for homologous (sub)networks to a given pattern network. The ability to efficiently compare biological networks promises to be useful, e.g. for interaction predictions, functional annotation, data integration, knowledge transfers and for developing a better understanding of biological network organization (Kelley et al., 2004; Sharan and Ideker, 2006). In a recent survey, Sharan and Ideker (2006) advance the opinion that ‘network comparison techniques promise to take a leading role in bioinformatics […] in elucidating network organization, function and evolution’.

Unfortunately, the task of performing a network homology search turns out to be quite hard as it can be traced back to the NP-complete SUBGRAPH ISOMORPHISM problem, i.e. an algorithm that aligns networks in a biologically meaningful way can usually also solve SUBGRAPH ISOMORPHISM.

SUBGRAPH ISOMORPHISM

Input: Two graphs \( G_p \) (the pattern) and \( G_H \) (the host).

Task: Find whether \( G_H \) contains a subgraph that is isomorphic to \( G_p \).

This problem remains NP-complete even when restricted to graph classes that usually render NP-complete problems tractable, for instance, if the pattern is a forest and the host is a tree or if the pattern is a tree and the host has bounded treewidth (Garey and Johnson, 1979). Polynomial-time algorithms are only known if the host has bounded treewidth \( \omega \) and the pattern has either a high connectivity or bounded degree; in these cases, SUBGRAPH ISOMORPHISM can be solved in \( O(n_p^{\omega+1} \cdot n_H) \) time for an \( n_p \)-vertex pattern and \( n_H \)-vertex host (Dessmark et al., 2000; Hajiaghayi and Nishimura, 2002; Matoušek and Thomas, 1992).

1.2 Previous approaches

To overcome the hardness of SUBGRAPH ISOMORPHISM when performing a network homology search on biological networks, various algorithms have been proposed:

- For protein interaction networks, Kelley et al. (2004) presented an algorithm called PATHBLAST that, given a linear pathway as a query, randomly decomposes the host...
graph into linear pathways to find homologous pathways among these.

- For linear patterns, Shlomi et al. (2006) proposed an algorithm that is based on random graph colorings.
- Pinter et al. (2005) presented a homology search algorithm for metabolic pathways that is based on restricting the host and pattern graph to be trees, in which case polynomial-time algorithms are possible (Pinter et al., 2004).

Notably, all of these algorithms basically take the same approach: they make network homology searches algorithmically feasible by restricting the topology of the pattern network and sometimes also the host networks to be cycle-free. This, in turn, causes all of these algorithms to suffer from basically the same problems, namely at least one of the following three:

1. **Limited applicability.** Many biological networks of interest contain cycles. Hence, they cannot be aligned by these algorithms per se.
2. **Long running time.** To deal with cycles, the input networks can be automatically decomposed into one or more cycle-free subgraphs. Whether these decompositions are randomized or deterministic, many of them are necessary to ensure that all good matches for the pattern are found. This leads to exponential running times that limit the applicability of the algorithm: e.g. the PATHBLAST algorithm of Kelley et al. (2004) requires $O(\ell!)$ runs for a pattern that is a simple length-$\ell$ path, effectively limiting the size of feasible patterns to about six vertices.
3. **Requirement of manual labor and expert knowledge.** As an alternative to automatic decompositions, one can use expert knowledge and manually decompose the input networks into cycle-free subgraphs. Such an approach was chosen by Pinter et al. (2005) to obtain the dataset for their pathway alignment tool (besides excluding some of the networks that have cycles) (Rokhlenko, 2006, personal communication). Such a process seems tedious, error prone and not always applicable (e.g. when little information is known about a network beforehand or we have no idea what the result should roughly look like).

Intriguingly, there is another thing that the existing network homology search algorithms have in common besides taking a common approach of topological restriction: they do not algorithmically expose the fact that vertices in biological networks are functionally labeled. Rather, the vertex labels are used only for scoring the similarity of the pattern to various subgraphs in the host.

In the context of metabolic pathway alignment, this work proposes an approach that does expose vertex labels in order to obtain a network alignment algorithm that is simple, fast and imposes no topological restrictions upon the input networks.

### 1.3 Organization of this work

After introducing some notation, Section 2 presents our new alignment algorithm for metabolic pathways in three steps: first, Subsection 2.1 formalizes our network alignment problem and presents a simple—yet impractical—algorithm for it called MATCH. Second, Subsection 2.2 introduces the local diversity property of metabolic networks which, third, is exploited in Subsection 2.3 by slightly modifying the MATCH algorithm so as to obtain our new metabolic pathway alignment algorithm FIT-MATCH.

The FIT-MATCH algorithm has been implemented in C++, Section 3 reports experiments with it on a test bed of metabolic pathways from the BioCyc database (Karp et al., 2005). These indicate that FIT-MATCH is much faster than the algorithm of Pinter et al. (2005) and yet—because the topology of the input networks is not restricted—is simpler to use, yields new insights, and has a wider range of applicability. The FIT-MATCH algorithm is the basis for a user-friendly metabolic pathway alignment tool called METAPAT, which is freely available at http://them1.informatik.uni-jena.de/metapat/

Concluding in Section 5, we point out how our concept of local diversity could be transferred to obtain alignment algorithms for other types of biological networks besides metabolic pathways (such as protein interaction networks) and state some open questions that appear to be of interest for future research.

## 2 A NEW FAST AND SIMPLE PATHWAY ALIGNMENT ALGORITHM

We model metabolic pathways as connected directed graphs. Each vertex represents a metabolite and each edge represents an enzyme-catalyzed reaction that has the source metabolite as a reactant and the target metabolite as a product. Each edge is labeled with the Enzyme Commission number (EC number) of the catalyzing enzyme. The pattern for which we seek a homolog is always denoted $G_p=(V_p, E_p)$, the host in which we seek an occurrence of the pattern is denoted $G_h=(V_h, E_h)$.

A vertex with exactly one outgoing and one incoming edge (not counting self loops) is called a path vertex; all other vertices are called branch vertices. Although somewhat counterintuitive, to simplify the overall presentation we chose to use the term ‘branch vertex’ also for vertices with degree one. A path in a graph that—except for its start and end vertex—consists just of path vertices and where every vertex occurs at most once is called simple.

An isomorphism between two graphs $G=(V, E)$ and $G'=(V', E')$ is a one-to-one mapping $\Phi: V \rightarrow V'$ such that $(u, v) \in E \iff (\Phi(u), \Phi(v)) \in E'$ (note that this definition ignores any labels of the vertices and edges). If there exists an isomorphism between two graphs, we call them isomorphic.

---

2 This modeling is different from the extended abstract of this work and the work of Pinter et al. (2005), which both use vertices as representatives of enzymes. There are two reasons for our model choice: first, modeling enzymes as vertices tends to ‘fragment’ the input networks into several unconnected components, leading to an increased number of less meaningful results and a longer average running time. Second, we found that having edges represent enzymes is somewhat more intuitive to handle. As will subsequently become clear in this article, the choice of the model has no significant on the discovery of (biologically meaningful) network alignments with our algorithm.
Two graphs are called homeomorphic if we can subdivide their edges (i.e., edges can be replaced by simple paths of arbitrary length in the same direction) in such a way that the resulting graphs are isomorphic. The corresponding homeomorphism is a function \( \varphi \) that bijectively maps the branch vertices of the two graphs onto each other.

### 2.1 Formalization and a simple backtracking algorithm

In order to formalize the alignment problem we are trying to solve, we must define two things, namely what we mean by 'alignment' and what we view as a 'good' or 'high-scoring' alignment. Concerning a formalization of alignments, Pinter et al. (2005) argue that subgraph homeomorphism is well-suited for aligning metabolic networks because branch vertices are well-conserved whereas paths may be elongated or shortened. We follow this by using the notion of an embedding to formalize an alignment.

**Definition 1.** An embedding of a pattern graph \( G_P \) into a host graph \( G_H \) is a tuple \((\hat{G}_H, \varphi)\) where \( \hat{G}_H \) is a subgraph of \( G_H \) that is homeomorphic to \( G_P \) and \( \varphi \) is a homeomorphism between \( G_H \) and \( G_P \).

We can use this notion to phrase metabolic pathway alignment as a combinatorial problem called **Maximum-Score Embedding**.

**Maximum-Score Embedding**

**Input:** Two directed labeled graphs \( G_P = (V_P, E_P) \) and \( G_H = (V_H, E_H) \).

**Task:** Find the maximum-score embedding of \( G_P \) into \( G_H \).

It remains to define the scoring scheme that we plug into this problem definition. Since topological similarity is already ensured by relying on homeomorphisms, we solely rely on pairwise similarity scores between the vertices. In defining this pairwise score, we again follow Pinter et al. (2005) and make use of a scoring scheme due to Tohsato et al. (2000) which calculates the similarity of two enzymes based on their functional EC numbers: the longer the common prefix of two EC numbers is, the more similar the respective enzymes are considered to be.\(^3\) The scoring scheme also incorporates an information-theoretic consideration, namely that the similarity of two enzymes is more significant the less their common EC number prefix occurs among all enzymes.

**Definition 2.** Let the edges \( e_1 \) and \( e_2 \) represent two enzymes \( E_1 \) and \( E_2 \), respectively. If the lowest common enzyme class of \( E_1 \) and \( E_2 \) as determined by their EC numbers contains \( h \) enzymes, then the similarity of \( e_1 \) and \( e_2 \) is defined as \( \text{sim}(e_1, e_2) := -\log_2 h \).

Using the scoring for pairwise similarity, we can define a similarity score for two simple paths that is based on the notion of a sequence alignment.

**Definition 3.** Given two simple paths \( p_1 = u_1 \ldots u_s \) and \( p_2 = v_1 \ldots v_t \) and a negative gap penalty \( g \), the similarity \( \text{sim}(p_1, p_2, g) \) between \( p_1 \) and \( p_2 \) is defined as the maximum possible score of a sequence alignment between the edge sequence \( (u_1, u_2, \ldots, u_{s-1}, u_s) \) of \( p_1 \) and the edge sequence \( (v_1, v_2, \ldots, v_{t-1}, v_t) \) of \( p_2 \), using \( g \) as the gap penalty and the scoring scheme of Definition 2 to evaluate the pairwise similarities between edges.

To score an embedding of a pattern \( G_P \) into a host \( G_H \), we sum over the similarity scores of the simple paths branch vertices that are mapped onto each other. Two special cases need to be dealt with:

1. If two or more paths connect a pair of branch vertices, it is ambiguous how these paths are to be mapped onto each other. We resolve this by mapping them so as to maximize the overall score of the embedding. This is a maximum bipartite matching problem, which can be solved in polynomial time.

2. If the pattern is a simple cycle, then there are no branch vertices where a simple path could start or end. We resolve this issue by letting two path vertices in the pattern graph and two vertices in the subgraph \( G_P \) of the host graph artificially be considered as branch vertices so as to maximize the resulting alignment score.

For the sake of simplicity in our presentation, we assume from now on that there is at most one simple path between any two branch vertices and that neither the pattern nor the host is a simple cycle. Our implementation of the algorithm does not make any of these restrictions, but handling them explicitly in the remaining theoretical part of this work rather obfuscates the main ideas.

To render the scoring of an embedding more precisely, we use the following definition:

**Definition 4.** Given an embedding \((\hat{G}_H, \varphi)\) of a pattern graph \( G_P \) in a host graph \( G_H \), let \( B(\hat{G}_H) \) denote the branch vertices of \( G_P \). For two branch vertices \( u \) and \( v \) let \( p(u, v) \) be the simple path between \( u \) to \( v \) (including \( u \) and \( v \)); if no such path exists, then \( p(u, v) \) is the empty graph. Given a gap penalty \( g < 0 \), the score of \((\hat{G}_H, \varphi)\) is defined as

\[
\text{score}(\hat{G}_H, \varphi) := \sum_{u, v \in B(\hat{G}_H)} \text{sim}(p(u, v), \varphi(u), \varphi(v), g).
\]

Naïvely, **Maximum-Score Embedding** can be solved by a simple backtracking algorithm that exhaustively explores all possible embeddings of a given pattern graph \( G_P \) into a host graph \( G_H \). Formally, this algorithm is best described by using the notions of a partial embedding and extensions thereof. Basically speaking, a partial embedding of a pattern graph is an embedding of one of its subgraphs into the host graph (as illustrated in Fig. 1), whereas an extension of a partial embedding additionally embeds a previously unembedded path of the pattern graph (as exemplified in Fig. 2).

**Definition 5.** A partial embedding of a pattern graph \( G_P \) into a host graph \( G_H \) is an embedding of a connected subgraph \( G_P' \) of \( G_P \) into \( G_H \), that is, it consists of \( G_P' \), a subgraph \( \hat{G}_H' \) of \( G_H \), and a homeomorphism \( \varphi \) between \( G_P' \) and \( \hat{G}_H' \). A partial embedding is denoted by \((G_P', G_H', \varphi)\) (where \( \varphi \) is the homeomorphism between \( G_P' \) and \( G_H' \)).
Solving MAXIMUM-SCORE EMBEDDING, which we call MATCH.

DEFINITION 6. Let p be a simple path in $G_p$ that connects two branch vertices $u$ and $v$ such that at least one of these branch vertices is in $G_p$ but no path vertex of $p$. An extension of a partial embedding $(G_p, G_H^j)$ by $p$ is a partial embedding of the subgraph induced in $G_p$ by $G_p', u, v,$ and $p$ that is identical to $(G_p, G_H^j)$ when restricted to the vertices of $G_p'$.

We can now describe our naïve backtracking algorithm for solving MAXIMUM-SCORE EMBEDDING, which we call MATCH. A pseudocode description of MATCH is shown in Figure 3. The algorithm starts out by aligning a branch vertex of the pattern to a branch vertex in the host graph and then uses a recursive subprocedure called EXTEND that takes as input a partial embedding and then considers all possible extensions of it, thus enumerating all embeddings of the pattern graph into the host graph.

The running time of MATCH is primarily determined by the number of recursive calls that are made in lines 05 and 04 of the algorithm. While this number is upper bounded by a constant—both the maximum path length and the maximum degree of a metabolic pathway are naturally bounded by some constant for biological reasons—it turns out to be rather large.\(^4\) In our experiments, we have found that if the pattern graph consists of $k$ simple paths, then the size of the search tree that is explored by MATCH is, on average, around $6^k$. Considering that our dataset of metabolic pathways (drawn from the BioCyc database) contained a considerable amount of pathways with more than ten paths, this leads to a very long running time for MATCH.

2.2 The concept of local diversity

As a typical example for a metabolic pathway, consider the anaerobic respiration pathway of *Escherichia coli* that is shown in Figure 4. The following observation can be made here, which is hence crucial to our approach and seems to hold for most metabolic pathways:

**Observation 7.** The sequences of reactions that are given by two paths that have the same starting vertex often carry out very different biological functions.
This observation describes what we refer to as the local diversity property of metabolic networks. There are plausible reasons why a metabolic network is expected to generally have this property: first, most metabolic products offer only very few possibilities where a certain reaction can chemically take place. Second, identical reactions for a certain substrate within a pathway are usually carried out by enzymes with identical EC numbers.

Local diversity is an important property for the algorithmic alignment of metabolic pathways: intuitively, SUBGRAPH ISOMORPHISM is hard because even very different graphs might appear similar based on local information. The local diversity property, however, means that metabolic pathways usually provide very rich and diverse local information that can be exploited to overcome this phenomenon. This is the main idea of our modification to MATCH that we propose in the next subsection in order to render the algorithm efficient in practice.

2.3 Exploiting local diversity

When we compute all extensions of a partial embedding by a path \( p \), some of these might not make sense from a biological perspective: the biological function of \( p \) might not fit the biological function of the path in the host that it is aligned to. The key to making MATCH more efficient is to observe that the local diversity property implies that usually the majority of extensions of a partial embedding does not make sense from a biological perspective. Thus, to exploit local diversity and make MATCH more efficient, we can devise a formal (and biologically meaningful) definition of ‘fitting biological function’ for two given paths and then modify MATCH so that it only explores fitting embeddings.

**Definition 8.** Given a real number \( 0 \leq f \leq 1 \), a gap score \( g \), a simple \( x \)-vertex path \( p_1 \), and a simple \( y \)-vertex path \( p_2 \), we say that \( p_1 \) and \( p_2 \) fit if a maximum-score alignment between them aligns at most \( \min \left( \left\lceil (1-f) \cdot x \right\rceil, \left\lceil (1-f) \cdot y \right\rceil \right) \) vertices to a gap. An extension of a partial embedding \( \left( G_f, G_H, \Phi \right) \) fits if every simple path between two branch vertices \( u, v \in V_f \) fits the corresponding simple path between \( \Phi(u), \Phi(v) \in G_H \).

As an illustration, if we consider a fitting parameter of \( f = 0.50 \), then a four-edge path fits no path that consists of seven or more edges; a higher fitting parameter of \( f = 0.75 \) would cause it to fit no path that consists of six or more edges. For some applications, Definition 8 might be considered too strict in its handling of very short paths: in particular, a one-edge path never fits a length-3 path, regardless of the fitting parameter. One can easily circumvent this by introducing a minimum number of gaps that is always allowed regardless of the path lengths or fitting parameter. In our implementation, for instance, we consider the case ‘one gap is always allowed’.

To exploit local diversity, we now modify MATCH so that it only explores fitting embeddings. For this purpose, lines 05 and E4 in Figure 3 need to be modified so that the EXTEND-subprocedure is only called for fitting extensions. We name the resulting algorithm of this modification FIT-MATCH.

In order to analyze FIT-MATCH, two questions need to be answered, namely to what extent this algorithm is indeed more efficient than MATCH and how the fitting parameter influences the quality of the alignments that are found.

Concerning efficiency, Figure 5 exemplifies that exploring only fitting extensions seems to be an effective pruning strategy; this can be explained by the local diversity property of metabolic networks. In our experiments, we found that whereas MATCH explores a search tree of size around \( 6^k \) on average in order to align a \( k \)-path pattern, a fitting parameter of \( x = 0.5 \) reduces this to around \( 2.5^k \).

Concerning the quality of alignments, we found that setting the fitting parameter to \( x = 0.5 \)—the same setting that gives the majority of gain in efficiency—causes no alignment to be missed that one would see as biologically meaningful.

3 EXPERIMENTS ON METABOLIC NETWORKS

We implemented FIT-MATCH in C++ in order to test its practical performance. The program—which Section 4 discusses in more detail—is available at http://theinf1.informatik.uni-jena.de/metapat/.

3.1 Method and results

Our testing machine is an AMD Athlon 64 3400+ with 2.4 GHz, 512 KB cache and 1 GB main memory running under Debian GNU/Linux 3.1. Sources were compiled with the GNU g++ 4.2 compiler using the option ‘-O3’.

To evaluate the performance of FIT-MATCH, we used the metabolic pathways of five different organisms that are contained in the BioCyc database (Karp et al., 2005), namely 145 pathways of Bacillus subtilis, 220 pathways of E.coli, 190 pathways of Homo sapiens, 176 pathways of Saccharomyces cerevisiae and 267 pathways of Thermus thermophilus. The highly connected metabolites H₂O, ATP and ADP were excluded from all pathways. If the full EC number of an
Table 1. Running times of fit-match for all-against-all alignments between the five datasets described in the text

<table>
<thead>
<tr>
<th>Pattern</th>
<th>B. subtilis</th>
<th>E. coli</th>
<th>H. sapiens</th>
<th>S. cerevisiae</th>
<th>T. thermophilus</th>
</tr>
</thead>
<tbody>
<tr>
<td>B. subtilis</td>
<td>0.25/0.61</td>
<td>0.41/0.86</td>
<td>0.33/0.85</td>
<td>0.32/0.78</td>
<td>0.53/1.08</td>
</tr>
<tr>
<td>E. coli</td>
<td>0.33/0.75</td>
<td>0.57/1.13</td>
<td>0.48/1.09</td>
<td>0.44/0.97</td>
<td>0.70/1.31</td>
</tr>
<tr>
<td>H. sapiens</td>
<td>0.73/1.26</td>
<td>3.47/4.07</td>
<td>6.74/7.42</td>
<td>5.69/6.25</td>
<td>6.61/7.28</td>
</tr>
<tr>
<td>S. cerevisae</td>
<td>0.53/0.98</td>
<td>1.03/1.57</td>
<td>0.89/1.48</td>
<td>0.82/1.33</td>
<td>1.22/1.81</td>
</tr>
<tr>
<td>T. thermophilus</td>
<td>0.86/0.14</td>
<td>3.60/4.22</td>
<td>6.96/7.66</td>
<td>5.84/6.48</td>
<td>6.74/7.45</td>
</tr>
</tbody>
</table>

For each combination of host and pattern sets, we show the total runtime excluding I/O overhead and including I/O overhead. All values are given in seconds, the fitting parameter was 0.5.

enzyme was not specified, the unknown part of the code was treated as ‘do not care’, meaning that the enzyme is scored as if it were identical to every enzyme for which the known part of the codes match. All 25 possible all-against-all inter- and intra-species alignments between the five datasets where performed, resulting in a total of 960,004 homology searches.

Following the suggestion of Pinter et al. (2005) to set the gap score to about one third of the worst enzyme–enzyme similarity score, we set \( g = -4.5 \). The fitting parameter \( x \) was set to 0.50 in accordance with our observations from the previous section concerning efficiency and the quality of results obtained. The obtained running times are shown in Table 1; some sample alignments are shown in Figure 6.

3.2 Discussion

Our experiments show that fit-match is capable of aligning metabolic pathways very quickly: the complete dataset can be aligned in roughly 70 s on our testing machine, including the complete I/O overhead. This is much faster than the pathway alignment tool of Pinter et al. (2005): their implementation—called MetaPathwayHunter—requires some hours alone to align the simplified trees of the E. coli and S. cerevisiae pathways whereas fit-match can align the corresponding unsimplified data in < 4 s.

The alignments shown in Figure 6 exemplify some interesting application scenarios where fit-match can efficiently be used:

- **Pathway comparison.** Figure 6a shows the highlighting of alternative metabolic pathways by comparing the classical TCA cycle with a more complex variant (note how the complex variant uses more pathways and the succinate dehydrogenases 1.3.99.1 instead of 1.3.5.1).
- **Enzyme classification.** In Figure 6b, our results align all unclassified enzymes (denoted ‘-.-.-.-’) with already known enzymes, possibly hinting at their function.
- **Identifying enzyme complexes.** The pathways shown in Figure 6c are almost identical, except that B. subtilis does not possess the enzyme 2.3.1.157 (an acyltransferase) but is rather aligned to a gap. The preceding enzyme is unclassified in B. subtilis, suggesting that it fulfills a task that requires two enzymes in T. thermophilus.
- **Data integration.** Figure 6d shows an example where we can use fit-match to detect the consistency of a database: the two enzyme classification numbers that are seemingly totally different are the result of a change in nomenclature.

The results we found moreover demonstrate that the topological restrictions imposed by the algorithm of Pinter et al. (2005) cause relevant alignments to be missed in several cases. As an example, we mention the alignment of the methylglyoxal pathway and the chorismate super pathway of E. coli: MetaPathwayHunter does not produce any results whereas fit-match finds an alignment. As a second example, MetaPathwayHunter misses the possible alignment between the cobalamin biosynthesis and the KDO2 lipid biosynthesis super pathway of E. coli (which fit-match found).

4 THE METAPAT TOOL

Based on the fit-match algorithm, we have developed a user-friendly tool called Metapat to facilitate the alignment of metabolic pathways. The Metapat tool is written in the C++ programming language and consist of approximately 5000 lines of non-library code. The graphical user interface and other system-dependent features are implemented with the multi-platform wxWidgets framework (Smart et al., 2005).

The Metapat tool is able to process two types of input:

- Flatfiles from the BioCyc database (Karp et al., 2005), from which the metabolic networks are automatically extracted.
- A simple ascii format representation of a metabolic network in which each line represents a reaction and contains the EC number of the catalyzing enzyme.

The mutual similarity scores for enzymes can either be calculated on the fly from the given input networks or be looked up in a static table that was generated for the BioCyc database. In both cases the scoring scheme of Tohsato et al. (2000)—as introduced in Section 2.1—is used to obtain the similarities between enzymes.

After the alignments have been performed, the results are presented graphically. For convenience, the tool offers the possibility to drag and drop the vertices into a suitable shape before exporting the resulting image as a JPEG, PNG or XPM image.
The user interface, which is shown in Figure 7, is structured in accordance with the steps that are necessary to set up the algorithm. The tool offers a variety of useful options such as the ability to name metabolites which shall be excluded from the network (e.g. highly connected metabolites such as ATP) and the possibility to select the metabolic pathways to be aligned (BioCyc flatfiles always contain all pathways of a species).

The result dialog offers a number of filtering functions. For instance, the user can restrict the results to some select pathways of interest. Another filter causes exact matches between pathways to be excluded as these rarely reveal new insights. Finally, METAPAT can also detect alignments which differ only very slightly from higher scoring ones and might thus not be of interest.

5 CONCLUSION

We have presented the concept of local diversity for metabolic networks and shown how this property can be exploited to obtain a simple alignment algorithm FIT-MATCH for metabolic pathways that is both faster and more generally applicable than previous approaches. The algorithm has been made available as a free and user-friendly graphical tool for the discovery and analysis of metabolic pathway alignments.

Since all biological networks carry labels at their vertices, we think that the concept of local diversity is likely to occur in other types of biological networks than metabolic networks and could thus be exposed for alignment algorithms there, too. It is conceivable that the seemingly impractical tree decomposition-based algorithms for subgraph homeomorphism (Dessmark et al., 2000; Hajaghayi and Nishimura, 2002; Matoušek and Thomas, 1992) can be made practically applicable for the alignment of biological networks by using local diversity.

Another interesting open question for future research is that of scoring the statistical significance of network alignments. Pinter et al. (2005) propose to score the significance of a metabolic pathway alignment by generating a large number of random graphs over the vertex set of the host graph, aligning the pattern graph with these and reporting the percentage of random hosts that yield a better-scoring alignment.

Fig. 6. Four examples for the alignments that were found by the FIT-MATCH algorithm (metabolites are not shown due to space constraints). The pattern enzyme is at the top and the host enzyme is at the bottom of the highlighted differing edges. The four alignments that are shown are (a) superpathway of glycolysis, pyruvate dehydrogenase, TCA, and glyoxylate bypass versus the Embden-Meyerhof pathway in *B.subtilis* (b) anaerobic respiration pathways of *E.coli* and *B.subtilis* (c) peptidoglycan and lipid A precursor biosynthesis in *B.subtilis* and *T.thermophilus* (d) super pathways of leucine, valine, and isoleucine biosynthesis in *E.coli* and *T.thermophilus*.
This, however, does not seem to be a very convincing approach because we can expect this percentage to be extremely low. What are alternative approaches to determining alignment significance? An idea here might be to avoid the explicit generation of random host graphs but rather use mathematical insight on the structure of randomly generated host graphs (see Wernicke (2006) for a similar approach in the context of scoring the significance of so-called ‘network motifs’).

ACKNOWLEDGEMENTS

An extended abstract of this work appears in the Proceedings of the 5th Asia-Pacific Bioinformatics Conference (APBC 2007) held in Hong Kong, January 2007. Our work was supported by the Deutsche Telekom Stiftung (Sebastian Wernicke) and the Deutsche Forschungsgemeinschaft (Florian Rasche) project PEAL (parameterized complexity and exact algorithms, NI 369/1). Finally, we are grateful to Rolf Niedermeier (Jena) for discussions and comments on this work.

Conflict of Interest: none declared.

REFERENCES


