Automatic Layout and Visualization of Biochemical Pathway

鄭鈺森　　蔡志忠　　黃國根
國立中正大學資訊工程學系

Abstract

The main contribution of this paper is a new visualization algorithm that combines and improves well-known forced-directed layout approach and hierarchical layout approach, and can handle complex pathways composed of cycles and hierarchies. Experiment shows that our approach can clear render the global component structure of complex pathways as well as the local structure in each component. Based on our algorithm, we developed a biochemical pathway visualization system called VisualPathway, which consists of pathway category browsing , pathway layout, and pathway visualization. For category browsing, we apply fisheye visualization technique to enhance the browsing efficiency so that the user can find desired pathway more conveniently. For pathway visualization, we adopt the Petri Net as pathway representation. Users can access information behind the pathway by clicking components in the visualization.

1. Introduction

Visualization of pathways always pose a challenge to bioresearchers. Though the idea is simple, drawing of complex pathways is not easy. In the past, pathways were drawn on papers. Recently, several pathway editors have been developed. Such editors take advantages of input devices and graphical interfaces. Constructing a drawing by mouse and keyboard are easier than by hand and pen. Furthermore, graphics components are developed to match pathway components.

However, pathway data is frequently updated. That makes maintaining and drawing of pathway data tedious and time-consuming, even we use editors. A solution to the problem is to develop an editor that can automatically layout and visualize pathways.

One of the main challenges of automatically drawing pathway is the layout problem. We can transform the relation between pathway components into a directed graph, and the drawing problem then becomes a general graph layout problem. Layout is a transformation from topology to geometry; that is, it generates coordinates for components. There are two well-known layout approaches: hierarchical layout [13,18] and forced directed layout[20]. However, each of them can only handle simple pathway structures such as hierarchies and cycles that are common components of biochemical pathways. In this paper, we develop a new layout algorithm HOLY (Hierarchically Organized LaYout) for hierarchically organized complex pathways. In HOLY, a complex pathway is decompose into basic components such as cycles and hierarchies that are layout, using conventional hierarchical layout and force-directed layout algorithms. HOLY then determines a joining order of basic components, and joins them one at a time to form the final layout. Experiment shows that our approach can preserve local as well as global structures of complex pathways, and renders better visualization for several pathways studied in previous related work[32,34].

2. System Overview

Our pathway visualization roadmap involves several function units illustrated in Figure 2.1. In general there are three significant parts: the data source, the category browser, and the pathway explorer. The data source means the management of pathway database. It loads the pathway data into memory, updates the content of database, and allows other software components to access the data through protocols. For increasing the practicability of our system, users could use a pathway data editor to modify the data. Furthermore, the data source could invoke a data importer transferring the data from other formats, like the MetaCyc[32] database or the KEGG XML, to our system.

The second part of our visualization is pathway category browser, which is aimed to improve the browsing efficiency of traditional way. We integrate a visualization technique, called hyperbolic tree[5], and pathway category information into an interactive browsing platform. The user find out where the interested pathway in the database belongs to and what comments researchers added by interacting with the platform iteratively. When the user found the desired pathway, he/she can additionally call our

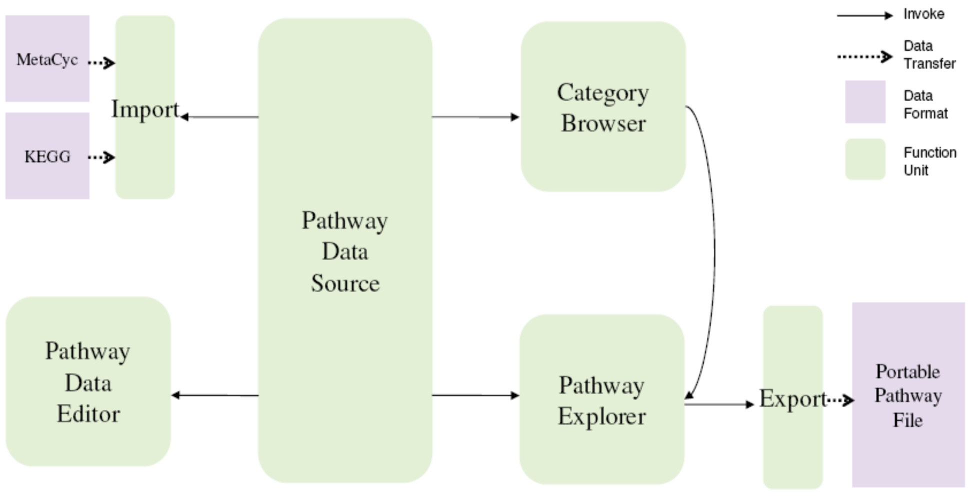
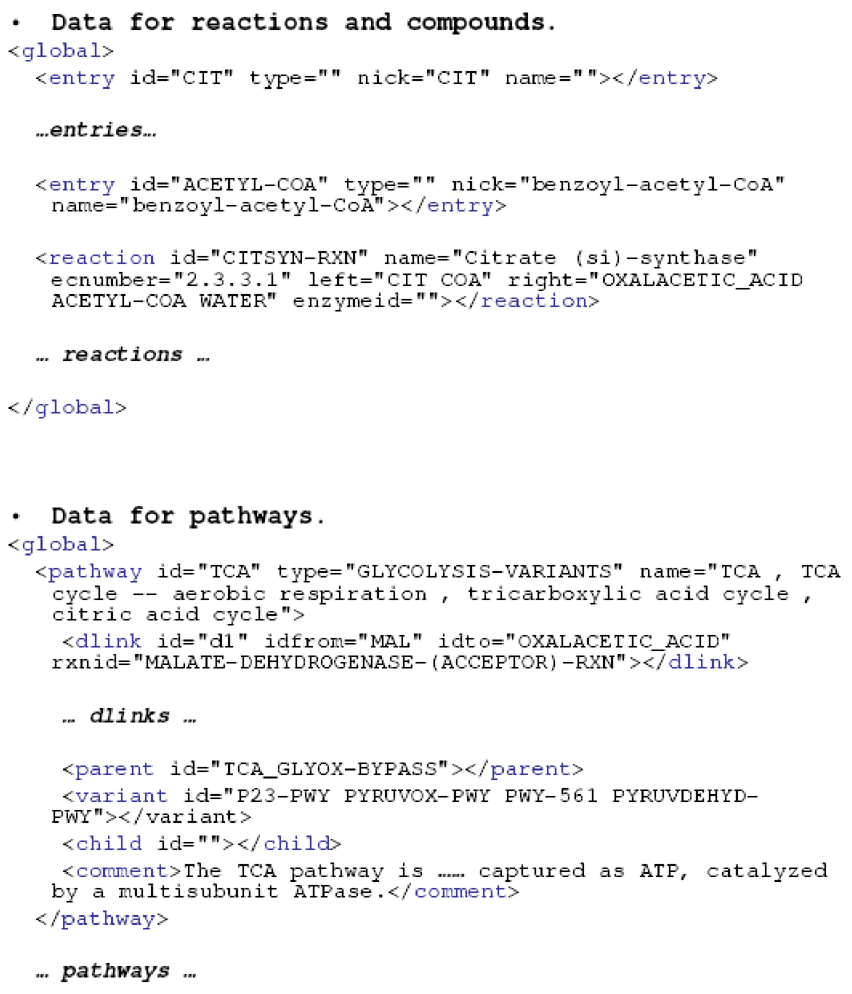


Figure 2.1: An overview of our visualization system. The symbol annotation is on the top-right corner.



```
•  Data for reactions and compounds.
<global>
  <entry id="CIT" type="" nick="CIT" name=""></entry>

  …entries…

  <entry id="ACETYL-COA" type="" nick="benzoyl-acetyl-CoA"
   name="benzoyl-acetyl-CoA"></entry>

  <reaction id="CITSYN-RXN" name="Citrate (si)-synthase"
   ecnumber="2.3.3.1" left="CIT COA" right="OXALACETIC_ACID
   ACETYL-COA WATER" enzymeid=""></reaction>

  … reactions …

</global>


•  Data for pathways.
<global>
  <pathway id="TCA" type="GLYCOLYSIS-VARIANTS" name="TCA , TCA
   cycle -- aerobic respiration , tricarboxylic acid cycle ,
   citric acid cycle">
    <dlink id="d1" idfrom="MAL" idto="OXALACETIC_ACID"
   rxnid="MALATE-DEHYDROGENASE-(ACCEPTOR)-RXN"></dlink>

    … dlinks …

    <parent id="TCA_GLYOX-BYPASS"></parent>
    <variant id="P23-PWY PYRUVOX-PWY PWY-561 PYRUVDEHYD-
   PWY"></variant>
    <child id=""></child>
    <comment>The TCA pathway is …… captured as ATP, catalyzed
   by a multisubunit ATPase.</comment>
  </pathway>

  … pathways …
```

Figure 2.2: The XML storage used in our pathway system.

pathway explore to exploit the detail information of it.

The last important part is pathway explorer for visualizing pathways. Most known service of visualizing pathway try to dump all information of a pathway on screen in graphical way. However, heavy display makes people confused. Usually people like to know partial information of a pathway. We thereby design an interactive interface helping people exploit preferred pathway knowledge. For providing convenience to other bio-software, the explorer can also export a portable pathway file that contains all information of a single pathway, like KEGG XML. The graph layout algorithm acts as core techniques in the visualization.

Pathway data of the system is stored in two XML files illustrated in Figure 2.2. The top side of figure is the one containing reactions and entries, in which the tag global involves several reaction records and entry records. All attributes of each record indicate its content. The bottom side of the figure shows the XML file describing pathways where the tag dlink represents a reaction with primer substrate and product only. It is designed for constructing digraphs when performing layout. Tags parent, child, and variant are used to describe super-pathway the sub-pathway and the variance of a pathway.when it is hierarchically organized.

4. Pathway Visualization

The mission of pathway exploration is to convey the knowledge of pathway in the database. A proper interface interactively provides user preferred information on screen, while it hides inessential data. The difficulty of pathway visualization is to draw its structure nicely. We translate the pathway description data into directed graph so that we know where to draw pathway components according to the information of graph layout.

Petri Net representation of pathway [2] is used in our system. The Petri Net contains two types of node and directed arcs connecting nodes of different types. The two types of nodes are called place nodes and transition nodes. The place node denoted as circles means compound; the transition node denoted as boxes indicates reaction. As in Figure 4.1, reaction nodes are labeled by their EC numbers.

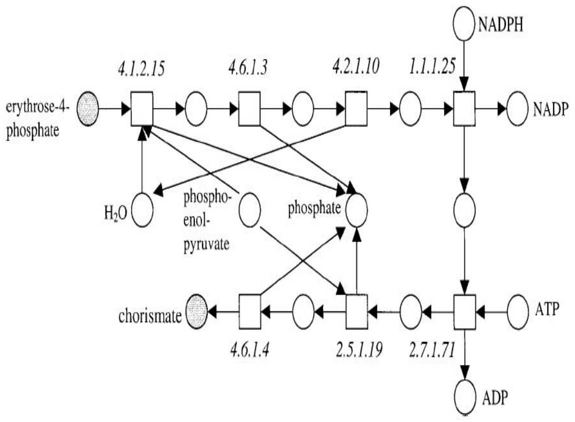
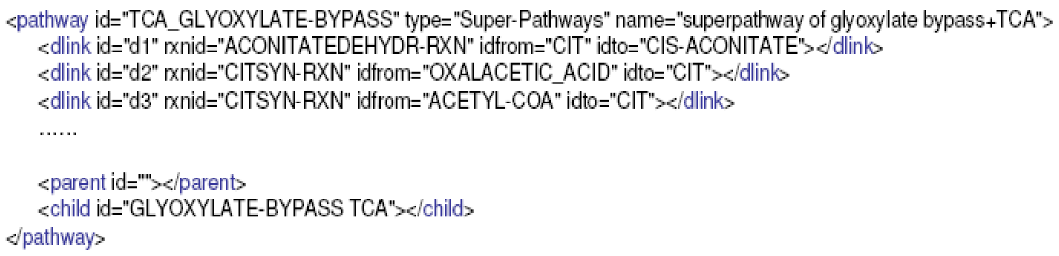


Figure 4.1: An example of Petri Net representation from [2].

The Petri Net becomes directed graph if we treat the two types of nodes as the same. They usually form hierarchies. The pathways may also contain cycles that are important features in biology. Known hierarchical layout approaches are capable to draw pathways. However, most hierarchical layout approaches will distorts the shape of cycle. For the capability of drawing various pathway structures graphically, we design a layout approach to organize both the hierarchies and cycles in a pathway. Also we integrate several known layout algorithms in our system.



```
<pathway id="TCA_GLYOXYLATE-BYPASS" type="Super-Pathways" name="superpathway of glyoxylate bypass+TCA">
    <dlink id="d1" rxnid="ACONITATEDEHYDR-RXN" idfrom="CIT" idto="CIS-ACONITATE"></dlink>
    <dlink id="d2" rxnid="CITSYN-RXN" idfrom="OXALACETIC_ACID" idto="CIT"></dlink>
    <dlink id="d3" rxnid="CITSYN-RXN" idfrom="ACETYL-COA" idto="CIT"></dlink>
    ……

    <parent id=""></parent>
    <child id="GLYOXYLATE-BYPASS TCA"></child>
</pathway>
```

Figure 4.2: A pathway record in our pathway database file

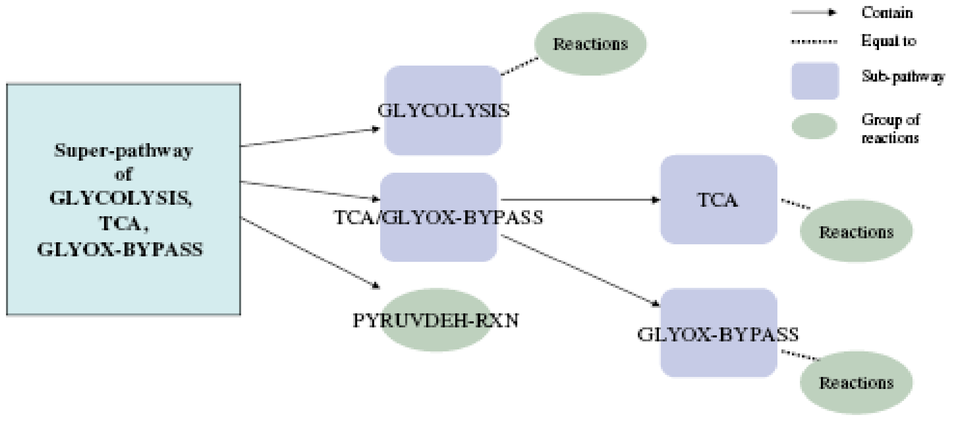


Figure 4.3: Pathway composition of Glycolysis/TCA/Glyox-Bypass.

```
function layoutWithDcmpTree(pathwaynode pw)
{
  Foreach sub-pathway of pw do
      layoutWithDcmpTree (sub-pathway);
  End

  /* Only reactions in the pw.*/
  If no more sub-pathways
  {
      If pw contains cycle
        call Force_Directed_Layout(pw);
      Else
        call Sugiyama_Hierarchical_Layout(pw);
      End

      Store the layout result;
      Return;
  }
}
```

Figure 4.4: Procedure of laying pathway out with a pathway decomposition tree.

4.1 Hierarchically Organized Layout

The forced-directed layout[20] produces nice pathway layout with cycles, while the hierarchical layout[18] performs good at the one without cycles. We develop a new layout algorithm HOLY (Hierarchically Organized LaYout) for hierarchically organized complex pathways. In HOLY, a complex pathways is decompose into basic components such as cycles and hierarchies that are layout, using conventional hierarchical layout and force-directed layout algorithms. HOLY then determines a joining order of basic components, and joins them one at a time to form the final layout. HOLY aims to explore local as well as global structures of complex pathways, and rendes better visualization for several pathways studied in previous related work[32,34].

HOLY consists of the following four nontrivial stages.

1. Parsing Stage: parse the pathway data.
2. Decomposition Stage: decompose the super-pathway into hierarchical groups of

pathways.

3. Layout Stage: layout each group with proper basic approaches.
4. Joining Stage: join group layouts to a large final layout.

In parsing stage, HOLY parses our pathway database which is stored in XML format. As in Figure 4.2, the element *pathway* containing attribute *id="TCA GLYOXYLATE-BYPASS"* represents a real pathway "TCA GLYOXYLATE-BYPASS". Note that the element *pathway* also contains a sub-element called *child*. Its attribute *id="GLYOXYLATE-BYPASS TCA"* shows this pathway has two sub-pathways. They are *GLYOXYLATEBYPASS* and *TCA*. With that information, we can find out the sub-pathways by looking for the element pathway containing attribute *id="GLYOXYLATE- BYPASS"* or *id="TCA"*. Using the child information we know how to decompose a super-pathway into sub-pathways in step two.

In decomposition stage, the decomposition recursively divides a super-pathway into sub-pathways until the current decomposition will break a cycle, or the current sub-pathways have high overlapping with each other. Cycles in biochemical pathway are important feature so we preserve the cycles. Overlapping between sub-pathways means that they are highly related and they should not be decomposed. We set 30 percent of nodes as a threshold of overlapping in our system. Figure 4.3 is an example of pathway decomposition. The superpathway Glycolysis/TCA/Glyox-Bypass has been decomposed to three subpathways: Glycolysis, TCA/Glyox-Bypass, and Pyruvdeh-rxn. The Pyruvdeh-rxn forms a group itself in this case Note that the TCA/Glyox-Bypass could be decomposed to TCA and Glyox-Bypass. However, the TCA covers most part of Glyox-Bypass, so we treat them as one.

In Layout Stage, HOLY performs layout on each sub-pathway in the decomposition tree. The procedure is given in Figure 4.4. The recursive function traverses the decomposition tree and lays out each sub-pathway with a proper basic approach. For the sub-pathway containing cycles, we call the forced directed layout approach to emphasize the symmetry of graph and uniform edge lengths; for the sub-pathway containing hierarchies, we perform the hierarchical layout approach to stress on the direction of arcs.

However, an issue should be considered before joining sub-pathways. The nodes connecting two sub-pathways should be placed in the outside of the layout; otherwise it could cause a large number of edge crossings after joining. For instance, both part (a) and part (b) in

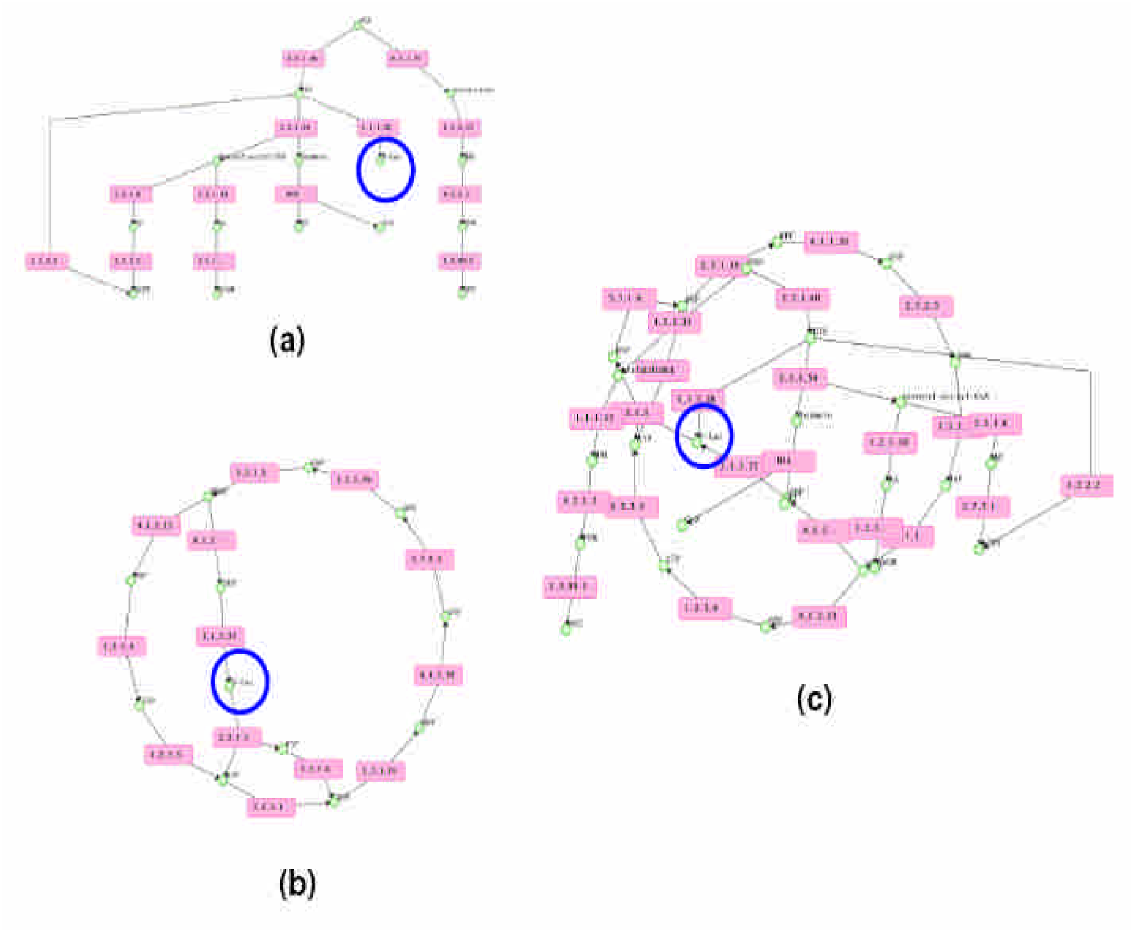


Figure 4.5: Joining two sub-pathway layouts. Part (a) is a hierarchies and part (b) contains cycle. Their connecting node marked by blue circle are both inside the layout. Part (c) shows the result of joining (a) and (b).

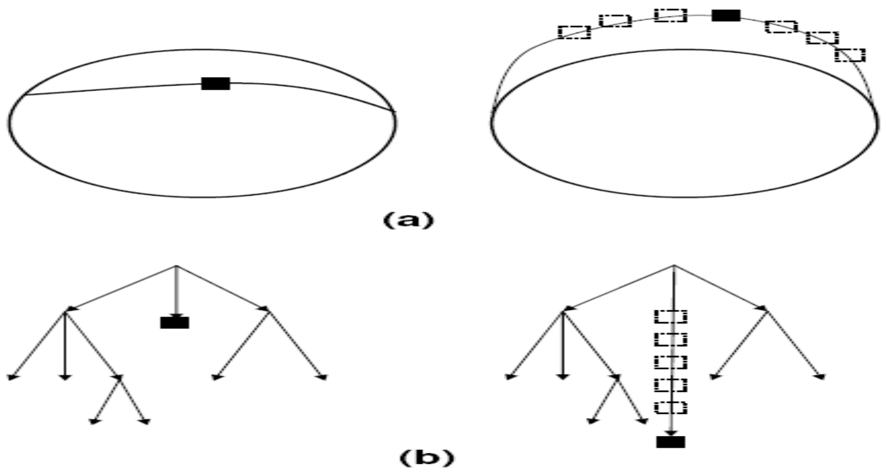


Figure 4.6 : Avoiding the inner connecting node. (a) inserting pseudo-nodes for cycle case; part (b), for hierarchies. Square filled with solid black represents the connecting node and squares with dotted lines are pseudo-nodes.

Figure 4.5 are well visualized. However, their joining result in the part (c) contains a large number of edge crossings. We avoid this problem by inserting pseudo-nodes into sub-pathway to push connecting nodes outside boundary of the layout. Our node insertion has two policies: one for cycles and the other for hierarchies. As illustrated in Figure 4.6 Part (a) left, we insert pseudo-nodes into both sides of a connecting node in a cycle. The inner path becomes longer than the outer path. Using forced-directed layout, the inner path will be flipped to the outside, like Part (a) right. For hierarchies, we simply insert pseudo-nodes between the connecting node and its parent node. The connecting node will then be pushed to the bottom of the layout, like Part (b) in Figure 4.6. Figure 4.7 gives new layout of the pathway in Figure 4.5. As connecting nodes are pushed outside, we obtain a better joining result with much less crossings.

In the final step, we join layouts of sub-pathways. We decide one sub-pathway in the decomposition tree as the first main component, and join other pathways one at a time. Figure 4.8 illustrates how to join sub-pathways. In step (1) we calculate the mass of each sub-pathway which is marked as a red crossing in Figure 4.8. In step (2), we shift and attach the new sub-pathway to main component. Note that attaching points are nodes where pathways split during decomposition. At step 3, we fix the attaching point, and rotate the new component to maximize the distance between the positions of two masses.

To further join another sub-pathway, we rotate the third pathway and maximize the summation of distance A and B as indicated in Figure 4.8. The idea is the last mass should leave all other masses as far as possible, especially the large one. Therefore, we maximize the following function to join the *k*th sub-pathway. Let ComponentSize(*i*) denotes the number of nodes in component *i*.

$$\sum_{i=1}^{k-1} \mathrm{dist}(\mathrm{Mass}_k, \mathrm{Mass}_i) * \mathrm{ComponentSize(i)}$$

We also design a heuristic to determine the joining order to speed up the joining process, and make joining results more suitable to biochemical pathways. If there is a sub-pathway containing cycles, usually other sub-pathways are hierarchies and are inserted into the cycle. This implies that cycles should have higher priority when choosing joining candidates. In addition, we give higher priority to large sub-pathways as small sub-pathways are often attached to large sub-pathways. The joining priority of a sub-pathway is the defined as the number of nodes in the sub-pathway, and is doubled if the sub-pathway contains cycles. The sorted order of priorities is the joining order.

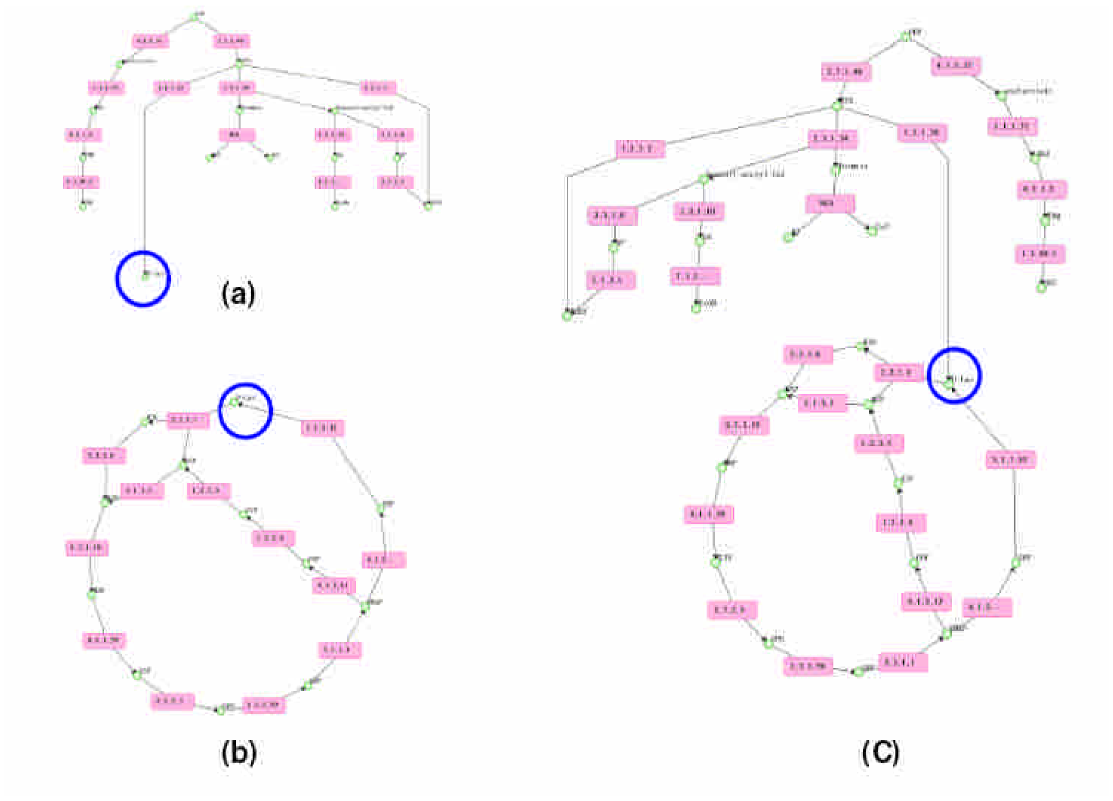


Figure 4.7: Same example with Figure 4.17, but avoid the inner connecting node by inserting pseudo-nodes.

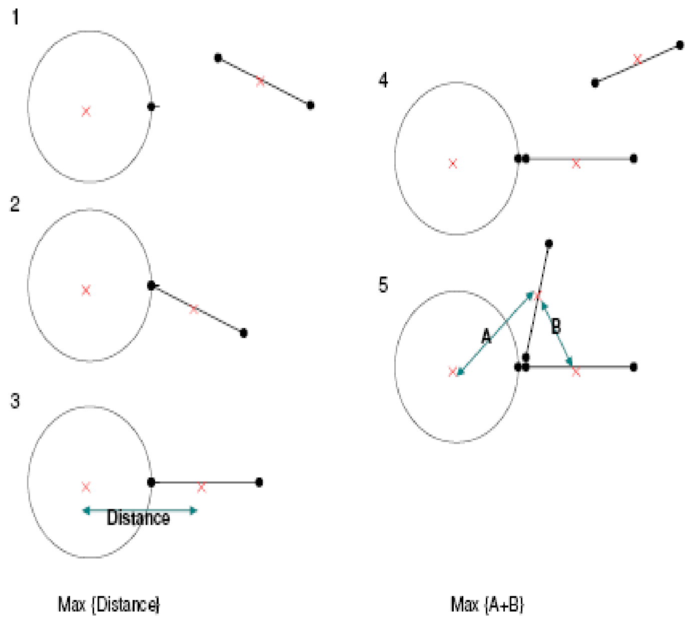


Figure 4.8: An example of combining three components into one large component from step 1 to step 5.

improves a lot in statistics of running times. That is, our approach is specified for biochemical pathway.

5. Experimental Results

We compare HOLY to the approaches in MetaCyc[32], a patway database that provides automatic layout and visualization of pathways, and CADLIVE[34], a pathway editing and visualization tool. MetaCyc contains 658 pathways including 60 super-pathways. Most of them are hierarchies except 20 cycles and one complex pathway. We have tested our approach on their pathway data. All types of the pathway in MetaCyc are well visualized. We present and discuss our experimental results for the three types of pathways: Hierarchies, cycles, and complex structures.

5.1 Hierarchies

The pathway *Mixed Acid Fermentation* in Figure 5.1 is a hierarchy and does not contain any sub-pathway. As shown in the left part (a), MetaCyc produces a layout that contains a bottom-up arc that we may not want it to happen. HOLY gives a visualization in which all arcs are drawn toward the bottom as shown in (b).

Figure 5.2 gives results for pathway C1 compounds oxidation to CO2. The pathway is composed of seven sub-pathways marked by different colors and numbers in Figure 5.2(c). All of them are hierarchies. Both HOLY and MetaCyc produce similar result in which sub-pathways are easily distinguished. However, our system usually performs better for the case that components are not visually organized as illustrated in the cases for cycles.

5.2 Cycles

Figure 5.3 gives results for pathway TCA Cycle. The pathway is a simple cycle with one alternated reaction bypassing the inside of the cycle.

Both HOLY and MetaCyc produce acceptable results. The TCA Cycle was drawn in a shape of sphere and the bypassing arc was symmetrically placed along the circle.

Figure 5.4 gives results of a more complex pathway which contains a cycle and long and multiple bypassing arcs inside the cycle. In our result, Figure 5.4(b), the Calvin Cycle has two bypasses in geometry. One is composed of two reactions that their EC-Numbers are 5.3.1.6 and 2.2.1.1; one is composed of three reactions with EC-Number 4.2.1.¡, 3.1.3.37,

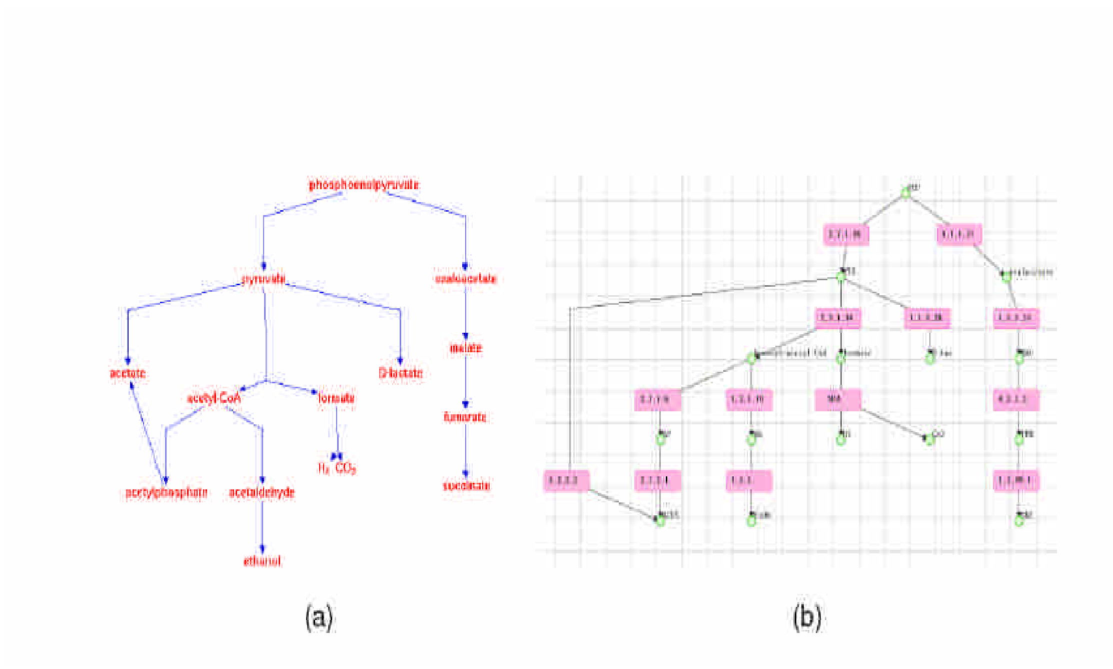


Figure 5.1: The pathway layout of *Mixed Acid Fermentation* from (a)MetaCyc and (b)VisualPathway.

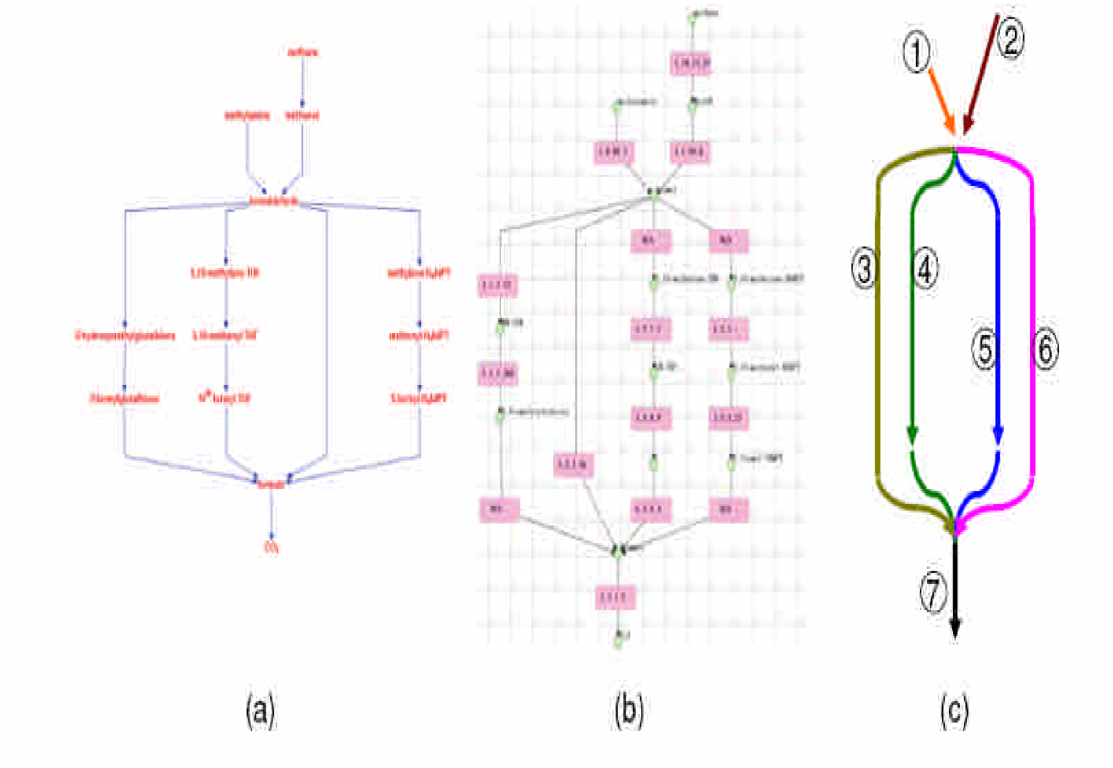


Figure 5.2: The pathway layout from (a)MetaCyc and (b)VisualPathway. Part (c) sketches seven sub-components of the super-pathway. Each color represents an individual sub-pathway of Super-pathway of C1 compounds oxidation to CO2.

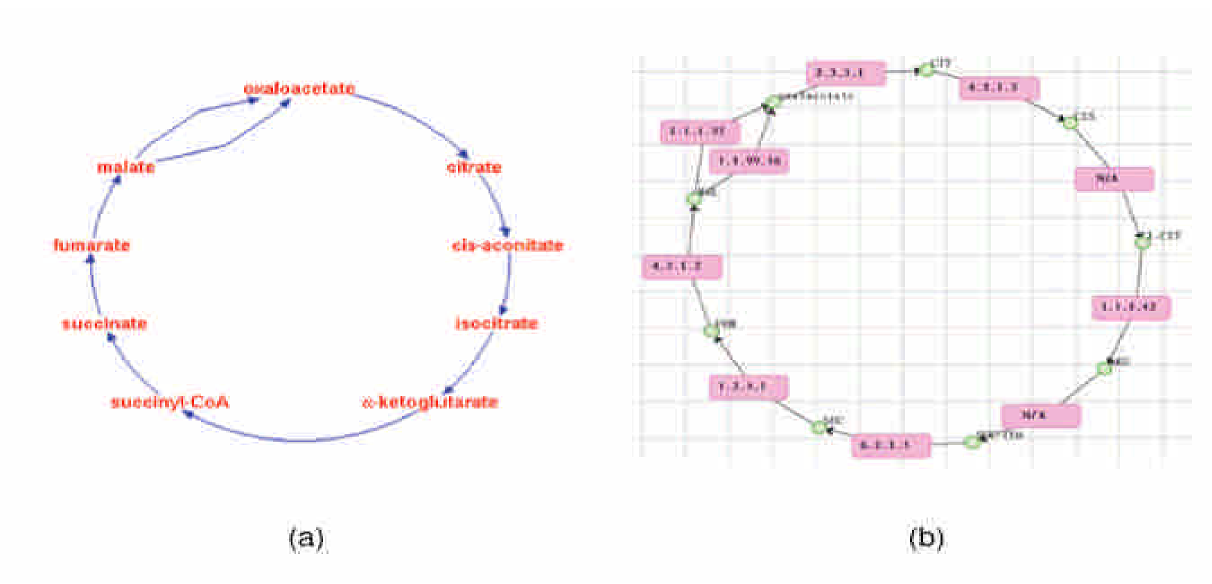


Figure 5.3: The pathway layout of TCA Cycle from (a)MetaCyc and (b)VisualPathway.

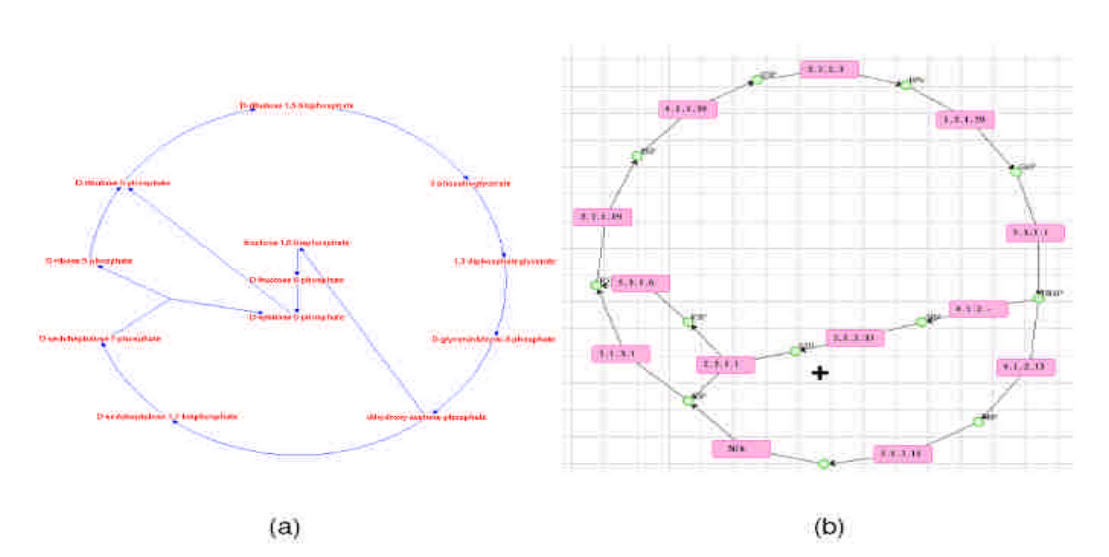


Figure 5.4: The pathway layout of *Calvin Cycle* from (a)MetaCyc and (b)VisualPathway.

and 2.2.1.1. The short bypass, just like the TCA, is along the circle, while the longer goes straight across the inside. As in Figure 5.4 (a) MetaCyc draws bypasses with zigzag and distorted paths. However, HOLY clearly displays the geometry of Calvin Cycle. In this case, HOLY generates more comprehensible layout.

5.3 Complex Structures

Complex pathways

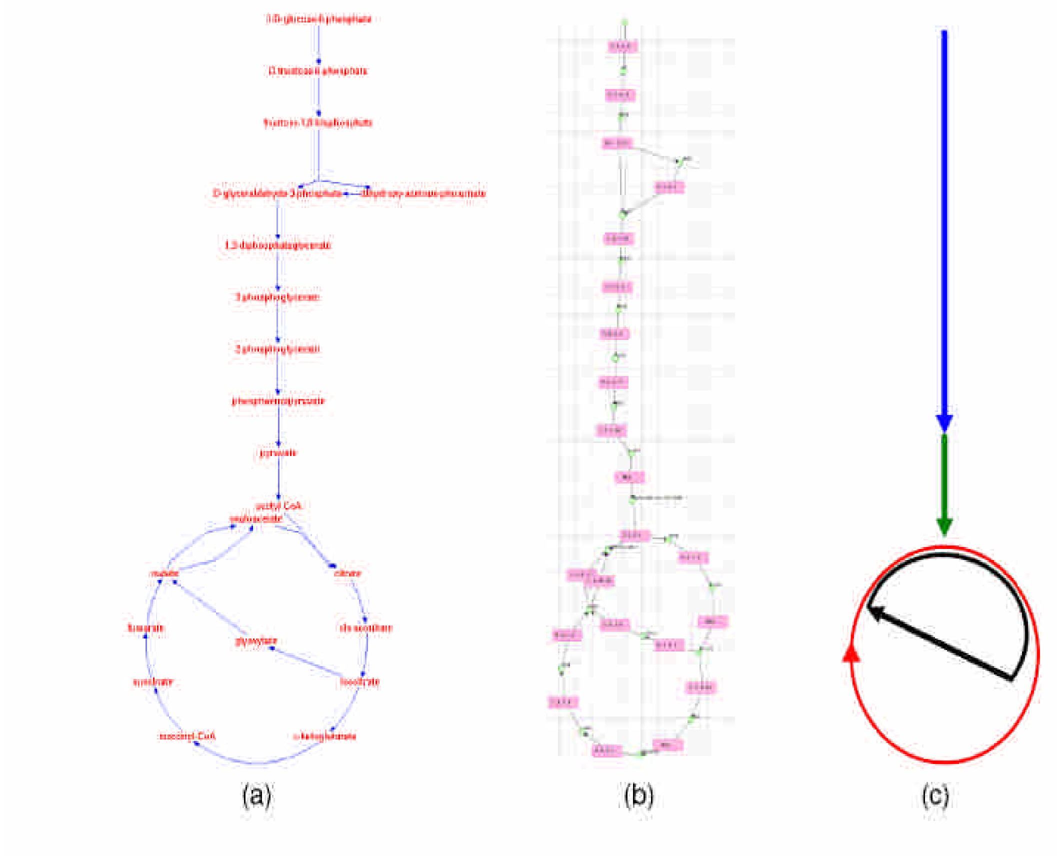


Figure 5.5: The layout from (a)MetaCyc and (b)VisualPathway. Part (c) sketches four sub-components of the super-pathway. The blue arrow represents the hierarchy of *glycolysis*; the green one, the reaction *pyruvdeh* itself; the red, the *TCA cycle*; finally the black cycle is the *glyoxylate bypass*.

contain both hierarchies and cycles, and have complex hierarchical organizations in which a component can have multiple parents.

Figure 5.5 is the comparison of the layout of super-pathway of glycolysis, pyruvate dehydrogenase, TCA, and glyoxylate bypass. Note that the black cycle and the red cycle as shown in (c) are partially overlapped. The two cycles correspond to two different sub-pathways that should be distinguished in the visualization. However, as in (a), MetaCyc draws only a circle. HOLY, as shown in part (b), gives two connecting bubbles that clearly distinguish both components and preserves the shape of each cycle. This example shows the significant feature of HOLY that emphasizes both the local structures and the component structures of complex pathways in the visualization.

We also compare HOLY to another pathway tool, CADLIVE. We take the pathway given in [34] that is composed of pentose-phosphate, glycolysis, and TCA cycle. Figure 5.6 gives the result. Both HOLY and CADLIVE successfully produce clustered pathway layout in which each cluster corresponds to an individual sub-pathway. However, HOLY produces much better layout inside each cluster. In the layout from CASLIVE, reactions (arcs) are entangled with each other, and the structure of each cluster is difficult to read. In the layout from HOLY,

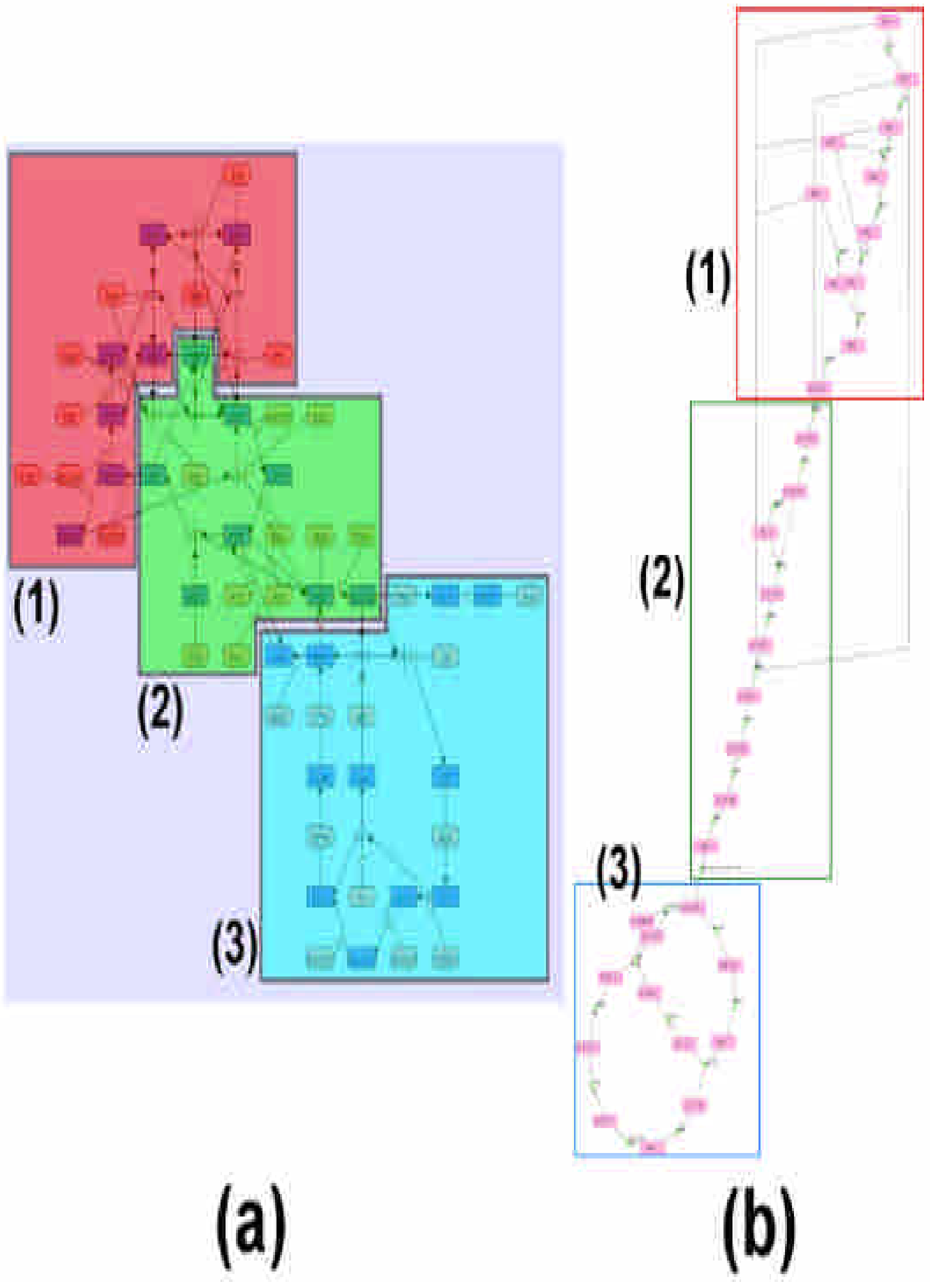


Figure 5.6: The pathway layout of pentose-phosphate, glycolysis, and TCA cycle from (a)CADLIVE and (b)VisualPathway.

The cluster in red area is drawn in forms of multiple hierarchies with some long feedback arcs, which reveals the real inner structure of that sub-pathway. The main structure of the cluster in green area is a long path, and the cluster in the blue structure consists of two overlapped cycles.

Above experiment shows that HOLY peforms better than MetaCyc and CADLIVE, and is able to gives visualization of complex pathways that clearly renders the global component structures as well as the local structure of each component. In addition, with priority assignment to determine joining order, HOLY, runs very fast, usually accomplishing layout of a complex pathway in seconds. That means the overall performance of HOLY is good enough for use in interactive applications.

6. Conclusions

In this paper, we propose an biochemical pathway visualization system called VisualPathway, which consist of pathway category browsing , pathway layout, and pathway visualization. For category browsing, we apply fisheye visualization technique to enhance the browsing efficiency so that the user can find desired pathway in short time. For pathway layout, we combine well-known forced-directed layout approach and hierarchical layout approach, and propose a new hierarchically organized layout algorithm. Our approach is able to handle complex pathways composed of both cycles and hierarchies. Experiment shows that our approach can clearly render local as well as global structures of complex pathways. For the pathway visualization, we adopt the Petri Net as pathway representation. By using the layout information and the graphical representation, we develop an interactive pathway visualization system in which the pathway is automatically produced and drawn immediately, and users can access information behind the pathway by clicking components in the visualization.

REFERENCES

[1] **R. M. Karp, Reducibility among combinatorial problems Complexity of Computer Computations, pp. 85-103, 1972.**

[2] **J.W. Pinney, D.R. Westhead, and G.A. McConkey, Petri Net representations in systems biology. Biochemical Society Transactions, pp. 1513-1515, 2003**.

[3] **J. Bingham and S. Sudarsanam, Visualizing Large Hierarchical Clusters in Hyperbolic Space. Bioinformatics, vol. 16, no. 7, pp. 660-661, 2000.**

[4] **J. Lamping and R. Rao, Visualizing Large Trees Using the Hyperbolic Browser. Proceedings of ACM Conference on Human Factors in Computing Systems, pp. 388-389, 1996.**

[5] **J. Lamping, R. Rao, and P. Pirolli, A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. Proceedings of ACM Conference on Human Factors in Computing Systems, pp. 401-408,1995.**

[6] **G. Robertson, J. Mackinlay, and S. Card, Cone Trees: Animated 3D Visualizations of Hierarchical Information. Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 189-194, 1991.**

[7] **G. G. Robertson, K. Cameron, M. Czerwinski, and D. Robbins, Polyarchy Visualization: Visualizing Multiple Intersecting Hierarchies. Proceedings of the Conference on Human Factors in Computing Systems, pp. 423-430,2002.**

[8] **E. Kleiberg, H. van de Wetering, and J. J. van Wijk, Botanical Visualization of Huge Hierarchies. IEEE Symposiumon Information Visualization, pp. 87-94, 2001**

[9] F. van Ham and J. J. van Wijk, Beamtrees: Compact Visualization of Large Hierarchies. Information Visualization, pp.93-100, 2002.

[10] B. Shneiderman, Tree Visualization with Tree-Maps: 2-D Space-Filling Approach. ACM Transactions on Graphics, vol. 11, pp. 92-99, 1992.

[11] T. Masui. Evolutionary Learning of Graph Layout Constraints from Examples. Proceedings of the ACM Symposium on User Interface Software and Technology, pp. 103-108, 1994.

[12] A. M.S. Barreto, H. J.C. Barbosa, Graph Layout Using a Genetic Algorithm. VI Brazilian Symposium on Neural Networks, pp. 179, 2000.

[13] P. Eades and K. Sugiyama. How to draw a directed graph. Journal of Information Processing, pp. 424-437, 1990.

[14] L. Carmel, D. Harel and Y. Koren, Combining Hierarchy and Energy for Drawing Directed Graphs. IEEE Transactions on Visualization and Computer Graphics, vol. 10, no. 1, pp. 46-57, 2004.

[15] C. Friedrich, F. Schreiber, Flexible Layering in Hierarchical Drawings with Nodes of Arbitrary Size. Proceedings of the 27th Conference on Australasian Computer Science, vol. 26, pp. 369-376, 2004.

[16] E.B. Messinger, L.A. Rowe, and R.H. Henry, A Divide-and-Conquer Algorithm for the Automatic Layout of Large Directed Graphs. IEEE Transaction on Systems, Man, and Cybernetics, vol. SMC-21, no. 1, pp. 1-12,1991.

[17] K. Sugiyama, Cognitive Approach for Graph Drawing. Cybernetics and Systems: An International Journal, vol. 18, pp. 447-488, 1987.72

[18] K. Sugiyama, S. Tagawa, and M. Toda, Methods for Visual Understanding of Hierarchical Systems. IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-11, no. 2, pp. 109-125, 1981.

[19] T. Kamada and S. Kawai, An algorithm for drawing general undirected graphs. Information Processing Letters, pp. 7-15, 1989.

[20] E.R.Gansner, S.C. North, Improved Force-directed Layouts. Proceedings of the 6th International Symposium on Graph Drawing, London, UK, 1998.

[21] Y. Koren and D. Harel, Axis-by-Axis Stress Minimization. Proceedings of Graph Drawing, 2003.

[22] T. Fruchterman and E. Reingold, Graph Drawing by Force-Directed Placement. Software-Practice and Experience, vol. 21, no. 11, pp. 1129-1164, 1991.

[23] W. Salamonsen, K. Mok, P. Kolatkar, and S. Subbiah, BioJAKE: A Tool for the Creation, Visualization and Manipulation of Metabolic Pathways. Pacific Symposium on Biocomputing, pp. 392-400, 1999.

[24] K. Sugiyama, S. Tagawa, and M. Toda, Methods for Visual Understanding of Hierarchical System Structures. IEEE Transactions on Systems, Man and Cybernetics, pp. 109-125, 1981.

[25] F.J. Brandenburg, M. Forster, A. Pick, M. Raitner, and F. Schreiber, Biopath: Exploration and Visualization of Biochemical

Pathways. Mathematics and Visualization, pp. 215-236, 2003.

[26] E. Trost, H. Hackl, M. Maurer, and Z. Trajanoski, Java Editor for Biological Pathways. Bioinformatics, pp. 786-787, 2003.

[27] P. D. Karp and S. M. Paley, Automated Drawing of Metabolic Pathways. Proceedings of the Third International Conference on Bioinformatics and Genome Research, pp. 225-238, 1995.

[28] L.B.M. Ellis, C.D. Hershberger, and L.P. Wackett. The University of Minnesota Biocatalysis/Biodegradation Database: Microorganisms, Genomics, and Prediction. Nucleic Acids Research, pp. 377-379, 2000.

[29] M. Kanehisa and S. Goto, KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Research, pp. 27-30, 2000.

[30] The ExPASy (Expert Protein Analysis System) proteomics server, http://www.expasy.ch/ .

[31]Metabolic Pathway Editor, http://mpe.empproject.com/ .

[32]MetaCyc Encyclopedia of Metabolic Pathways, http://metacyc.org/ .

[33] Eclipse.org, http://www.eclipse.org/ .

[34]CADLIVE,http://kurata21.bse.kyutech.ac.jp/cadlive .

[35] W. Li and H. Kurata, A Grid Layout Algorithm for Automatic Drawing of Biochemical Networks. Bioinformatics, vol. 21, no. 9, pp. 2036-2042, 2005.

[36] MathWorld, http://mathworld.wolfram.com/ .

[37] S. Skhiri dit Gabouje and E. Zim˜anyi, A New Compound Graph Layout Algorithm for Visualizing Biochemical Networks Poster Proceedings Volume of the 4th International Workshop on Efficient and Experimental Algorithms, WEA 05, 2005.