

# Customizable Visualization in the Context of Metabolic Networks

Von der Naturwissenschaftlich-Technischen Fakultät  
der Universität Siegen

zur Erlangung des akademischen Grades

**Doktor der Naturwissenschaften**  
**(Dr. rer. nat.)**

genehmigte Dissertation

von

**Peter Droste**

- 1. Gutachter:** Prof. Dr. Wolfgang Wiechert, Forschungszentrum Jülich
- 2. Gutachter:** Prof. Dr. Andreas Kolb, Universität Siegen
- Vorsitzender:** Prof. Dr. Volker Blanz, Universität Siegen

Tag der mündlichen Prüfung: 02.12.2011



# **Customizable Visualization in the Context of Metabolic Networks**

Peter Droste

March 21, 2011

**Permanent Address:**

Peter Droste  
Uferstraße 27  
57368 Lennestadt  
Germany

To my wife



# Deutschsprachige Kurzfassung

Die Systembiologie hat das ehrgeizige Ziel, die biologischen Zusammenhänge in lebenden Organismen ganzheitlich zu verstehen. Um dieses Ziel zu erreichen, werden experimentelle Studien gekoppelt mit Computersimulation in einem iterativen Prozess eingesetzt. Biologische Prozesse werden gemeinhin als Netzwerke aufgefasst, welche die unterschiedlichen Stoffe in einer Zelle und die Stoffübergänge beschreiben. Solche sogenannten *metabolischen Netzwerke* sind Gegenstand vieler Studien zum Verständnis des zellulären Stoffwechsels.

Moderne Hochdurchsatz-Techniken und Hochleistungs-Computersimulation führen heutzutage zu Unmengen von Daten, welche diese Stoffwechsel-Netzwerke betreffen. Die Auswertung dieser Daten stellt eine große Herausforderung für die heutige Forschung dar. Visualisierung ist hier ein bedeutendes Hilfsmittel. Diese Doktorarbeit befasst sich mit der Visualisierung von Daten in metabolischen Netzwerken. Dabei werden neuartige Ansätze in vier verschiedenen Themengebieten entwickelt:

## **Zeichnen von Netzwerkdiagrammen**

Es wird ein Zeichenwerkzeug für Stoffwechsel-Diagramme entwickelt, welches eine hohe Anwenderfreundlichkeit aufweist und die speziellen Anforderungen metabolischer Netzwerke unterstützt.

## **Netzwerklayout**

Im Zusammenhang mit metabolischen Netzwerken sind *de facto* Standards für das Anordnen der Netzwerkkomponenten entstanden, die von gängigen Netzwerklayout-Algorithmen nicht unterstützt werden. Es werden semi-automatische Layouttechniken entwickelt, welche das Zeichnen von Netzwerken nach konventionellen Standards unterstützen.

## **Datenvisualisierung**

Zur Visualisierung von Daten in Netzwerkdiagrammen wird eine neuartige Herangehensweise vorgestellt: die skript-basierte Visualisierung. Es wird eine Skriptsprache eingeführt, mit der Anwender die Datenvisualisierung nach eigenen Anforderungen programmieren können.

## **Erweiterbarkeit durch Plugins**

Damit ein Visualisierungswerkzeug an zukünftige Anforderungen anpassbar bleibt ist eine Plugin-Schnittstelle unumgänglich. Der Entwurf dieser Schnittstelle und bestehende Modellierungs- und 3D-Visualisierungs-Plugins werden vorgestellt.

Eine hohe Anzahl an Beispielanwendungen in der Arbeit macht den großen Nutzen der hier entwickelten neuen Ansätze für die Systembiologie deutlich.





# Preface

*“Of making many books there is no end, and much study wearies the body.  
Now all has been heard; here is the conclusion of the matter:  
Fear God and keep his commandments, for this is the whole duty of man.”*

Ecclesiastes 12:12b-13, New International Version

With these biblical words I want to begin my dissertation because, in my honest opinion, science and faith belong together. Science is founded upon human curiosity, upon human quest for understanding. Above all, faith reminds us that everything has a deeper sense that is far beyond human ability to grasp. Hence, belief in God brings modesty and gratitude. Not the ego comes first but the privilege to participate in something that is bigger than oneself and bigger than one can recognize.

In five years working on my doctoral graduation I was more than ever confronted with the situation that one cannot learn everything, that the whole is much more enormous than one alone can ever know in detail. It was and will be a pleasure to contribute my expertise to a science that aims at understanding the complexity of life.

Finally, faith reminds us about our ethic base that is an indispensable prerequisite for a responsible research. In the context of the current debate about intellectual property and plagiarism in Germany’s society it is advisable to remember:

“You shall not steal.”

“You shall not give false testimony against your neighbour.”

Now all has been heard; here is the conclusion of the matter: It is the duty of any scientist to follow these ethical principles.

My doctoral work started in the Simulation Group at the Institute of Systems Engineering at the University of Siegen in 2006. The research project was financially supported by several funding agencies and industrial partners which I have gratefully acknowledged on page 14. I would like to thank the colleagues of the Simulation Group, especially Prof. Dr. Roland Reichardt and Dr. Marc Kalkuhl. I also want to thank Prof. Dr. Andreas Kolb from the Computer Graphics group at the University of Siegen for all valuable scientific discussions in the colloquia and correspondence.

Since 2009 the work was continued in the Modeling & Simulation Group at the Institute of Bio- and Geosciences; IBG-1 of the research center Jülich. I want to thank all members of the “ModSim” group for cooperativeness and collegiality, in particular Dr. Michael

## Preface

---

Weitzel, Remo Winz, Tobias Vehrkamp, Andreas Dietrich, Sebastian Niedenführ, Elisabeth Zelle, Tolga Dalman and our group leader Dr. Eric von Lieres.

All members of the institute deserve my thanks that supported my work with valuable scientific discussions and by employing the tool that I have developed. My special acknowledgment have Dr. Stephan Noack and Stephan Miebach. Additionally, I want to thank all users of my visualization tool that gave encouraging comments and interesting suggestions.

I am furthermore very grateful to Dr. Katharina Nöh for good advisory and for all encouragements. My particular gratitude goes to my supervisor Prof. Dr. Wolfgang Wiechert for encouraging me to work on my doctorate, for guiding me into the research field of the life sciences and for all support during the years. Thank you very much.

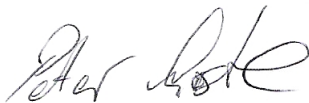
Acknowledgements to family and friends are given in my mother tongue:

Ich möchte mich bei all meinen Freunden bedanken, dass sie in der Zeit des Schreibens meiner Doktorarbeit zu mir gestanden haben, auch wenn ich nicht viel Zeit hatte. Ein besonderer Dank gilt meinem besten Freund Michael Bender.

Ich danke meinen Eltern Hermann und Beatrix für die liebevolle Unterstützung während meiner Schulzeit, meines Studiums und meiner Promotion. Ohne euch würde ich heute nicht da stehen, wo ich stehe.

Meine größte Dankbarkeit gilt meiner lieben Frau Petra, der ich diese Arbeit widme. Du hast all die Jahre zu mir gestanden und mich hingebungsvoll unterstützt, auch wenn du oft auf mich verzichten musstest. Du bist unseren Kindern eine großartige Mutter, was mich sehr stolz macht. Du hast mir immer den Rücken gestärkt. Ohne dich wäre diese Arbeit nicht möglich gewesen. Dafür danke ich dir von Herzen.

Lennestadt, March 21, 2011



Peter Droste

## Statement of Authorship

I certify that this dissertation is my original work and I properly referenced all sources. Except where reference is made in the text, this document contains no material extracted in whole or in part from a document previously published by me. This thesis was not previously submitted to another academic institution.

## Software Resources

This document was prepared using the (pdf)L<sup>A</sup>T<sub>E</sub>X<sup>1</sup> typesetting system and *Kile*<sup>2</sup>. The figures in this document were prepared using the software systems *Omix*<sup>3</sup>, *GIMP*<sup>4</sup>, *Inkscape*<sup>5</sup>, *Microsoft PowerPoint*<sup>6</sup>, *CytoScape*<sup>7</sup>, *OpenOffice.org*<sup>8</sup> and *yEd*<sup>9</sup>.

---

<sup>1</sup>see <http://www.latex-project.org>

<sup>2</sup>see <http://kile.sourceforge.net>

<sup>3</sup>see <http://www.13cflux.net/omix>

<sup>4</sup>see <http://www.gimp.org>

<sup>5</sup>see <http://www.inkscape.org>

<sup>6</sup>see <http://office.microsoft.com/en-us/powerpoint>

<sup>7</sup>see <http://www.cytoscape.org>

<sup>8</sup>see <http://www.openoffice.org>

<sup>9</sup>see <http://www.yworks.com>



# List of Symbols and Abbreviations

## Symbols

$E$	Set of edges of a graph
$E(v)$	Set of edges connected to vertex $v$
$F$	Set of flux edges of a metabolic network
$F'$	Set of cofactor edges, $F' \subseteq F$
$G$	Graph
$G_P$	Pathway sub graph
$M$	Set of metabolites
$P$	Single pathway
$R$	Set of reactions
$V$	Set of vertexes of a graph
$X$	Set of effector edges of a metabolic network
$f$	Single flux edge $f \in F$
$f'$	Single cofactor edge $f' \in F'$
$v$	Single vertex $v \in V$
$x$	Single effector edge $x \in X$
$ E(v) $	Connectivity of vertex $v$
$\Pi_r$	Periphery of a reaction $r$

## Abbreviations

API	application programming interface
CC	connected component
CSV	Comma/Character Separated Values
DNA	deoxyribonucleic acid
EC	Enzyme Commission number

## List of Symbols and Abbreviations

---

GC	gas chromatography
GUI	graphical user interface
IDE	integrated development environment
ILE	isotope labeling experiment
ILN	isotope labeling network
JNI	Java Native Interface
KEGG	Kyoto Encyclopedia of Genes and Genomes
LC	liquid chromatography
MDI	multiple document interface
MFA	metabolic flux analysis
mRNA	messenger RNA
MS	mass spectrometry
NMR	nuclear magnetic resonance spectroscopy
OVL	Omix Visualization Language
PDB	Protein Data Bank
RNA	ribonucleic acid
SBGN	Systems Biology Graphical Notation
SBML	Systems Biology Markup Language
SCC	strongly connected component
SVG	Scalable Vector Graphics
TCA	citric acid cycle
VoD	Visualization on Demand
XML	Extensible Markup Language

# Glossary

***in silico***

(Latin: within the silicon) performed on computer.

***in vitro***

(Latin: within the glass) performed on whole or on parts of organisms in a controlled environment.

***in vivo***

(Latin: within the living) performed on whole, living organisms.

**Comma/Character Separated Values**

Textual spreadsheet file format.

**Enzyme Commission number**

Standard for enzyme classification.

**Excel**

Proprietary table calculation software by Microsoft Corporation.

**Extensible Markup Language**

Textual all-purpose file format.

**FML**

File format for models of the *13C FLUX 2* simulation framework based upon XML.

**FWDSim**

File format for simulation results of the *13C FLUX 2* simulation framework.

**Java Native Interface**

Allows to call platform dependent native code from inside Java code and vice versa.

**MathML**

XML based data type for the representation of mathematical formulas.

**Matlab**

Name of a proprietary simulation environment of *MathWorks* and likewise of its scripting language (derived from matrix laboratory).

## **Glossary**

---

### **Modelica**

Equation-based programming language for the modeling of complex systems.

### **OpenOffice.org**

Open source office software providing a text processing, slide show, spreadsheet, drawing and formula tool.

### **Scalable Vector Graphics**

Vectorial graphics file format based upon XML.



# Contents

<b>Deutschsprachige Kurzfassung</b>	<b>vii</b>
<b>Preface</b>	<b>ix</b>
<b>List of Symbols and Abbreviations</b>	<b>xiii</b>
Symbols . . . . .	xiii
Abbreviations . . . . .	xiii
<b>Glossary</b>	<b>xv</b>
<b>1. Introduction</b>	<b>1</b>
1.1. The Complexity of Life . . . . .	1
1.2. Systems Biology . . . . .	2
1.3. Benefit of the Holistic Approach . . . . .	4
1.4. “Ome” and “Omics” . . . . .	4
1.5. Biological Processes represented by Networks . . . . .	6
1.6. Multi-Omics Data . . . . .	7
1.7. Visualization of Data . . . . .	9
1.8. Existing Visualization Tools . . . . .	10
1.9. Restrictions of Existing Tools . . . . .	12
1.10. Research Context . . . . .	14
1.11. Contribution of this Thesis . . . . .	14
1.12. Organization of this Thesis . . . . .	16
<b>I. Network Drawing</b>	<b>19</b>
<b>2. Introduction to Network Drawing</b>	<b>21</b>
2.1. State-of-the-Art in Network Drawing . . . . .	21
2.2. Tool Survey . . . . .	22
2.3. Proposed Standards . . . . .	23
2.4. Requirements . . . . .	23
<b>3. The Network Diagram Editor</b>	<b>25</b>
3.1. Ease of Use . . . . .	25

## Contents

---

3.2. Network Symbols . . . . .	27
3.2.1. Network Nodes and Edges . . . . .	28
3.2.2. Metabolic Pathways . . . . .	32
3.2.3. Compartments . . . . .	32
3.2.4. Additional Network Components . . . . .	33
3.3. Network Style . . . . .	33
3.4. Levels of Detail . . . . .	34
3.5. Joining Networks . . . . .	35
3.6. Layout-less Components . . . . .	37
3.7. Example Drawing Process . . . . .	38
<b>Feature Sheet</b>	<b>41</b>
<b>II. Network Layout</b>	<b>43</b>
<b>4. Introduction to Network Layout</b>	<b>45</b>
4.1. Layout Conventions . . . . .	45
4.2. Automatic Versus Manual Approaches . . . . .	45
4.3. Circular Arrangements . . . . .	49
4.4. Layout Requirements of Metabolic Networks . . . . .	50
<b>5. The Layout Pattern: Shaping the Coarse Network Structure</b>	<b>51</b>
5.1. Using the Layout Pattern . . . . .	52
5.2. Challenges and Requirements . . . . .	53
5.3. Semi-automatic Path Search . . . . .	54
5.3.1. Human-Interactive Path Search . . . . .	54
5.3.2. Software-Assisted Pre-Selection and Exclusion of Network Paths . . . . .	55
5.3.3. Pathway-Oriented Layout . . . . .	56
<b>6. Motif Stamps: Cloning Small Reaction Patterns</b>	<b>57</b>
6.1. Creating Motif Stamps . . . . .	58
6.2. Appending Motifs to Reactions . . . . .	59
<b>7. Exemplary Work Flow</b>	<b>63</b>
7.1. Metabolic Pathways . . . . .	63
7.2. Using the Layout Pattern . . . . .	63
7.3. Using Motif Stamps . . . . .	64
7.4. Completing the Drawing Process . . . . .	64

<b>III. Visualization</b>	<b>65</b>
<b>8. Introduction to Customizable Visualization</b>	<b>67</b>
8.1. State-of-the-Art in Multi-Omics Data Visualization . . . . .	67
8.2. Network-Integrated Visualization of Data . . . . .	68
8.3. Basic Concept of the Visualization Approach . . . . .	69
<b>9. Script-based Visualization</b>	<b>71</b>
9.1. The Omix Visualization Language . . . . .	72
9.1.1. Extending Network Components . . . . .	72
9.1.2. Simplified Access . . . . .	73
9.1.3. Data Types in OVL . . . . .	74
9.1.4. Interactive Components . . . . .	74
9.1.5. Additional Information Carriers . . . . .	78
9.1.6. Accessory Arrays . . . . .	79
9.1.7. Meta-Information . . . . .	79
9.2. OVL Development . . . . .	80
9.3. Visualization on Demand . . . . .	81
9.3.1. Interactive Visualization of Custom Properties . . . . .	82
9.3.2. Custom Data Types . . . . .	84
9.3.3. Information Management . . . . .	85
9.3.4. VoD Example . . . . .	85
9.4. Advantages of OVL . . . . .	85
<b>10. Application Examples</b>	<b>89</b>
10.1. Comparative Visualization of Metabolome Data . . . . .	89
10.2. Visualization in the Context of Metabolic Flux Analysis . . . . .	91
10.3. Visualization in the Context of Synthetic Biology . . . . .	93
10.4. Visualization of Transcriptome Data . . . . .	95
10.5. Animation of a Pulse Experiment . . . . .	96
10.6. Visualization of Isotopic Mass Distributions . . . . .	99
10.7. Visualization of Enzyme Kinetics . . . . .	100
<b><i>Feature Sheet</i></b>	<b>103</b>
<b>IV. Extensibility</b>	<b>105</b>
<b>11. Introduction to Extensibility</b>	<b>107</b>
11.1. Motivation . . . . .	107
11.2. Differentiation to OVL . . . . .	108

## Contents

---

<b>12. The Omix API</b>	<b>109</b>
12.1. Interface Concept . . . . .	109
12.2. Features of the Omix API . . . . .	110
<b>13. Existing Plug-In Solutions</b>	<b>113</b>
13.1. Compatibility and External Resources . . . . .	113
13.1.1. Model Describing Formats . . . . .	113
13.1.2. Database Connectivity . . . . .	113
13.2. Visualization of Chemical Structures . . . . .	114
13.3. Metabolic Flux Analysis Work Flow . . . . .	115
13.3.1. Background . . . . .	115
13.3.2. 13C FLUX 2 Modeling Plug-In . . . . .	116
13.3.3. 13C FLUX 2 Launcher Plug-In . . . . .	117
13.3.4. FWDSim Import Plug-In . . . . .	117
13.3.5. Network Analysis Plug-Ins . . . . .	117
13.4. 3D Visualization of Isotope Labeling Networks . . . . .	118
13.4.1. Foundations . . . . .	118
13.4.2. The CumoVis Plug-In . . . . .	119
13.5. 3D Visualization of Network Thermodynamics . . . . .	122
13.5.1. Foundations . . . . .	122
13.5.2. The ThermoVis Plug-in . . . . .	122
13.6. Further Features . . . . .	124
<b>Closing</b>	<b>125</b>
<b>14. Summary and Discussion</b>	<b>127</b>
14.1. Aims . . . . .	127
14.2. Results . . . . .	128
14.2.1. Novel Solutions . . . . .	128
14.2.2. Extraordinary Achievement . . . . .	129
14.2.3. Realized Vision . . . . .	130
14.3. Future Perspectives . . . . .	132
14.3.1. Other Network Layers . . . . .	132
14.3.2. Network Layout . . . . .	132
14.3.3. Metabolic Flux Analysis . . . . .	132
14.3.4. 3D Visualization . . . . .	133
<b>Appendix</b>	<b>134</b>
<b>A. Drawing Tool Survey</b>	<b>137</b>
A.1. Office Software . . . . .	137

A.2. Graphics Software . . . . .	138
A.3. Drawing Software for Biochemical Networks . . . . .	139
A.3.1. CytoScape . . . . .	140
A.3.2. Vanted . . . . .	142
A.3.3. CellDesigner . . . . .	143
<b>B. Evaluation of the Semi-Automatic Layout Approach</b>	<b>145</b>
B.1. Case Study Design . . . . .	145
B.2. Elaboration . . . . .	145
B.3. Results . . . . .	146
B.4. Conclusion . . . . .	147
<b>Original Publications</b>	<b>149</b>
<b>Documentations</b>	<b>149</b>
<b>Related Publications</b>	<b>149</b>
<b>Further Publications</b>	<b>150</b>
<b>Bibliography</b>	<b>151</b>



# Chapter 1.

## Introduction

### 1.1. The Complexity of Life

The notion of what is life is not an easy task. What makes the difference between non-living and living matter? Respiration and reproduction? Fire, for instance, respire and reproduces itself but it does not live. On the other hand, viruses do neither breathe nor be able to reproduce themselves but we might consider them as to be living. Koshland (2002) defines life by seven essential principles:

- **Program** – The genes encoding the organization of a living organism.
- **Improvisation** – The ability to change the program as response to a changing environment (evolution).
- **Compartmentalization** – The property of living organisms to be spatial separated from its environment and, in case of higher organisms, to be subdivided into multiple subunits (organelles, cells, organs).
- **Energy** – Due to a constant loss of energy by moving and entropy, living matter requires energy to exist. Almost all life on earth gets the energy from the sun.
- **Regeneration** – Living organisms are able to compensate losses and degradation in various mechanisms including the import of molecules from the environment, the synthesis of new components and the complete system's restart by creating a new generation (cell division, reproduction).
- **Adaptability** – The ability of living things to temporarily adapt to requirements, danger and environmental changes as an inherent part of the program.
- **Seclusion** – Refers to the specificity of molecules and enzymes and the subdivision of the living system into different functional pathways.

Life is complex and its complexity can be observed in different layers. Some examples: The coexistence of multiple different species of animals, plants and microbes in an ecological system composes a complex mesh of population control leading to a relative equilibrium. The human body is composed of highly specialized subcomponents building

a complex functioning system. Every single cell is a complex “engine” consisting of thousands of mechanisms to collect energy, to adapt to environmental changes, to regenerate and to reproduce. And last but not least, every single enzyme in this cell is a complex macro molecular structure consisting of one or several specifically folded polypeptide chains leading to their particular catalytic function.

Biology is the science dealing with the understanding of life in all its diversity and complexity. This includes the structure of living organisms, their function, growth, origin, evolution and distribution. In the context of this work, the living cell, its composition and function is of main interest.

### 1.2. Systems Biology

In order to understand a complex system like the cell it does not suffice to know the list of ingredients and their specific properties and function. Kitano (2002) compares this situation with understanding the technical complexity of an airplane from a catalog of its sole components which is simply impossible. The whole must come into focus because it is greater than the sum of its parts (Saraiya et al. 2005). The interrelations between all single components and their contribution to the system need to be identified in order to gain a holistic understanding of the living cell.

A system theoretic view for biological issues has already been postulated at mid-twentieth century as figured out by Wolkenhauer (2001). Since a rapid technological advance in molecular biology research occurred in the later decades – at least expressed by the publication of the complete human genome (Venter et al. 2001) – a paradigm shift has taken place in biology from the reduced observation of single components changing to a system-oriented thinking. In these years, *systems biology* occurred as a new scientific discipline aiming at the understanding of cellular processes by applying a system-level approach. Wolkenhauer (2001) defines: “Genomics is the field of biological research taking us from the DNA sequence of a gene to a structure of the product for which it codes (usually a protein) to the activity of that protein and its function within a cell and, ultimately, the organism. [...] Systems theory is [...] the study of organisation and behaviour *per se* and a natural conclusion is therefore to consider systems biology as the application of systems theory to genomics.”

Today, the research in systems biology is promoted in many countries in Europe, Asia and North America. For instance the German *Federal Ministry of Education and Research* (BMBF) funds systems biology research projects by about 25 million € per year according to the funding catalog data base published by the Government of the Federal Republic of Germany (2011). Scientific symposia like the *International Conference on Systems Biology* (ICSB) have evolved in recent years and the availability of multiple high-ranking journals like *BMC Systems Biology* or *Molecular Systems Biology* demonstrate that systems biology today is well-established.

The investigations of systems biology include the analysis of the system’s structure and organization as well as its dynamics (behavior over time, control mechanisms). Systems



biology combines *in vitro* and *in vivo* experimental approaches with mathematical modeling and *in silico* simulation. Increasing knowledge about the intracellular processes is achieved by a closed loop cycle of different research steps as depicted in Fig. 1.1. The cycle begins with a particular biological question. Upon that question experimental studies are made leading to an amount of measurement data. The next step is to analyze the experimental data and to formulate mathematical models about what has happened in the biological system during the experiments. Thereafter, computer simulation is applied to ensure the significance of the model. The mathematical model is successively validated in iterated simulation steps where the model and model parameters are fitted until simulated results correspond to the real experimental measurements. The valid mathematical model again implies a new question. Is the model suitable under altered conditions? Here, a new circle run starts by performing new experiments, modeling, simulation, data processing and so on.

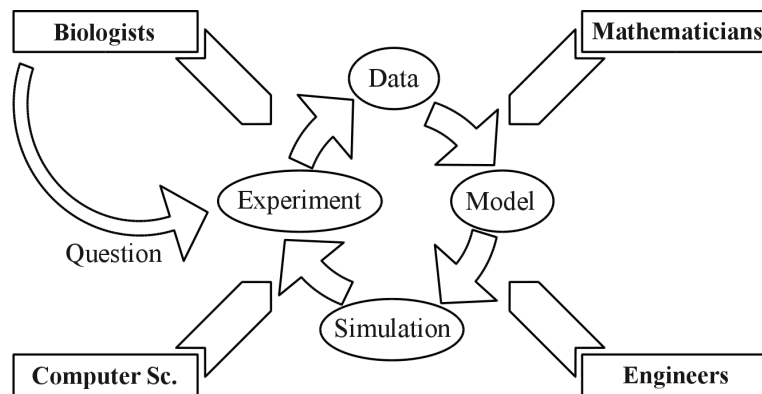


Figure 1.1.: Iterative process of increasing knowledge about biological systems in an interdisciplinary research field.

There are basically two different approaches in modeling of biological systems. The first one is called *top-down*. In top-down modeling the complete system with all known properties and processes is included without knowing the details. In order to fill these gaps, many experiments are made and the model is successively improved upon the experimentally obtained insights. The second approach, called *bottom-up*, starts with small models of parts of the system which are successively enlarged upon the experimentally obtained insights until a holistic systems description is achieved.

Systems biology is highly interdisciplinary (cf. Fig. 1.1): Biology asks the general questions about the organization and control of living systems. It provides decades of experiences in analyzing the life on a microscopic level. Biology and biochemistry contribute their knowledge about the single components inside the cell as well as the techniques to investigate them. Engineering provides experience in systems modeling as well as high-throughput and nano-technologies for experiment and data processing. Mathematics brings the ability to handle complex computational issues like differential equations and

optimization. Computer science and bioinformatics contribute high-performance computation, data management, data analysis and the necessary software infrastructure. Experts from these different disciplines closely work together in systems biology research projects. Because every research group has its own disciplinary terminology, way of thinking and view of the whole, interdisciplinary communication becomes an important issue. Visualization is the key to solve this issue. In many situations, especially in the discourse about models and data as well as for the presentation of results visualization can significantly benefit communication processes.

### 1.3. Benefit of the Holistic Approach

Systems biology with its holistic approach changes the traditional biology from a descriptive to a predictive science able to make realistic assertions about the dynamics of a living organism. The understanding of living organisms on a systems level is valuable in many application fields. Kitano (2002) argues: “The most feasible application of systems biology research is to create a detailed model of cell regulation, focused on particular signal-transduction cascades and molecules to provide system-level insights into mechanism-based drug discovery [. . .]. Such models may help to identify feedback mechanisms that offset the effects of drugs and predict systemic side effects.” A medical vision, for instance, is to build computational models of the intracellular processes in a patient’s individual tumor and, basing on this, to simulate the impact of certain pharmaceuticals in order to find the most effective treatment (Reiß 2002).

Another promising application field for systems biology approaches is biotechnology being a discipline that deals with the usage of cells or cellular processes in technical applications. A classical application are the fermentation processes used since millennia to produce wine and beer. Here, microorganisms produce alcohol from carbohydrates as a natural behavior under anaerobe conditions. In modern biotechnology microorganisms are functionalized for the production of various chemical substances like fine chemicals, bulk chemicals, food and feed additives, pharmaceuticals and biofuels. Therefore, the metabolism of production organisms is specifically manipulated. A key discipline in this context is *metabolic engineering* which is concerned with the purposeful alteration of an organism’s genes to achieve a specific characteristics (Stephanopoulos 1994, Stephanopoulos et al. 1998, Vemuri and Aristidou 2005). Metabolic engineering aims at the manipulation of intracellular reaction pathways and even at the design of new pathways in order to decrease the cellular growth in favor of the over-production of desired substances. Here, an indispensable precondition for metabolic engineering is a holistic understanding of the complex interrelations in the metabolism maintained by systems biology.

### 1.4. “Ome” and “Omics”

In biology one speaks of *genotype* in order to refer the genetic program life bases on given in the DNA and of *phenotype* as the observable physical characteristics of an organism.

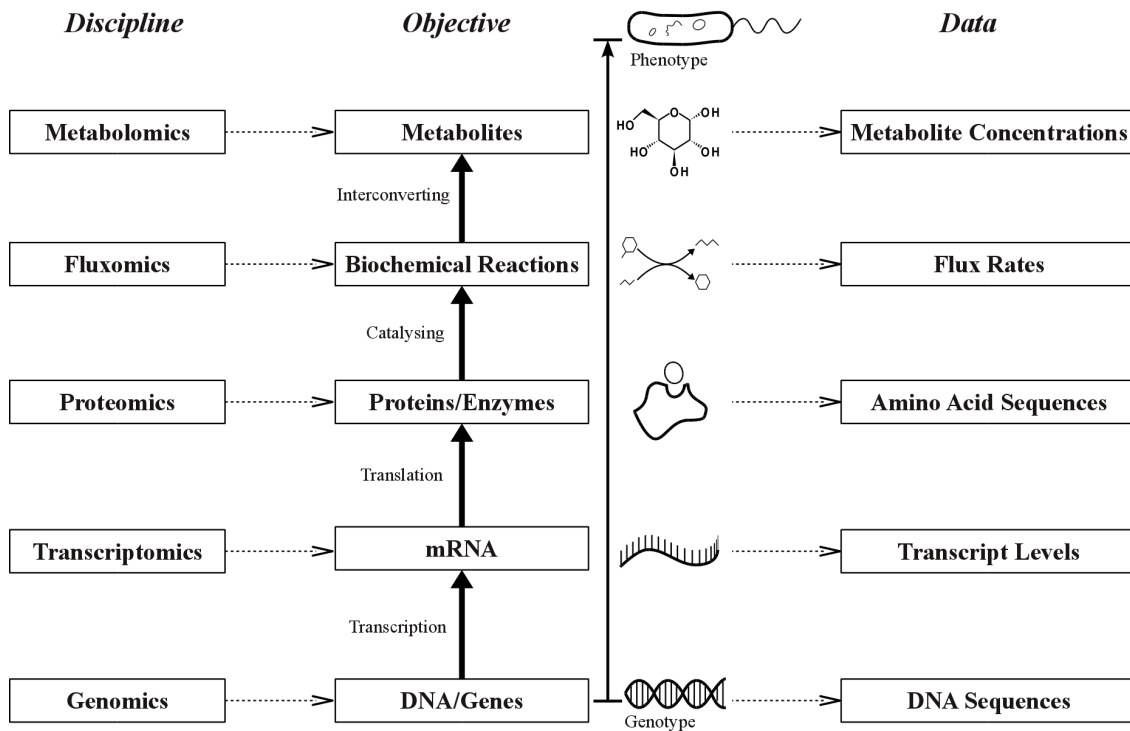


Figure 1.2.: Different levels of intracellular processes between genotype and phenotype and their relation to omics disciplines and data. The figure was inspired by (Vemuri and Aristidou 2005).

The intracellular process generating a specific phenotype upon a genotype is called *gene expression*. Fig. 1.2 shows a schematic overview of different steps of the gene expression. Another use of the term gene expressions only refers the process of generating proteins from a genetic code.

A gene encodes a specific protein in the cell. When a protein is to be created, the corresponding gene is transcribed into messenger RNA (ribonucleic acid). This mRNA is translated into an amino acid sequence, a so-called polypeptide. Proteins consist of one or multiple specifically folded polypeptides. An important type of proteins are the enzymes catalyzing biochemical reactions in the metabolism. In a biochemical reaction, small molecules so-called metabolites are converted into each other. By this the organism can convert nutrients into energy and cell components (biomass).

The investigations of the intracellular processes can be subdivided into different disciplines regarding the levels gene expression takes place as illustrated in Fig. 1.2.

- Genomics is the study of the genome being the entire set of genes of an organism.
- Transcriptomics is concerned with the transcriptome, meaning the transcription process leading from gene to proteins.

- Proteomics aims at understanding the complex interactions between proteins (proteome, interactome).
- Fluxomics deals with the identification and quantification of all intracellular fluxes, i.e. biochemical reactions (Sauer 2004).
- Metabolomics is the research field profiling the range of all metabolites which is consequently referred as metabolome (Khan and Ather 2006).

The neologism *omics* (Lederberg and Mccray 2001) has arisen as a term standing for the totality of all levels to study biological systems. Many other -ome and -omics combinations have been created in the life sciences in recent years like regulomics, bibliomics, ribosomics, metallomics etc. (cf. <http://omics.org>) which are not necessarily sensible. The word nonsensomics hits the bull's eye. Nevertheless, the -ome and -omics terms are useful to refer a certain field of study and classify experimental data. Especially in combination with data, often used terms are *poly-omics* and *multi-omics* representing the diversity of available data concerning a biological system. As a certain extend, systems biology can be regarded as the integration of different omics approaches by modeling and computer simulation.

### 1.5. Biological Processes represented by Networks

Intracellular biochemical processes are usually considered as networks whose nodes represent the actors of the complex biological system. These actors can be metabolites, enzymes, transcripts, genes etc. The edges in the networks represent relations between these actors, for instance, conversion, activation, inhibition, degradation, synthesis etc.

Biochemical networks are substructured in several layers: Genetic networks describe the processes of the gene expression (Kolpakov et al. 1998). Signal transduction networks show the stimulus-response mechanisms of a cell (Paek et al. 2004). Protein-protein networks display interaction between different proteins (Schwikowski et al. 2000). The network layer this work is focused on is the metabolic network (Michal 1999). Metabolic networks describe the steps by which metabolites are converted in enzymatically driven reactions.

Fig. 1.3 a) shows a typical metabolic network. In the diagram, the metabolites are represented by their names and the reactions by the connecting arrows. Biochemical reactions do not necessarily convert metabolites in a one-to-one relation but can convert several educt metabolites into several products. Hence, the interconnecting edges can have splitting and/or joining characteristics. In graph theory the term *hyperedge* denotes an edge that can have multiple start and end nodes. Consequently, a metabolic network is a *hypergraph*. Another way of representing metabolic networks is by inserting particular nodes for the biochemical reactions. By this, the network consists of two kinds of nodes whereas edges always connect nodes of different kind. This is called a *bipartite graph*. The bipartite graph representation of metabolic networks is depicted in Fig. 1.3 b) where metabolites are symbolized by rectangles and reactions by small squares.

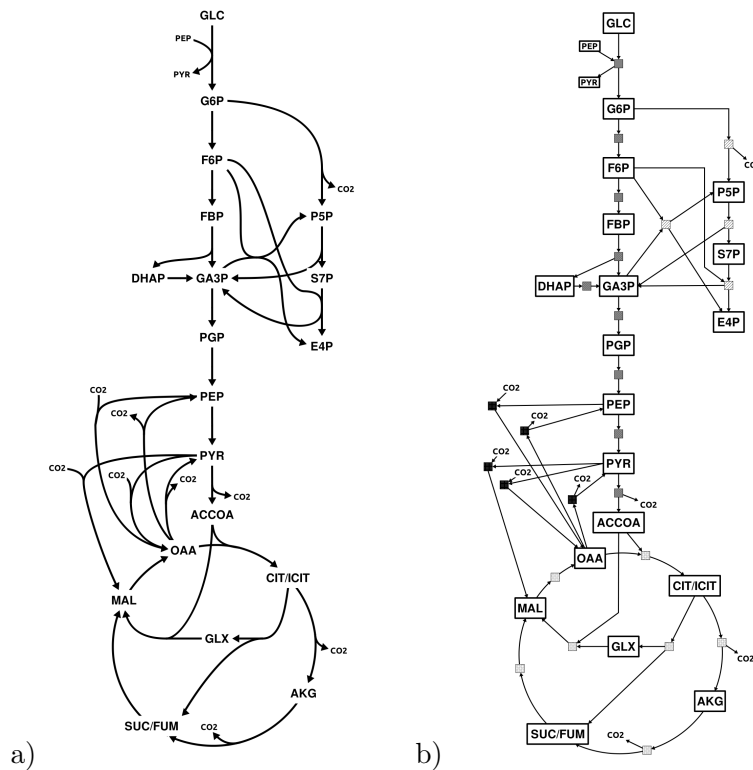


Figure 1.3.: A metabolic network in a hypergraph representation (a) and in a bipartite graph representation (b).

The different filled reaction symbols of Fig. 1.3 b) already indicate, that biochemical reactions are classified in so-called *metabolic pathways* being series of reactions which share a certain functional role in the metabolism. Prominent pathways are, for instance, the glycolysis for breakdown of glucose and other sugars and the citric acid cycle producing high-energy reduction equivalents (both displayed in Fig. 1.3 a and b). Pathways are the building blocks for the graphical representation of the metabolism (Michal 1998).

Another aspect frequently included in metabolic networks is regulatory interaction between metabolites and reactions (Noack et al. 2007). A metabolite can take influence on the velocity of a reaction without being a reactant. Here, positive effects are called *activation* and negative ones *inhibition*. These regulatory effects are usually displayed by a second kind of edge between the metabolites and the reactions.

## 1.6. Multi-Omics Data

Technologies like rapid sampling, screening robots, gel electrophoresis, gas and liquid chromatography (GC, LC), mass spectrometry (MS), nuclear magnetic resonance spectroscopy (NMR), DNA sequencing, micro arrays and others have been developed in recent

## Chapter 1. Introduction

---

years (cf. Gehlenborg et al. 2010). They allow fast and detailed measurements from the biological system in *in vitro* and *in vivo* experiments. Therefore, a huge and continually increasing amount of available multi-omics data arises continually from classical as well as high-throughput experiments (cf. Fig. 1.2):

- concentrations of the individual metabolites in the cell,
- uptake of extracellular nutrients as well as excretion rates,
- occurrence of RNA transcripts representing the amount of gene-to-protein translation processes,
- DNA sequences of the genome of an organism,
- amino acid sequences of the individual proteins,
- mass distributions of isotopic enriched compounds,
- concentrations of proteins and their localization and modification,
- kinetic parameters of enzymes,
- binding sites and concentration of regulating factors.

Beside experiments, simulations are performed producing all of these types of omics data *in silico*. Additionally, in combination with modeling and simulation further data occurs that is computationally derived from measurement data:

- intracellular flux rates (i.e. reaction velocities) recomputed from the isotopic distributions inside of the cell,
- quantified regulatory influences of certain metabolites on enzymes resulting from the enzyme kinetics combined with measured concentrations,
- molecular structures of enzymes,
- thermodynamic potentials of the metabolites derived from their concentration.

Beside data based on measurements, further complex information are related to biochemical networks, such as:

- kinetic laws of biochemical reactions,
- chemical structures of metabolites,
- exchange pathways of single atoms in metabolic reactions,
- literature citations,

- database references,
- research notes etc.

It is a big challenge of systems biology to post-process all the data arising from current research activities in order to enforce scientific insights.

## 1.7. Visualization of Data

The plain presentation of experimental data in scientific discussions and talks in form of tables has been observed at countless opportunities. This is as amazing as suspicious considering the broad presence of data visualization in today's life. It can be compared as if the daily weather forecast was presented by tables showing the temperature, sunshine hours, precipitation amount, air pressure, wind speed and direction and other information related to all cities in a country. However, in fact this data is visually displayed on the map of the country usually by color-coding the local temperature (cf. Fig. 1.4 a), graphical symbols representing sunshine and rainfall (Fig. 1.4 b), isobars, front lines as well as arrows indicating the wind speed and direction. The weather changes over a daytime are usually visualized by animation. In this way, the viewer can immediately identify the important information by keeping the eyes on their local region. The overall view of the map, furthermore, allows to understand the global coherences.

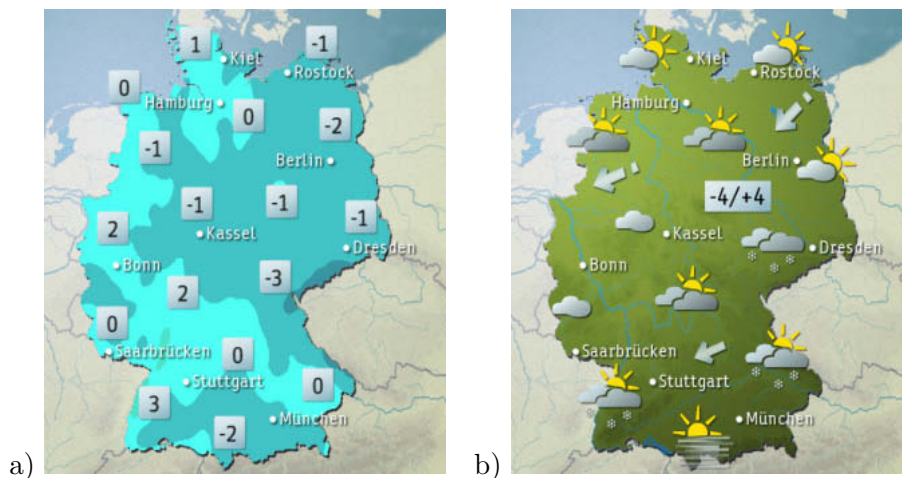


Figure 1.4.: Information visualization in meteorology. Examples taken from *ZDF* weather <http://wetter.zdf.de>.

Visualization is an important tool that helps to get a visual image of data connected to an underlying system which can be a map or, as in the context of this work, a biochemical network. Information visualization is the scientific discipline dealing with the visual representation of abstract data amplifying human cognition i.e. allowing the viewer

to acquire insights into the data inherent structures and relations (Card et al. 1999). In contrast to scientific visualization, information visualization is typically concerned with the representation of non-numeric, non-spatial, and high-dimensional data (Chen 2005).

In systems biology, information visualization is important in analyzing and understanding the plenty of experimental and simulated data concerning biochemical systems. The challenge lies “in the analysis of a huge amount of data to extract meaningful information and use them to answer some of the fundamental biological questions. Given the heterogeneity and the sheer amount of data it is a challenge to detect the relevant information and to provide a way to communicate the findings of the researcher in an efficient and appropriate way” (Pavlopoulos et al. 2008). “The pure amount of data and their heterogeneity pose a challenge for efficient visualization tools. The main goal of the visualization tools should be the intuitive representation of data to provide an efficient interpretation and to allow a hypothesis driven planning of the next experiment” (ibid.). Saraiya et al. (2005) wrote “it is critical that pathway visualizations depict this richness of information in order to be biologically relevant.”

Particularly, multi-omics data (cf. Section 1.6) has a direct relation to the nodes and edges of a metabolic network, for instance, the compound concentrations are related to the metabolites and the flux rates are related to the reactions. However, in the context of life sciences it is still predominating that experimental data is presented by isolated bar charts, scatter plots, histograms or function graphs of all state variables of the biological system even in case of time series of experimental data (e.g. Qeli et al. 2005, Wiklund et al. 2008, Schroer et al. 2009). This solution is not satisfying because the network context is completely lost. It hampers the ability to grasp the visualized data in a system-oriented manner. Hence, a network-integrated visualization of data is advocated here.

### 1.8. Existing Visualization Tools

In the context of biochemical networks many visualization tools have been developed in recent years. About that, informative reviews have been given by Suderman and Hallett (2007), Pavlopoulos et al. (2008) and Gehlenborg et al. (2010). Many software tools arise from a top-down approach of network modeling especially in the field of protein-protein interaction and gene expression profiling. Here, the network size is typically up to several hundred thousand nodes. In this situation, the network itself is the information to be visualized. Hence, a suitable automatic network layout, interactive network exploration and analysis of the interconnections in the network are of tremendous importance for the human viewer to gain knowledge from the visualization, for instance, by identifying highly connected nodes.

Networks from top-down approaches are not composed manually but are assembled from literature and database information (Yao et al. 2004) or automatically generated from measurement data (Zupan et al. 2003). Thus, powerful layout algorithms are necessary to generate a network diagram from the large-scale network topology. Tools like *Cytoscape* (Shannon et al. 2003), *Osprey* (Breitkreutz et al. 2003), *Ondex* (Köhler et al. 2006),



## 1.8. Existing Visualization Tools

*ProViz* (Iragne et al. 2005), *PIVOT* (Orlev et al. 2004), *VisANT* (Hu et al. 2005; 2007) and *Pajek* (Batagelj and Mrvar 2002) are designated for this field of application. They support the insights from the visualization by node filtering capabilities, collapsed subnetworks, network analysis, clustering etc.

Quite different is the situation in bottom-up approaches of network modeling. Here, networks are small, about a few dozens up to thousands of nodes and, particularly, the topology of the network is well-known in most cases. The quantities of the model components are of main interest. Hence, the network with all its nodes and edges is not the information that is to be visualized. In contrast, the quantitative experimental and simulated data and its relation to the network nodes and edges are to be visually represented in a network visualization. By this, an overall view of the data can be achieved. The visual representation of the network serves as environment for the information visualization.

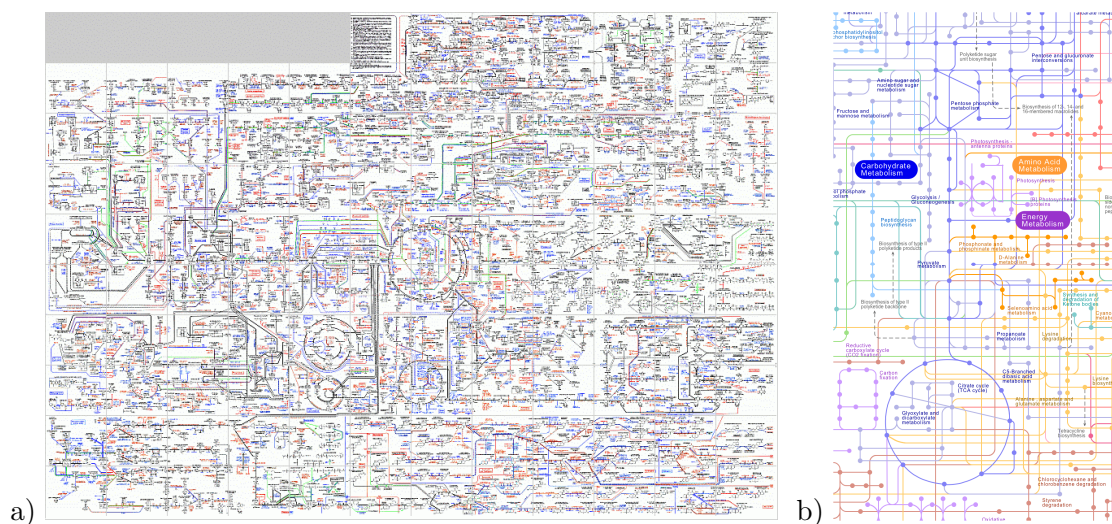


Figure 1.5.: Example metabolic networks: (a) Biochemical pathway map, taken from <http://www.expasy.ch/tools/pathways/>. (b) Sub-section of the global pathway map of *KEGG Atlas* (Okuda et al. 2008), taken from <http://www.genome.jp/kegg/>.

This work focuses on the visualization of data in metabolic networks because it arises from the research context of metabolomics and fluxomics. The metabolic network has two advantages significant for information visualization. The first is, almost all types of omics data have a direct or indirect relation to the nodes or edges of the metabolic network. Therefore, they can be visualized by being mapped to the visual appearance of the network items. A further advantage is that metabolic network drawings have become generally known among life scientists. The metabolic level is being visually presented since decades. A famous large-scale metabolic network has already been published as a wall chart in 1968 by Gerhard Michal (see Fig. 1.5 a). The arrangement of pathways in this wall paper has established itself in the representation of metabolic networks in

publications and textbooks (Stephanopoulos 1994, Madigan et al. 2003, Cortassa et al. 2002). For instance, Fig. 1.5 b) shows a subsection of the overview diagram used in the KEGG databases (Kanehisa and Goto 2000) for navigation in the pathway database contents (also compare Fig. 1.3). By serving the established layout conventions in a diagram, the viewer of a metabolic network visualization does not have to concentrate on localizing the single network components. In contrast, the data visualized by the network nodes has the observer's full attention in a well-known pathway map.

### 1.9. Restrictions of Existing Tools

Over one hundred different tools have been found in the literature that are mentioned in combination with network and/or omics data visualization. The full range of published visualization approaches cannot be exhaustively discussed here. The diversity of these tools with respect to their focus, scientific background and application field is very broad. Gehlenborg et al. (2010) make the important differentiation between tools that primarily focus on gene regulatory networks, protein-protein networks and metabolic networks. Many software tools for data visualization in metabolic networks have been published in recent years. Here, different scopes can be identified the individual tools aim at:

- Focus on diagram editing, for instance:
  - *Edinburgh Pathway Editor* (Sorokin et al. 2006)
  - *CellDesigner* (Funahashi et al. 2003)
  - *JDesigner* (Sauro et al. 2003)
  - *Cell Illustrator* (Nagasaki et al. 2010)
  - *MetVis* (Qeli et al. 2003)
  - “Java editor for biological pathways” (Trost et al. 2003)
  - *VitaPad* (Holford et al. 2005)
  - *PathwayEditor* (Byrnes et al. 2009)
  - *PATIKA* (Demir et al. 2002)
- Focus on network modeling and simulation. Examples:
  - *CellDesigner* (Funahashi et al. 2003)
  - *JDesigner* (Sauro et al. 2003)
  - *Cell Illustrator* (Nagasaki et al. 2010)
  - *INSILICO* (Müller et al. 2005)
  - *FCModeler* (Dickerson et al. 2001)

- Focus on database navigation, for instance:
  - *KEGG Atlas* (Okuda et al. 2008)
  - *PGViewer* (Tao et al. 2004)
- Focus on network topology visualization and layout, for instance:
  - *Cytoscape* (Shannon et al. 2003)
  - *Arcadia* (Villéger et al. 2010)
  - *Vanted* (Junker et al. 2006)
- Focus on network-integrated data visualization, for example:
  - *Cytoscape* (Shannon et al. 2003)
  - *ProMeTra* (Neuweger et al. 2009)
  - *MetVis* (Qeli et al. 2003)
  - *Paintomics* (García-Alcalde et al. 2010)
  - *Caleydo* (Streit et al. 2009)
  - *ArrayXPath* (Chung et al. 2004)
  - *PATIKA* (Demir et al. 2002)
  - *Vanted* (Junker et al. 2006)
  - *PathwayExplorer* (Mlecnik et al. 2005)
  - *VitaPad* (Holford et al. 2005)
  - *CellPublisher* (Flórez et al. 2010)
  - *PathwayEditor* (Byrnes et al. 2009)

Further tools whose scope is the visualization of metabolic pathways are listed by Gehlenborg et al. (2010). All listed applications focus on or are at least suitable in the context of metabolic networks. However, these tools come up with several limitations:

- Many tools cannot be employed because they are not publicly available (e.g. “Java editor for biological pathways”), they require expert knowledge to compile a running binary (e.g. *Arcadia*), the published version is instable (e.g. *VitaPad*) or the software is no longer maintained (e.g. *MetVis*).
- In all known approaches, the way to visualize data is implemented as a hard-coded, unchangeable part of the software. Data visualization methods only allow to be parametrized by the user.
- In many cases the visualization methods address very specialized application fields and/or require specific data formats (Killcoyne and Boyle 2009).

- Many tools realize a user guidance that is too technical for a target user group from life sciences (Killcoyne and Boyle 2009, O’Donoghue et al. 2010).
- Although a plenty of network visualization tools deal with network layout the established traditional network layout is insufficiently supported (Saraiya et al. 2005).
- Most tools have little degrees of freedom concerning the network appearance (e.g. *Paintomics*, *PathwayExplorer*, *CellDesigner*).
- In many cases, visualization is strictly separated from network drawing/modeling (e.g. *Paintomics*, *PathwayExplorer*, *ProMeTra*, *Arcadia*).
- A crucial gap in network data visualization is the representation of time. Most tools are not capable to visualize time series of data (Pavlopoulos et al. 2008, Suderman and Hallett 2007).

These restrictions make the realization of a new approach of visualization tailored to the application field of metabolomics and fluxomics reasonable.

### 1.10. Research Context

This research project began at the University of Siegen in the Simulation Group of the Institute of Systems Engineering in 2007. Since 2009 the work was continued at the Institute of Biotechnology in the research center Jülich within the Modeling and Simulation Group.

The project was partly funded by the German Research Foundation DFG (project grant WI 1705/13). Furthermore, Evonik Industries supported the research project financially within the SysMAP project co-funded by the German Ministry BMBF (project no. 0313704) and within the EU-funded SysInBio project (project no. 212766).

### 1.11. Contribution of this Thesis

This dissertation presents a novel approach to customizable data visualization in the context of metabolic networks. The approach has been realized in the software tool *Omix* that is successively introduced in this thesis. The name *Omix* is derived from the above-mentioned neologism “omics” standing for the vast possibilities in visualizing multi-omics data. The tool serves as experimental environment for the novel visualization methods discussed here. The restrictions mentioned in Section 1.9 lead to following requirements:

#### **Ease of use**

An intuitive user guidance is to be realized allowing a fast access to the functionality of the visualization tool even for newcomers. The software usability must be suitable for all levels of expertise corresponding to the interdisciplinary user community in systems biology.

### **Network construction**

It must be possible to create network diagrams from scratch as well as load networks from databases. This includes the combination of different small networks to a bigger one.

### **Rapidly customizable style**

The visual appearance of the network diagrams should be adaptable to individual preferences and requirements of the software user.

### **Focus on different levels of detail**

Metabolic networks should be abstracted in levels of different information depth in order to hide complexity or to show different facets of datasets if available.

### **Flexible visualization**

The way of data visualization in the network context should be freely and quickly reconfigurable in order to deal with frequently changing requirements in the interplay of experiments and data evaluation. Particularly, the dynamic visualization of time dependent data should be supported.

### **Network independent visualization**

It should be possible to apply different visualization methods on the same network and, vice versa, to use the same method on different networks.

### **Familiar layout**

The major goal of visualization is a rapid and comprehensive understanding of complex multi-omics datasets. For this reason the network layout should be familiar to the user in order to facilitate orientation and navigation. Techniques for drawing networks according to the traditional layout conventions must be provided.

### **Compatibility and extensibility**

Networks ought to be reusable in different contexts of simulation and data evaluation. Therefore, established model describing file formats must be supported. Furthermore, the software must allow to be combined with other software tools in data processing work flows. In order to realize extensibility and adaptability to changing requirements the visualization tool should be equipped with a plug-in interface.

The overall vision this thesis is inspired of is a software that is the visual front end in a scientific work flow as shown in Fig. 1.6. The work flow begins with the construction of a network model in form of a diagram. The next step is the export of the network to an external simulation framework. The user can start and control the simulation runs via the network editor. Finally, data resulting from experiments and simulations are visualized in the network diagram. Here powerful techniques are available for the convenient representation of data in the network. The diagram can be used to present the network model as well as the experimental and simulated data in scientific discussions

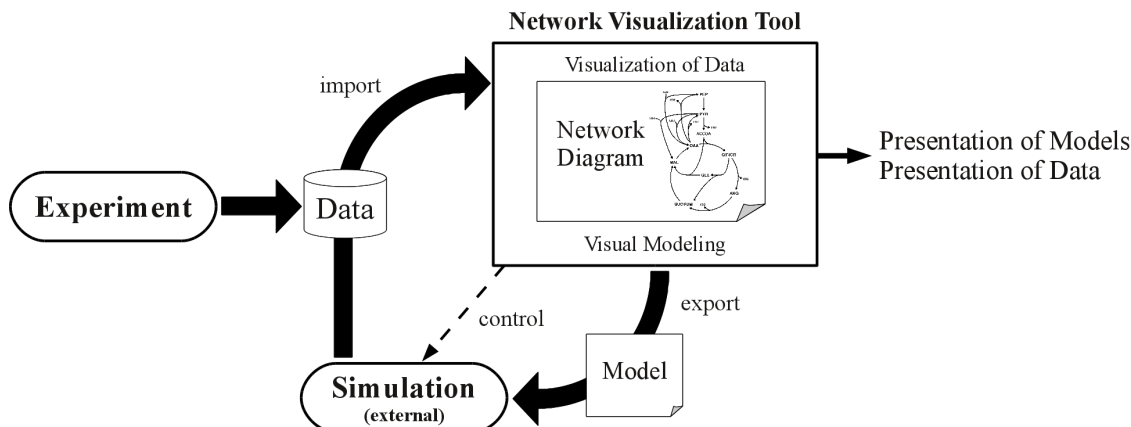


Figure 1.6.: Vision of a visual front end in a scientific work flow consisting of network modeling, simulations and experiments, data visualization and presentation of results.

and talks. The network visualization tool is designed such that the simulator and data formats sketched in the constellation are freely exchangeable.

## 1.12. Organization of this Thesis

In order to reach the goal given by the high vision (cf. Fig. 1.6) and to meet the requirements mentioned above this thesis is concerned with four main topics that are discussed in single parts of the document:

### Part I: Network drawing

Chapter 2 introduces the necessity of a new network drawing tool for visualization purposes. In Chapter 3 the particular features of the network drawing editor are discussed. The drawing features have in parts been published in (Droste et al. 2009a;b; 2010a; 2011b).

### Part II: Network layout

This Part introduces a novel semi-automatic layout approach for metabolic networks that is first of all motivated in Chapter 4. Thereafter, Chapters 5 and 6 introduce two novel layout techniques. Finally, Chapter 7 discusses an example work flow of applying the layout techniques in drawing diagrams. The semi-automatic layout techniques have recently been published in (Droste et al. 2011b).

### Part III: Visualization

In this Part, the customizable visualization approach is introduced. After giving an introduction in the issues of this part in Chapter 8, the script-based visualization

approach is presented in Chapter 9. Finally, Chapter 10 gives a number of application examples for visualization. Parts of these chapters base on original publications (Droste et al. 2009a; 2010b; 2011a).

### **Part VI: Extensibility**

This Part deals with the concept of plug-ins for the visualization software which is first of all motivated in Chapter 11. Chapter 12 illustrates the concepts and features of the plug-in interface and, finally, Chapter 13 lists available plug-in solutions. Excerpts of the chapters have been published in (Droste et al. 2008b;a).

Between the single Parts feature sheets give an overview about the novel developments introduced in this work concerning the network drawing (p. 41), network layout (p. 42), visualization (p. 103) and extensibility (p. 104). The feature sheet pages can be unfolded to be present during the reading of the dissertation.

Finally, Chapter 14 concludes the thesis by summarizing the achieved results and giving suggestions for potential future work. An Appendix supplies a drawing tool survey and a user case study for the evaluation of the semi-automatic layout approach introduced in Part II. The original publications of the author are listed preceding the general bibliography at the end of this thesis. Lists of mathematical symbols and abbreviations used in the dissertation as well as a glossary precede the list of contents.





**Part I.**

**Network Drawing**



## Chapter 2.

# Introduction to Network Drawing

*“Usability is an important consideration in the design of products because it is concerned with the extent to which the users of products are able to work effectively, efficiently and with satisfaction.”* (ISO 9241-11 1998)

### 2.1. State-of-the-Art in Network Drawing

In Chapter 1.9, a number of tools have been mentioned that realize drawing capabilities for metabolic network diagrams. However, as a prevalent practice biologists and biochemists predominantly use common graphics software or even office programs for drawing metabolic network diagrams (cf. network diagrams taken from publications shown in Fig. 4.1 on page 46). Clearly, all-purpose software solutions for graphics design enable the user to draw adequate diagrams. However, these tools have several restrictions that make the employment of a rather specialized drawing tool recommended:

- The special requirements and properties of metabolic networks are not supported because the software aims at a common drawing approach. For instance, the software cannot classify the graphical items as metabolites, reactions, pathways and so on. They are just single items composed to a figure.
- The diagram inherent network topology cannot be reused for any other purpose, for instance, simulation or network analysis.
- Common graphics software and office tools do not provide techniques for visualizing data in the diagram. When quantitative data shall be mapped on the appearance of the network components, every single drawing step has to be done manually.

In an email survey, scientists from systems biology have been asked if they ever have drawn a network diagram and what software they have used for this purpose, or otherwise, what tool would be first choice for drawing a metabolic network diagram. Nine people responded to that questionnaire. The drawing software *CorelDraw* has been mentioned 7 times followed by *Powerpoint* (6 times) and other common drawing tools (5 times). *ChemDraw* has been mentioned once, which is a drawing tool for chemical molecular structures. Only one time, a special-tailored visualization tool for biochemical networks

has been stated (*CytoScape* Shannon et al. 2003). Here, the question comes up: Why are existing drawing tools for biochemical networks rarely accepted for drawing diagrams in the community of the life sciences?

Possible answers may lie in the normal software usage patterns of life scientists and in the effort that is necessary to become familiar with specialized tools. In the normal work routine of biologists, software systems are just utilities. Usually, they do not spend much time on finding new software tools and learning the deep features of new applications because this does not lie in their primary professional interest. The scientists are familiar with office software because these tools are in daily use. When network diagrams are required, e.g. for presentation or publication purpose, office software is at hand. Additionally, professional or semi-professional graphics software tools are widely spread. If one is familiar with a graphics design program it is logically first choice for creating comprehensive drawings.

### 2.2. Tool Survey

In Appendix A widespread common drawing tools are compared with drawing tools for biochemical network diagrams. Three tools with objectives similar to this work have been selected for the drawing tool evaluation: *CellDesigner* (Funahashi et al. 2003), *Vanted* (Junker et al. 2006), and *Cytoscape* (Shannon et al. 2003). The tool comparison shows that the user guidance of *CellDesigner*, *Vanted* and *Cytoscape* concerning the drawing of nodes and edges is very different from common graphics and office software. Some criticisms are:

The drawing capabilities of *CytoScape* are not directly apparent. The corresponding functionalities are not available as toolbar buttons or menu entries. In *CytoScape* and *Vanted*, the interaction design for adding nodes and edges to the drawing is very unusual. The access to visual properties of the network components is cumbersome. The lack of biologically motivated network symbolism in *CytoScape* and *Vanted* hinders the intuitive access to the software for a biologist.

Only *CellDesigner* has a user guidance similar to, for instance, *Corel Draw*. However, in *CellDesigner* biochemical networks are drawn as process diagrams according to the Systems Biology Graphical Notation (SBGN) (Le Novere et al. 2009). The appearance of these process diagrams is very technical and reminiscent of electrical circuit diagrams. One must like this representation. There is no way to change it in *CellDesigner* except adapting colors and line thickness.

The network representation of *CellDesigner* rather corresponds to engineers' conceptions of biochemical processes whereas *CytoScape* and *Vanted* realize the network concepts of computer scientists (cf. Saraiya et al. 2005). It is hard to follow these conceptions for biologists. This also includes the vocabulary used in the graphical user interfaces. The biological sense of terms like "node", "edge", "entity" and "state transition" is not directly perspicuous for life scientists.

## 2.3. Proposed Standards

Electrical circuit diagrams have a clear symbolism which is a long established, worldwide applied standard. In the context of biochemical networks the situation is quite different. Several figures in this dissertation that are taken from various sources prove the lack of standards concerning the network symbolism (cf. Fig. 1.3 on page 7, Fig. 1.5 on page 11, Fig. 4.1 on page 46, Fig. 4.2 on page 47 a).

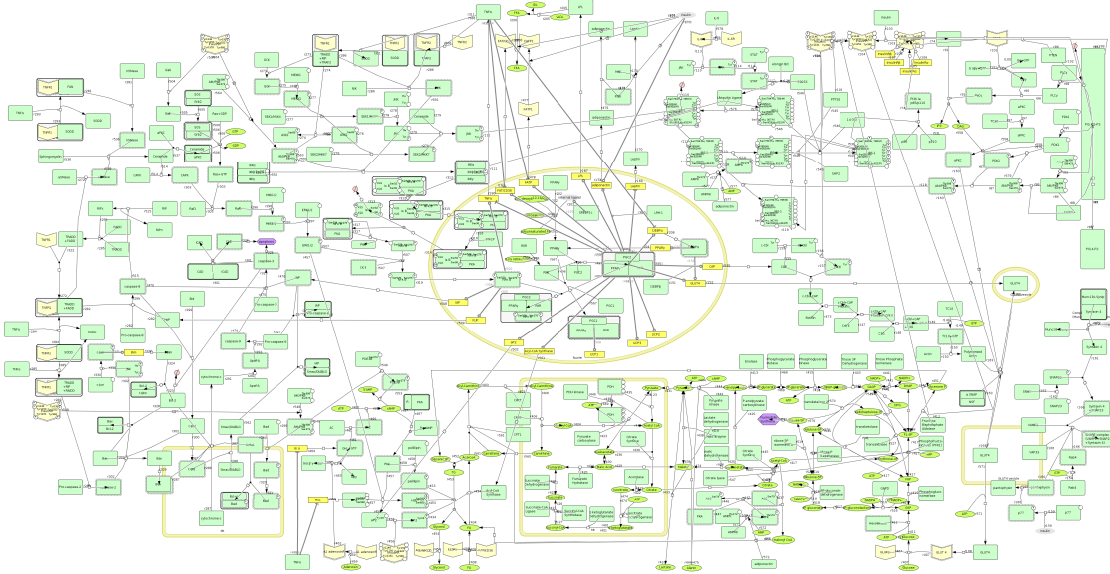


Figure 2.1.: Example diagram in the SBGN vocabulary created with *CellDesigner*. The model is published as supplemental material of (Kitano et al. 2004).

In recent years, several efforts have been made to standardize the representation of biological entities and relations in biochemical networks, for instance, by Pirson et al. (2000) or in the BioD language (Cook et al. 2001), MIP diagrams (Kohn et al. 2006) and the SBGN (Kitano et al. 2005, Le Novere et al. 2009). However, these proposed standards are not accepted and even not widely known in the community of the life science. For instance, Fig. 2.1 shows a network diagram in the SBGN vocabulary. The diagram has a rather technical appearance which hampers the acceptance by life scientists. Furthermore, in SBGN one is confronted with a huge set of network symbols which might confuse the novice, especially, because their biological meaning is not evident in each case.

## 2.4. Requirements

The development of a new network drawing software must be strongly oriented at the requirements, software usage patterns and knowledge of the intended end user group.

## Chapter 2. Introduction to Network Drawing

---

Saraiya et al. (2005) found that life scientists “are skeptical about the biological value of current pathway visualizations. When considering cost vs benefit, the cost seems to outweigh the benefits. They are reluctant to invest time required to overcome the learning curve for many of these systems. A large amount of effort is required to gain biologically meaningful insight for specific projects from most of these systems.”

Learning new software is labor- and time-intensive. Hence, it is important to diminish this effort by meeting a familiar user guidance. Here, *de facto* standards arose in the last decades especially due to the popularity of office software. By miming the user guidance of the popular drawing applications ease of access is warranted.

Proposed standards for network diagrams like SBGN should not be met due to a lack of acceptance by the users. On the other hand, “visualizations that look like simple ball-and-stick graph drawings are likely to be considered information-poor, and not biologically meaningful” (Saraiya et al. 2005). A new editor for metabolic networks must provide a clear, biologically motivated network symbolism with many degrees of freedom for highly customizable network diagrams.

## Chapter 3.

# The Network Diagram Editor

Visualization of data in metabolic networks requires the availability of network drawings. Therefore, a highly flexible drawing tool has been created for the manual preparation of network diagrams. The tool aims at providing extensive drawing features towards established standards in user guidance as a highly customizable graphics tool for the application field of metabolic networks. The network drawing facility is the first module of *Omix* introduced in this thesis.

The tool has been developed in *Java* (Oracle Corporation 2011) by using *Qt Jambi* (Sletta 2006, Qt Jambi Team 2011) for realizing the graphical user interface. *Qt Jambi* is the Java bindings to the *Qt* library (Nokia Corporation 2011). *Qt* allows to develop sophisticated drawing features by providing a 2D scenegraph technology called *Graphics View Framework*.

The software is provided for download on the website [www.13cflux.net/omix](http://www.13cflux.net/omix) and licensed for non-commercial, academic use. The download is available after registration. The website furthermore provides manuals teaching to employ *Omix* written for novices and advanced users (Droste 2008a;b; 2011). The source code of the software is not published. In the following, the outstanding features of the network editor are discussed in more detail.\*

### 3.1. Ease of Use

Since the late 1980s, computer science is aware of the value of *usability* (Dumas 2007). It is desirable to create usable and functional programs that improve productivity and allow to be easily learned (Gould 1988). The ISO defines usability as “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use” (ISO 9241-11 1998). Characteristic for *usability engineering* is the user-centric design of software. As illustrated in Chapter 2, a user-oriented approach is *sine qua non* for the acceptance of bio-related visualization software by life scientists (Saraiya et al. 2005). “The principles of any system developed in the research environment must be simple so that end users [...] can quickly learn and use the system” Killcoyne and Boyle (2009) wrote.

---

\*Excerpts of this chapter have been published in (Droste et al. 2009a; 2011b).

The drawing tool *Omix* has been developed in close communication with its end users. The intended end users of the software are, in general, all participants in systems biology research (cf. Chapter 1.2). A special focus at the design of the network editing features lies on researchers from the life sciences, i.e. biologists, biotechnologists, biochemists, chemists, biomedical researchers etc. (Saraiya et al. 2005). Here, the most important requirement is to satisfy familiarity. Life scientists are used to work with office software and know the user guidance of drawing tools. Hence, *Omix* allows to be operated with the intuition learned from daily employed software.

Fig. 3.1 shows the main window of the software consisting of a menubar, several toolbars and, in particular, a drawing area in the center. *Omix* is document-based because the scientific user community is most comfortable with this kind of content handling (Killcoyne and Boyle 2009). The program offers a so-called multiple document interface (MDI) for the parallel work on several networks. An MDI can contain multiple internal windows inside of the main window that operates as window manager, a technique that is implemented in most document-based tools.

Internationalization is an important prerequisite for usability (Gould 1988). The language of the dialog texts on the graphical user interface (GUI) can be changed between English, German and French. In order to minimize the user's effort to performing similar operations the GUI saves and reuses preferences of the user, for instance, window position and mode, recent documents (presented as shortcuts), preferred directories for loading and storing documents, last selected image export formats and so on.

The user guidance of the network editor corresponds to a vector graphics software. Sophisticated graphics can be created and exported to the most important vector and pixel oriented image formats (PNG, JPG, SVG, PDF and others). The drawing area is equipped with rulers and a background grid. A number of tool buttons on the left vertical toolbar can be activated to add graphical items on the drawing area (cf. Appendix A). The drawing tool is self-explanatory (ISO 9241-110 2006), i.e. the button icons indicate their functionality with a well-noticeable plus sign decorating the graphical symbol. Furthermore, the buttons are equipped with explaining "tool tips" being small prompts appearing when the mouse hovers an interactive element. While the different editing utilities are active, the mouse cursor indicates their functionality. For instance, an arrow decorated with a graphical symbol and a plus sign indicates that the corresponding element can be inserted by mouse click.

The graphical elements on the drawing area can be selected, moved, deleted, resized and arranged to item groups. For editing shapes and lines the editor uses Bézier splines (Salomon 2006), a primarily applied technique in vector graphics programs. The user can split lines into several segments, convert the single segments into straight lines, arcs or curves and define the smoothness at the join points. All editing steps are recorded and can be undone by the user.

Most important features of the software are available on toolbars. Basically all functionalities are accessible on the menubar. Furthermore, the GUI uses the platform-dependent standard keyboard shortcuts for file operations, print, undo/redo, document search etc. The network drawing editor is fully documented in the *Omix User Manual* (Droste 2008a),



## 3.2. Network Symbols

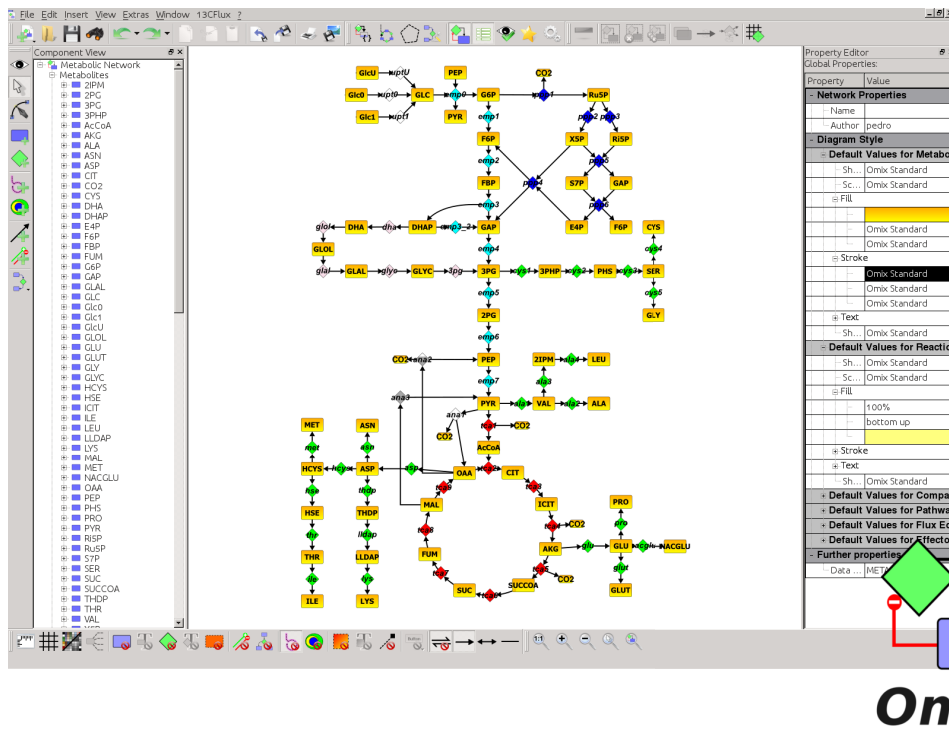


Figure 3.1.: Screen shot of the *Omix* main window containing a drawing area in the center, several toolbars and side windows. Additionally, the figure shows the logo of *Omix* bottom right.

an easy readable guide to the functionality of the software for newcomers. The design of the user interaction has been continually discussed with end users from the life sciences.

In a recently performed case study a group of sixteen voluntary participants was asked to solve a network drawing exercise with *Omix* and to assess its ease of use. The average gave a degree of 6.3 (1 corresponds to “most difficult to use” and 10 “very easy”). Seven participants gave additional comments on the user friendliness of the tool. Furthermore, a short questionnaire was sent to about eighty users that are registered at the download server. They were asked to value the software’s ease of use for newcomers. Seven people responded to the request valuing the ease of use with 6.7 in average (same scale).

## 3.2. Network Symbols

The network editor is designed with a deliberately small set of network symbols with a biologically meaningful semantics. As formerly introduced in Chapter 1.5, metabolic networks are commonly represented as directed hypergraphs where metabolites are symbolized by nodes and the connecting edges correspond to the biochemical reactions. Here, every edge may have several sources and destinations (i.e. substrates and products of a

reaction step). However, such directed hyperedges can also be represented in a bipartite graph structure by introducing individual nodes for reactions. Networks in this approach base on latter convention.

### 3.2.1. Network Nodes and Edges

As depicted in Fig. 3.2 the editor provides two distinct node symbols for the representation of metabolites (rectangles) and reactions (diamonds) in the diagrams. Reactions are labeled with an identifier that is unique in the set of reactions. Metabolites may occur multiply in a network diagram as discussed later. Reaction and metabolite symbols can be connected by edges indicating a flux relation (see the blue lines with arrowhead in Fig. 3.2). These so-called *flux edges* are directed, i.e. that a metabolite can in general be identified as substrate or product of a biochemical reaction.

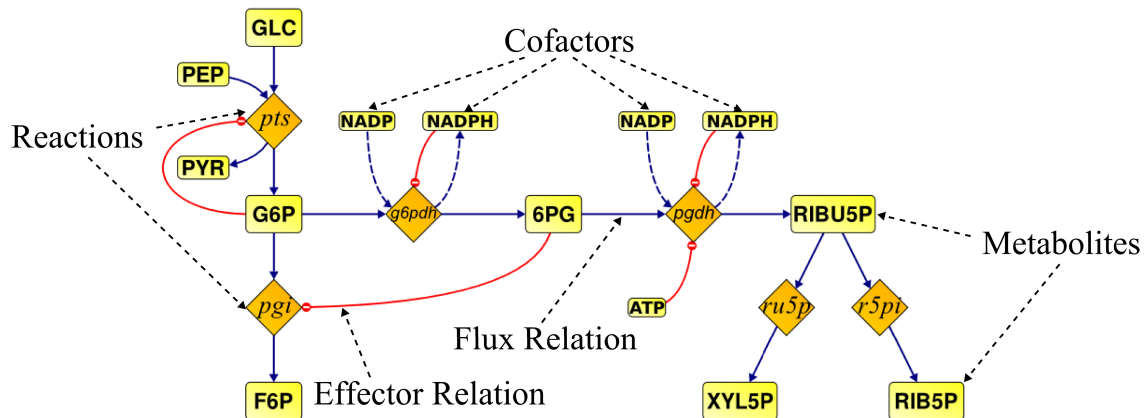


Figure 3.2.: Example for a metabolic network in *Omix*. Reactions are displayed by diamonds, metabolites by rounded rectangles. The edges with a triangular arrowhead indicate a flux relation, the edges with a circle arrow a regulatory effect.

A second type of connection between metabolites and reactions, the so-called *effector edge* represents regulatory effects in the network (Noack et al. 2007). Metabolites can inhibit or activate reactions which is indicated by the appearance of the individual effector edge. It is colored green and shows a plus sign in its circuit end marker in case of activation. Inhibition is indicated by red color and a minus sign as shown in Fig. 3.2. When the kind of regulation is unknown the effector edge is colored yellow with a question mark character displayed in the end marker.

Basically, regulation can take place on different levels. On the metabolic level, compounds can take effect on the activity of enzymes. On the genome level, the production of enzymes can be affected or stimulated by the occurrence of the regulating metabolite (Michal 1998). Here, effector relations represent regulation on all these levels as an abstraction.

### Formalization

The above described network symbolism leads to a directed graph  $G = (V, E)$  describing a metabolic network with set of nodes  $V = M \cup R$  and  $M \cap R = \emptyset$  containing metabolite nodes  $M$  and reaction nodes  $R$  connected by edges  $E \subseteq \{M \times R, R \times M\}$ . The set  $E$  is subdivided into disjoint sets of the flux edges  $f \in F \subseteq E$  and the effector edges  $x \in X \subseteq \{M \times R\}$ . A set of all edges connected to a node  $v \in V$  is denoted by  $E(v) \subseteq E$  and the connectivity of  $v$  by  $|E(v)|$ .

### Reaction Directions

Biochemical reaction steps are usually reversible (bidirectional) meaning that the enzymatic conversion proceeds in forward as well as in backward direction. Under certain thermodynamic conditions a reaction is expected to be irreversible (unidirectional). In contrast, in this approach flux edges are always directed even when they connect a reversible reaction. Reversibility is implemented as a flag of reaction nodes which can be quickly changed without redrawing the connected flux edges. The reversibility of reactions can be displayed in several ways. The user has the choice between showing a reversibility icon annotating the flux edges, showing bidirectional arrowheads and displaying flux edges without arrows.

The actual direction of a flux edge connected to a reversible reaction is considered as a nominal direction. Here, an advanced option is provided by the network editor that enables the user to reverse the directions of all edges of a reaction. This option wraps a numerous drawing steps in one action. Reversing a reaction direction in this way causes a real stoichiometric change in the network. A second option is the visual property “inverted” of the reaction symbols. By setting this option only the visual representation of the edge directions is changed. These two options are novel and very useful for modeling processes and visualization because assumed reaction directions frequently differ in the literature and stoichiometric shifts of reactions are not seldom in non-steady state experiments.

### Properties of Flux Edges

In combination with metabolic reactions it can be distinguished between main metabolites and cofactors. Examples for cofactors are the energy carriers NADP and NADPH that occur at many reactions in the metabolism. Here, the high-energetic NADPH is converted to the lower-energetic NADP. This parallel reaction step provides the energy that is necessary to energetically upgrade another biochemical compound in the main reaction step. This particular cofactor role of a metabolite in a reaction is property of their relation. Hence, the flux edge can be marked as to represent a cofactor relation (cf. Fig. 3.2). Flux edges that represent a cofactor relation are classified as *cofactor edges*  $F' \subseteq F$ .

Another important property of metabolites is, that they can participate in a reaction in an  $n : m$  relation with  $n, m \geq 1$ . For instance, two molecules of metabolite A are combined to one molecule of metabolite B. In a correct model two edges would be necessary

## Chapter 3. The Network Diagram Editor

---

to represent this relation. In this approach, the flux edges have a coefficient representing the proportions the metabolites participate in the biochemical reactions. Particularly, this coefficient is specified as real number. By this, fractional proportions are possible that are required in some cumulative reactions describing biomass formation.

### Duplicated Nodes

A distinguishing feature of metabolic networks is the existence of a few nodes that show a high degree of connectivity. This especially applies to the cofactors participating in many enzymatic reactions over the entire network. Displaying a metabolic network with only one representative of each of those cofactors would lead to a chaotic diagram with a myriad of edge crossings and edges that span over the entire diagram as shown in Fig. 3.3. In order to reduce edge crossings without affecting readability the diagram editor provides the possibility to duplicate metabolites. By this, a metabolite can appear several times in a network; i.e. several metabolite items in the diagram with the same label  $M.l$  represent one single metabolite species  $M(m) = \{m' \mid m'.l = m.l\}$ . In this way, the connectivity of the visual representation is strongly reducible.

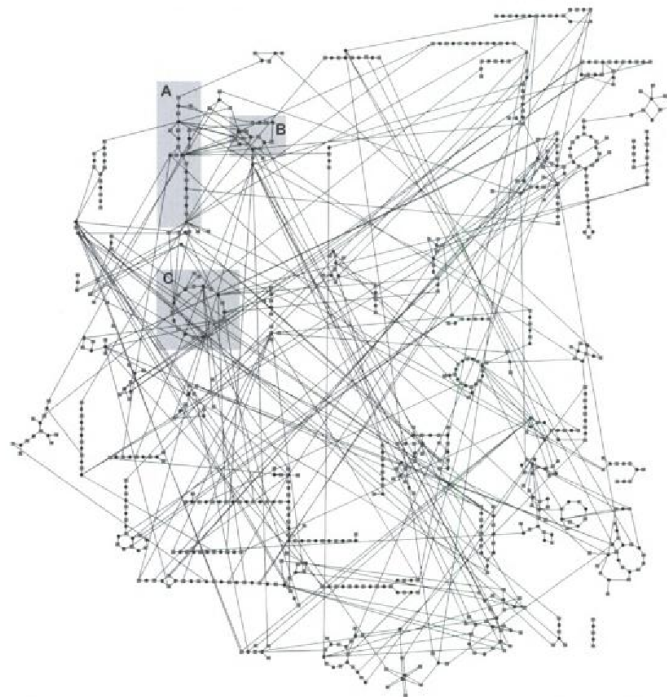


Figure 3.3.: Genome-scale network diagram without duplicated nodes. The figure was taken from (Müller et al. 2005).

An argument against node duplication is that it can distort the real network topology and, in this way, affect the correct interpretation of the diagram (Bourqui et al. 2006). Nevertheless, this does not apply to duplicated metabolite nodes in general. For instance, it is a well-established practice to have multiple representatives of cofactors in a diagram (cf. NADP and NADPH in Fig. 3.2 on page 28). The duplication of cofactors does not lead to a misinterpretation of the diagram because the main focus lies on the reactions' main metabolites.

The situation is quite different in combination with reaction nodes. It is difficult to obtain an overview of all substrates and products of a reaction when there are several representatives each connected with different metabolites. In such cases, the reaction stoichiometry can easily be misinterpreted. Hence, reaction nodes cannot be cloned in this approach.

Duplication of metabolite nodes can be performed by selecting the intended metabolite and pressing the duplication tool button. Next to the original one a cloned node appears, which is initially unconnected with the network and can be connected by manually inserting edges. If a metabolite node is multiply connected node duplication and the subsequent reconnection to the network can be strongly simplified. Here, the user can choose the option of duplicating the metabolite node  $m$  once per edge  $e \in E(m)$  as illustrated in Fig. 3.4. If this is done, several new nodes are created each connected to one of the edges. Thus, multiply connected cofactor metabolites can be distributed all over the network diagram. Although duplicated nodes are supported in other network diagram editors this particular feature of automatic duplication has not been found in any of them. The feature is strongly useful when a network topology is imported from third party sources which do not support duplicated nodes, as is commonly the case (e.g. SBML, Hucka et al. 2003, and FML, Dalman et al. 2010a).

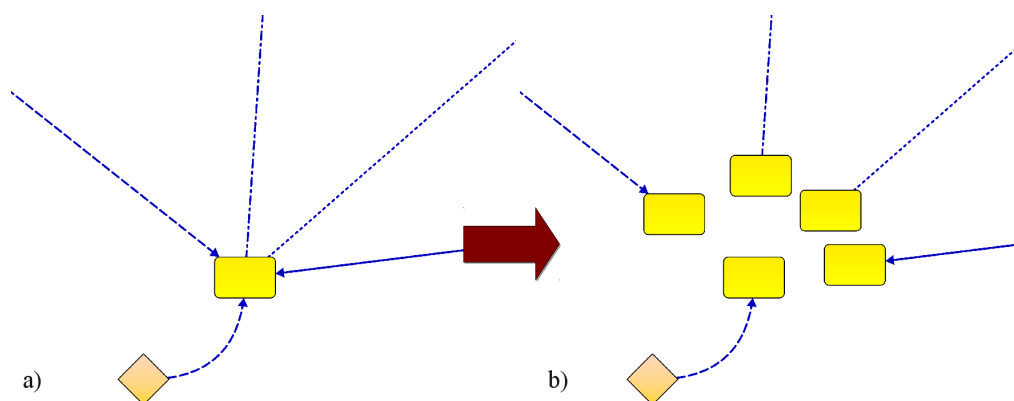


Figure 3.4.: Duplicating a metabolite once per connection. After duplication, the newly inserted clones are arranged around the original node. Subsequently, a manual repositioning step may follow.

### 3.2.2. Metabolic Pathways

In metabolic networks, series of reactions are grouped to *metabolic pathways* (cf. Chapter 1.5). This way to organize the metabolic network is fully supported by the drawing editor. A metabolic pathway  $P \subseteq R$  is a set of enzymatic reactions defining a special functional module of the network. Let  $P_1$  and  $P_2$  be two pathways, then  $P_1 \cap P_2 = \emptyset$ . Like metabolites and reactions, pathways are also labeled with a unique identifier.

The periphery  $\Pi_r = (E_r, M_r)$  of a reaction  $r$  is given by its connected edges  $E_r = E(r)$  and the connected metabolites  $M_r = \{m \in M \mid \exists e \in E(m) \cap E_r\}$  (distance 1). The peripheries  $\Pi_r$  of all reactions of a pathway  $P$  lead to the “pathway subgraph” which we denote  $G_P = (P \cup \bar{M}, \bar{E})$  with

$$\bar{M} = \bigcup_{r \in P} M_r \text{ and} \quad (3.1)$$

$$\bar{E} = \bigcup_{r \in P} E_r \setminus \{X \cup F'\}. \quad (3.2)$$

Equation 3.2 implies that effector edges  $X$  and cofactor edges  $F'$  are excluded from the set of pathway edges.

An outstanding feature presented here is, pathways are not only internal sets of reactions but also visualized in the diagram by highlighting the visible edges  $\bar{E}$  of the pathway subgraph  $G_P$  as is shown in Fig. 3.5 on the next page. Initially, a pathway is created as an empty set and by this invisible in the diagram. During the drawing process, reactions can be assigned to pathways. A pathway becomes visible when  $\bar{E}$  contains at least one element, i.e.  $|\bar{E}| > 0$ . The visual representation of pathways by highlighting the edges of the involved reactions with a contour is novel and unique in the present literature. Additionally, this pathway shapes are suited for data visualization purpose as illustrated in Part III.

In contrast to other approaches where metabolic pathways are considered as series of reactions and metabolites (Papin et al. 2003), pathways are deliberately defined different here. Metabolites are excluded from the definition of metabolic pathways in order to reduce redundancy since most metabolites participate in many pathways in the metabolism. The association of metabolites to pathways is established by the pathway subgraphs. Furthermore, according to the upper definition, pathways do not necessarily demand a consecutive series of reactions. This, consequently allows the composition of pathways consisting of non-connected parts in principle.

### 3.2.3. Compartments

Another structural element in metabolic networks basing on real spacial subdivision of the living cell is given in the *compartments*. Prominent compartments are for instance the mitochondria, chloroplasts and the nucleus. Compartments in the network editor are drawn as rectangular areas in the diagram as shown in Fig. 3.5 on the facing page. In particular, metabolites that are located inside of the compartment shape are associated with the

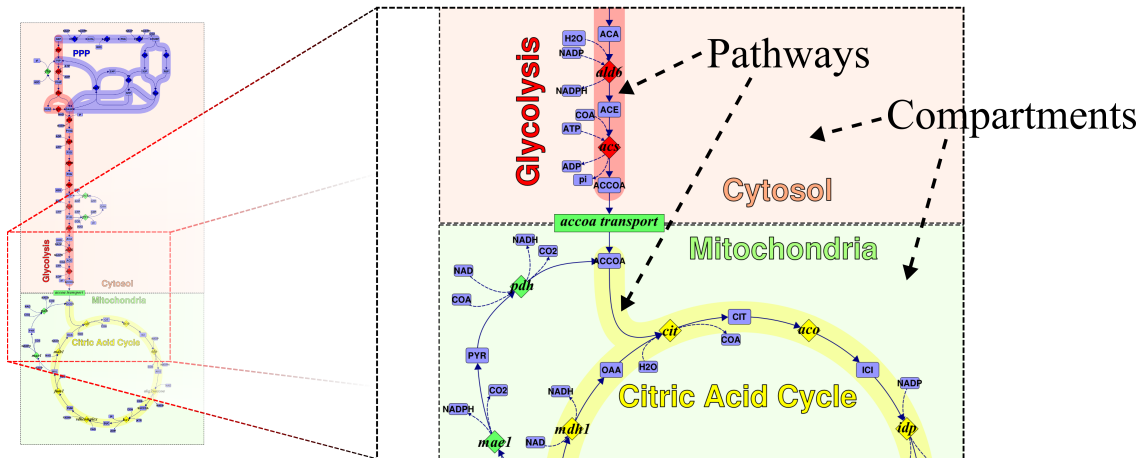


Figure 3.5.: Representation of metabolic pathways and compartments in *Omix*.

compartment. Since this network drawing approach allows duplicated metabolites there can be multiple clones of one metabolite positioned in different compartments. In this case, the internal network model contains distinguished “pools” of the same metabolite. A pool represents the occurrence of a metabolite in the cell given by its concentration. The concentration of a compound can vary between the compartments because they are physically separated by the compartment membranes.

### 3.2.4. Additional Network Components

The above presented set of network symbols is fully sufficient for the design of biologically meaningful and, likewise, well readable network diagrams. Nevertheless, the diagram editor tool provides a custom node type and edge type in order to offer more degrees of freedom in editing diagrams. Custom edges can connect all types of network nodes, for instance, two metabolites or a custom node and a reaction. These components can be used in the network diagrams occupied with arbitrary semantics the user requires.

## 3.3. Network Style

Michal (1998) stated that style is a very personal thing. The way biochemical networks are fashioned greatly differs between different authors and individual aesthetics plays an important role. Therefore, a network drawing software must offer many degrees of freedom in adapting the visual appearance of a network diagram to individual preferences and requirements.

The here realized network drawing approach allows to change the style of network components in various ways. Fig. 3.6 shows a selection of available style parameters of nodes and edges, the user can access. The amount of visual properties includes the shape,

fill, text and stroke of reactions and metabolites as well as the curve shape, line width, arrow size and color of edges in the network. Furthermore, the appearance of pathways and compartments can be customized in various ways. The main window contains a property editor (right window component shown in Fig. 3.1 on page 27). The property editor always lists the properties of the currently selected network component and, thus, allows the user to change them.

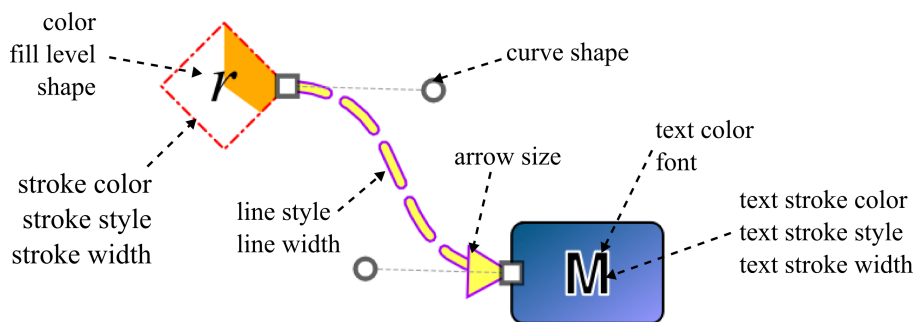


Figure 3.6.: Visual properties of nodes and edges in *Omix* diagrams

An outstanding feature of the editor is: changing visual properties cannot only be done on individual network components one by one, but can additionally be done on an entire type of network components globally. Thus, for instance, the color of all metabolites in the diagram can be changed quickly in one step. In this way, the network style can be changed very fast.

The totality of all globally defined attributes regarding the visual appearance of the network components is a so-called *style sheet*. Style sheets are exchangeable between different network documents. Therefore, style sheet files can be exported from and imported into diagrams. By applying a style sheet file on a network diagram, the appearance of all components in the diagram changes immediately according to the style sheet. This, furthermore, increases the flexibility and rapid adaptability of network diagrams.

An important property of the drawing tool is, the shapes of metabolite and reaction symbols are not fixed. The user can choose between a set of primitive shapes like rectangle, ellipse, hexagon etc. and even create custom shapes. By this, the actual symbolism used in the diagrams is customizable to the users' requirements. This includes the possibility to represent the metabolic networks in SBGN (Le Novere et al. 2009) if required.

### 3.4. Levels of Detail

Although the set of network components available in the drawing tool is kept small highly detailed network diagrams can be created. For instance, Fig. 3.7 a) shows a diagram containing multiple metabolites and reactions of several pathways from the central metabolism. Beyond displaying the substrate and product relations, the network shows



activating and inhibiting effects between the components. Furthermore, the network contains cofactors like ATP and ADP.

In many situations, certain details in metabolic networks are not of interest depending on the field of application (Michal 1998). For instance, energetic cofactors and network regulation mechanisms are often not considered in metabolic flux analysis (shortly introduced in Chapter 13.3). Hence, the network visualization tool supports a context-dependent view of network diagrams without the necessity to edit the diagram.

A network diagram can be displayed in various levels of detail because the user can hide certain types of components of the network. This is exemplified by the image series in Fig. 3.7. Entire classes of diagram components can be hidden in the diagram. In Fig. 3.7 b), for instance, pathways are hidden as well as cofactors and effector edges. Reaction nodes can be faded out leading to a hypergraph representation as shown in Fig. 3.7 c). Likewise, metabolite nodes can be hidden. In this case the metabolites are represented as dots (cf. Fig. 3.7 d and e). This dot representation as well as the label of a hidden reaction and metabolite only appears if the corresponding node is still connected with at least one visible edge. Finally, when all types of network components are hidden except pathways as shown in Fig. 3.7 e) the network can be examined from a higher abstraction level.

Beyond the option to globally hide entire classes of network components, all single reactions and metabolites as well as pathways and compartments can be hidden individually by changing their “visibility” property. The possibility of representing a diagram in various levels of details as well as the option to hide single components allows to use one network diagram for multiple purposes with different and, if the case may be, contradictory requirements.

## 3.5. Joining Networks

Like other document-editing software tools, the diagram editor supports copy and paste functionality, meaning that parts of a diagram can be selected, copied to the systems clipboard and pasted into other diagrams. This feature helps to join different networks in a new diagram or to build up networks from parts (e.g. pathway by pathway). Because metabolite and reaction identifiers are unique in a network diagram possibly occurring name collisions have to be resolved after a paste operation. Here, different cases have to be taken into account depending on the type of the inserted network component:

- If a reaction already exists, the user can decide between using the existing one, renaming the pasted reaction or omitting it.
- This also applies when the name collision occurs with a metabolite. Here, inserting a further duplication is additionally offered.
- If a node is skipped, all its connections are also omitted.

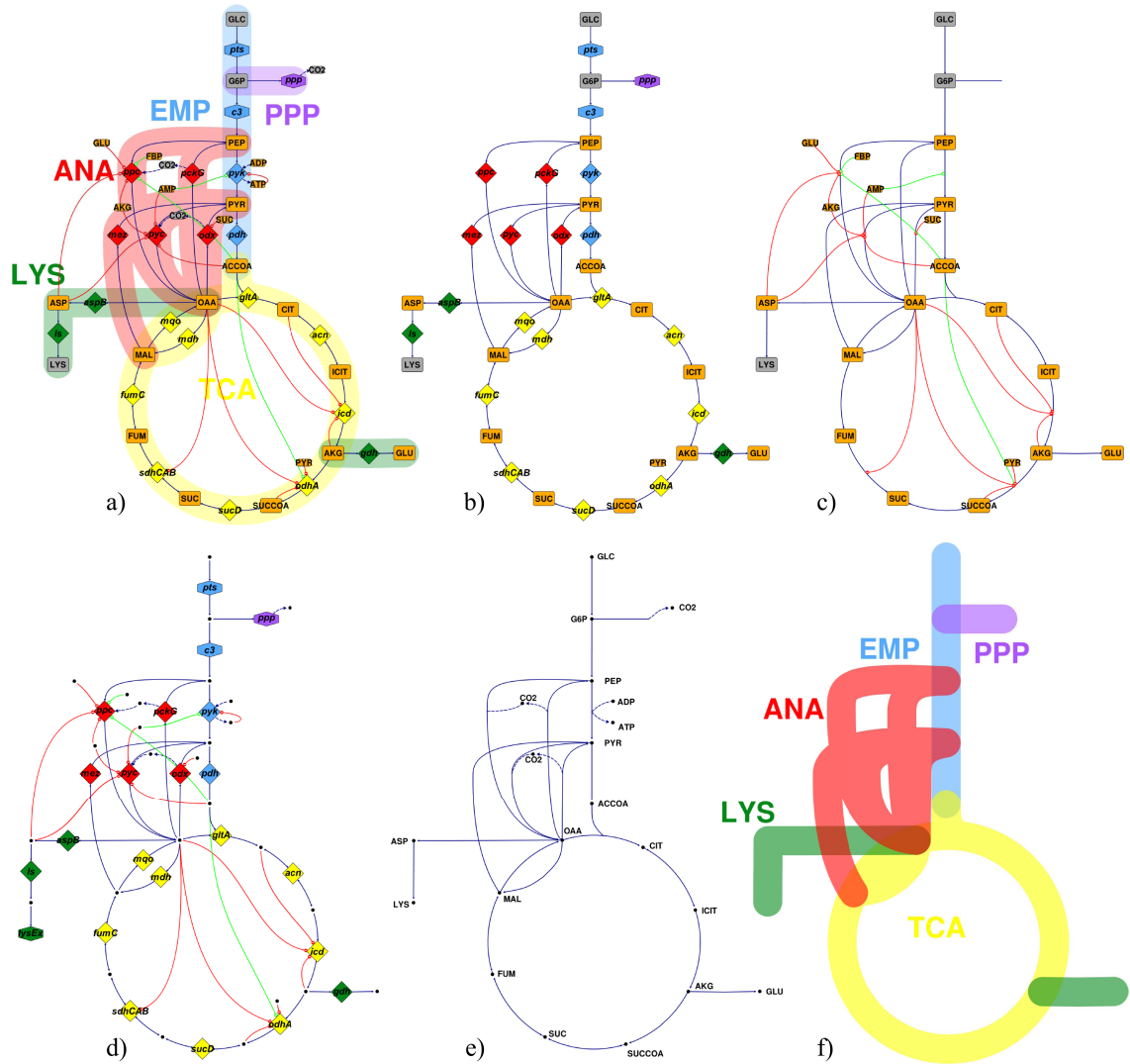


Figure 3.7.: Classes of network components can be faded out entirely. Figure a) shows a detailed diagram. In b), c), d) and e) certain network components are hidden leading to a reduced representation of the network. In figure f) everything but pathways is hidden. By this, a higher abstraction level of the network is given.

- If a pasted edge is already available between its source and destination node it is skipped automatically.

In addition to the copy paste feature, it is possible to merge (sub)networks from different documents by importing which likewise requires name collision handling.

### 3.6. Layout-less Components

In addition to the drawing area where the network diagram is edited, all components of the metabolic network are listed in the so-called *component view* being a side window of the main window as depicted in Fig. 3.8. All components of the network occur as elements in this list subclassified into reactions, metabolites, pathways and compartments. All reaction and metabolite items, furthermore, contain subitems representing the edges connected to the node in order to allow the inspection of these connections. The pathway and compartment items contain subitems representing the associated reactions and metabolites, respectively.

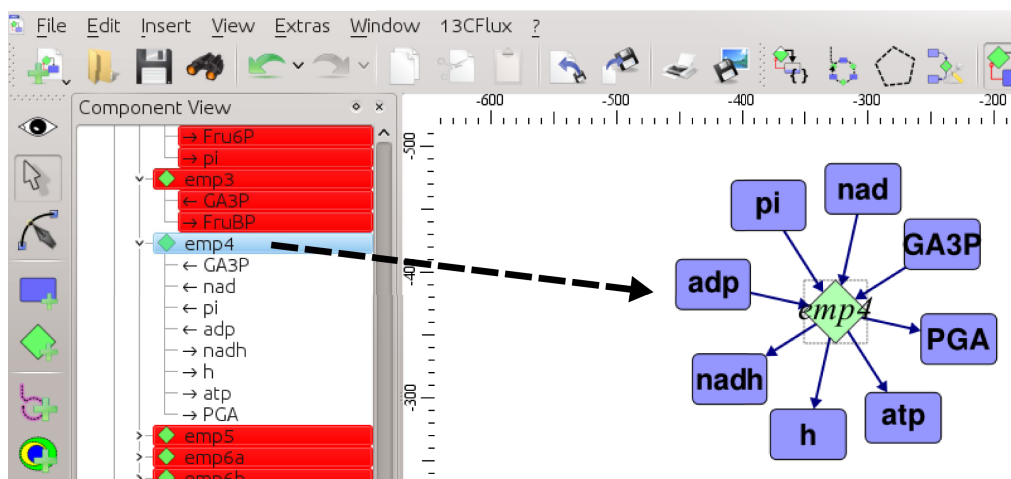


Figure 3.8.: Arrangement of layout-less networks on the drawing area.

When a network is imported from a file, database or other third party sources without any layout information provided, node positions and edge shapes are initially not available. These nodes and edges are not instantly visible in the diagram. Instead, they only appear in the component view. The corresponding node entries in the component view are labeled as to be non-positioned (cf. Fig. 3.8). The nodes represented in this way can be added to the diagram by drag-and-drop. Only nodes can be inserted in this way. Initially invisible edges are inserted automatically as soon as the source and destination nodes are added.

In order to accelerate the positioning of layout-less nodes in the diagram an option is offered when a reaction node is dropped on the drawing area. Now the user can choose whether all neighboring metabolite nodes should additionally be inserted. If this

functionality is chosen, the neighboring nodes are inserted automatically and arranged circularly around the reaction as shown in Fig. 3.8. Thereafter, they can be repositioned as preferred before further nodes are inserted. In this way, an imported network without positioning information must be arranged manually. Part II introduces a semi-automatic network layout approach that accelerates this drawing task.

### 3.7. Example Drawing Process

Finally, a case study is presented that has been performed in order to prove the stability and robustness of the drawing editor in a large-scale modeling process. Fig. 3.9 on the facing page shows a genome-scale metabolic network and a number of intermediate drawing steps. The diagram was created by stepwise importing single pathways from the KEGG databases (Kanehisa and Goto 2000) and combining them to a consistent diagram. The model consists of 826 metabolites, 846 reactions and 59 pathways. The diagram has been created by a biotechnologist who deals with modeling of metabolic networks.

The drawing process has taken eight days. Compared with automatic graph drawing this seems to be long. However, Path II will demonstrate that automatic graph drawing is not suitable to create a diagram according to historically entailed layout standards. These conventions necessarily require the human factor. Considering the costs of drawing such a diagram with a common graphics software tool eight days are acceptable.

Another aspect is, the here presented drawing process corresponds to a network modeling process that includes proofreading of the complete model against literature information. The diagram inherent network model can be used for multiple purposes for instance data visualization (cf. Part III) and simulation (cf. Part IV). Having this high reusability in mind, eight days are quite less compared with the time effort of separate drawing, modeling and visualization tasks.

### 3.7. Example Drawing Process

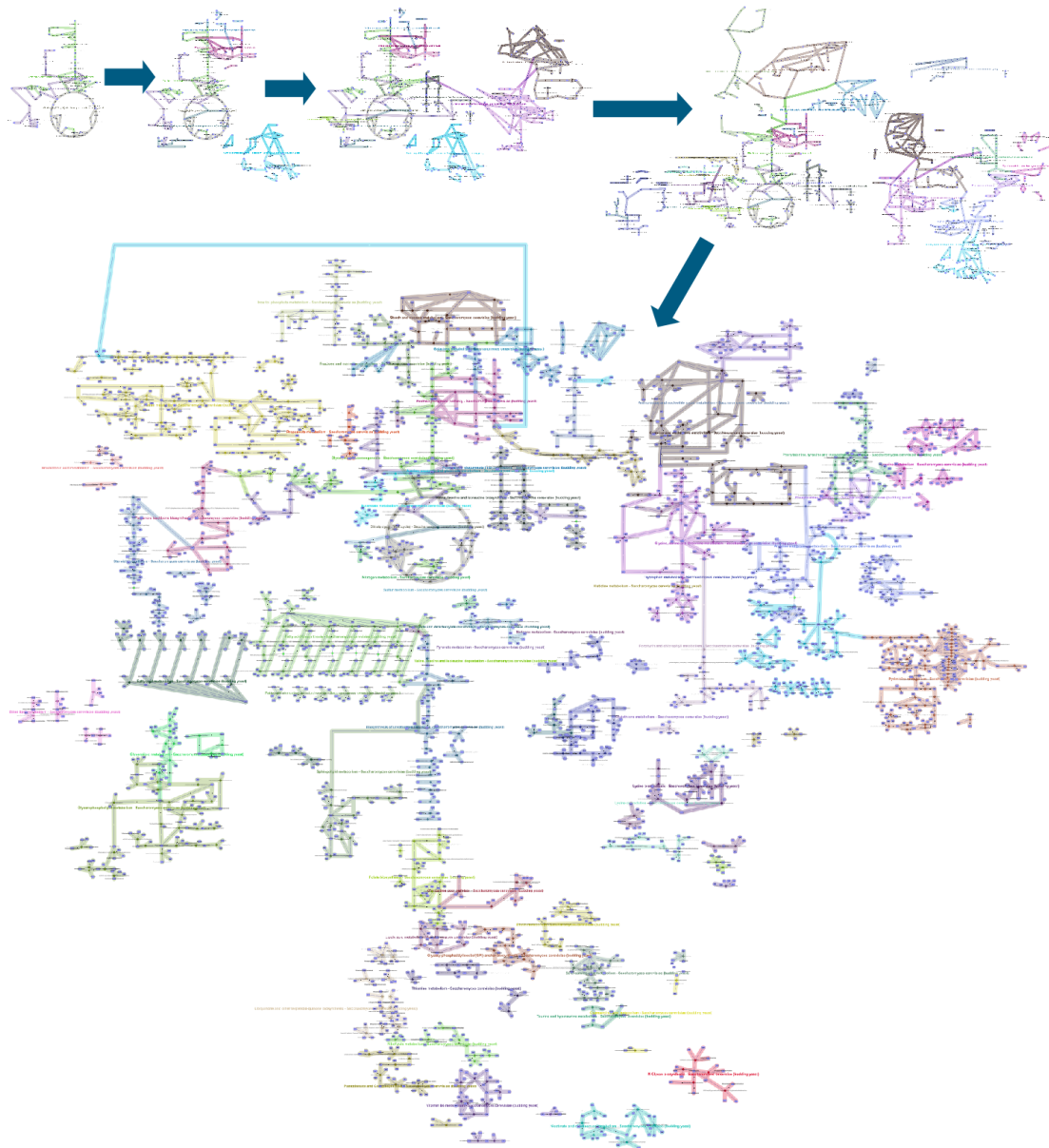


Figure 3.9.: Documentation of a drawing process.



# Omix Feature Sheet

Network Drawing Features	Page
• Sophisticated drawing features by using Qt*	25
• Publicly available for academic use	25
• Fully documented in a user manual*	25
• User-centric software design*	26
• Intuitive user guidance	26
• Simultaneously editing multiple documents	26
• Language support	26
• Diagram export to established pixel- and vector-oriented image formats	26
• Self-explanatory drawing tool	26
• Shape editing with Bézier splines	26
• Small set of biologically meaningful network symbols*	27
• Abstract representation of regulatory effects	28
• Reversibility as a flag; flexible visual representation of reversibility**	29
• One-click-reversing of a reaction direction**	29
• Inverting the visual reaction direction**	29
• Cofactor role represented by edges*	29
• Coefficient of flux edges	30
• Duplicated metabolites	30
• Duplication of a metabolite once per edge**	31
• Visual representation of metabolic pathways**	32
• Compartments and compartmentalized metabolite pools*	32
• Custom nodes and edges as optional semantic-free components of a network*	33
• Highly customizable network style*	33
• Individual and global change of item properties*	34
• Style sheets exchangeable between different documents	34
• Flexible network symbol shapes	34
• Displaying the diagram in different levels of detail*	34
• Copy-paste of network parts	35
• Joining multiple networks	35
• Importing networks	35
• Alternative model representation	37
• Representation and handling of layout-less components*	37
• Automatically inserting all substrates and products of a reaction*	37
• High reusability of network drawings*	38

\*particular feature of *Omix*, i.e. not novel in general but hardly realized by other tools for visualization in the context of biochemical networks.

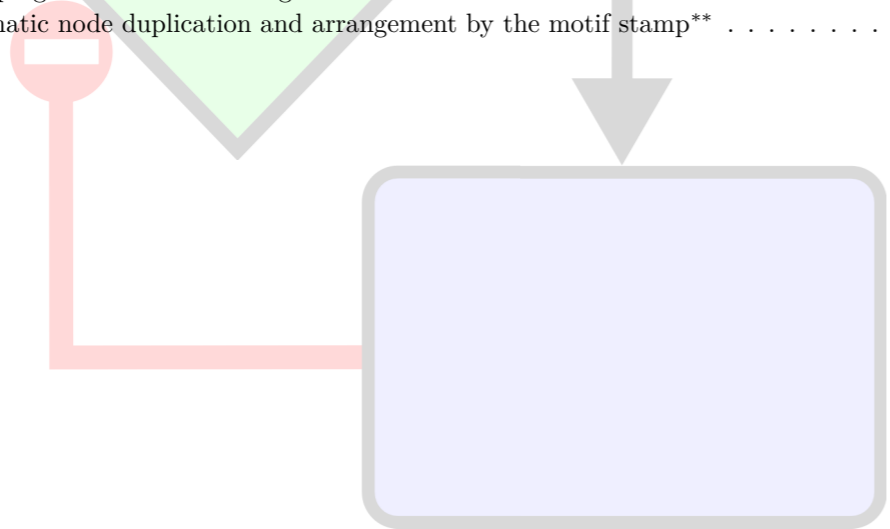
\*\*novel approach introduced in this thesis.

# Omix Feature Sheet

## Network Layout Features

Page

- The *layout pattern*, a skeleton for the arrangement of pathways in the diagram\*\* . . . 51
- Reshaping of edges according to the pattern section\*\* . . . . . 51
- Proper circular arrangement of the citric acid cycle pathway\*\* . . . . . 53
- Equidistant placement of nodes on an arbitrary shaped pattern section\*\* . . . . . 53
- Exchanging the layout pattern figure between different documents . . . . . 51
- Successive arrangement of metabolic pathways . . . . . 53
- Semi-automatic network path search\*\* . . . . . 54
- Pre-selection and exclusion of network paths for a fast overview\*\* . . . . . 55
- Pathway-oriented network layout\* . . . . . 56
- The *motif stamps* wrap multiply occurring drawing steps into one action\*\* . . . . . 57
- Capturing motif stamps from existing reactions\*\* . . . . . 59
- Exchanging motif stamps between different documents . . . . . 59
- Rotating, scaling and inverting of the motif for an adapted arrangement\*\* . . . . . 59
- “Snapping” to a smooth arrangement\*\* . . . . . 59
- Automatic node duplication and arrangement by the motif stamp\*\* . . . . . 60



\*particular feature of *Omix*, i.e. not novel in general but hardly realized by other tools for visualization in the context of biochemical networks.

\*\*novel approach introduced in this thesis.



**Part II.**

**Network Layout**



# Chapter 4.

## Introduction to Network Layout

*“Despite the wealth of existing algorithms, the layout problem still remains one of the crucial bottlenecks in network visualization.”*

(Pavlopoulos et al. 2008)

### 4.1. Layout Conventions

As introduced in Chapter 1\*, there are several different types of biochemical networks from the genome, transcriptome or proteome layer displaying different aspects of intracellular processes. Each of these layers has its own, very specific peculiarities. In the context of metabolic networks, we are concerned with historically established conventions for network layout. Biologists have these “familiar” layouts in mind when they draw networks inspired by popular biochemical text books (Stryer 1995, Michal 1999, Madigan et al. 2003). These textbooks share *de facto* layout rules which are commonly accepted although not casted into formal layout definitions, but rather informal layout rules. These rules deal with the arrangement and shape of metabolic pathways in a diagram as well as the design of single reactions. Fig. 4.1 shows a few examples for metabolic network drawings that can be found by searching for the term “metabolic network” in the Internet. This random selection demonstrates the existence of a *de facto* standard for network layout.

When simulated and experimental data has to be visualized in a network context, the established layout conventions should be obeyed in order to increase the acceptance and readability. Following established layout conventions facilitates rapid orientation. The human observer can concentrate on the essential quantitative or structural information related to the network. This makes discussion of, for instance, measurement values visualized in a metabolic network diagram much more efficient.

### 4.2. Automatic Versus Manual Approaches

As an illustrative example, Fig. 4.2 on page 47 shows a small network of microbial central metabolism. In Fig. 4.2 a) the network was manually drawn according to the

---

\*Part II has in altered form been published in (Droste et al. 2011b).

## Chapter 4. Introduction to Network Layout

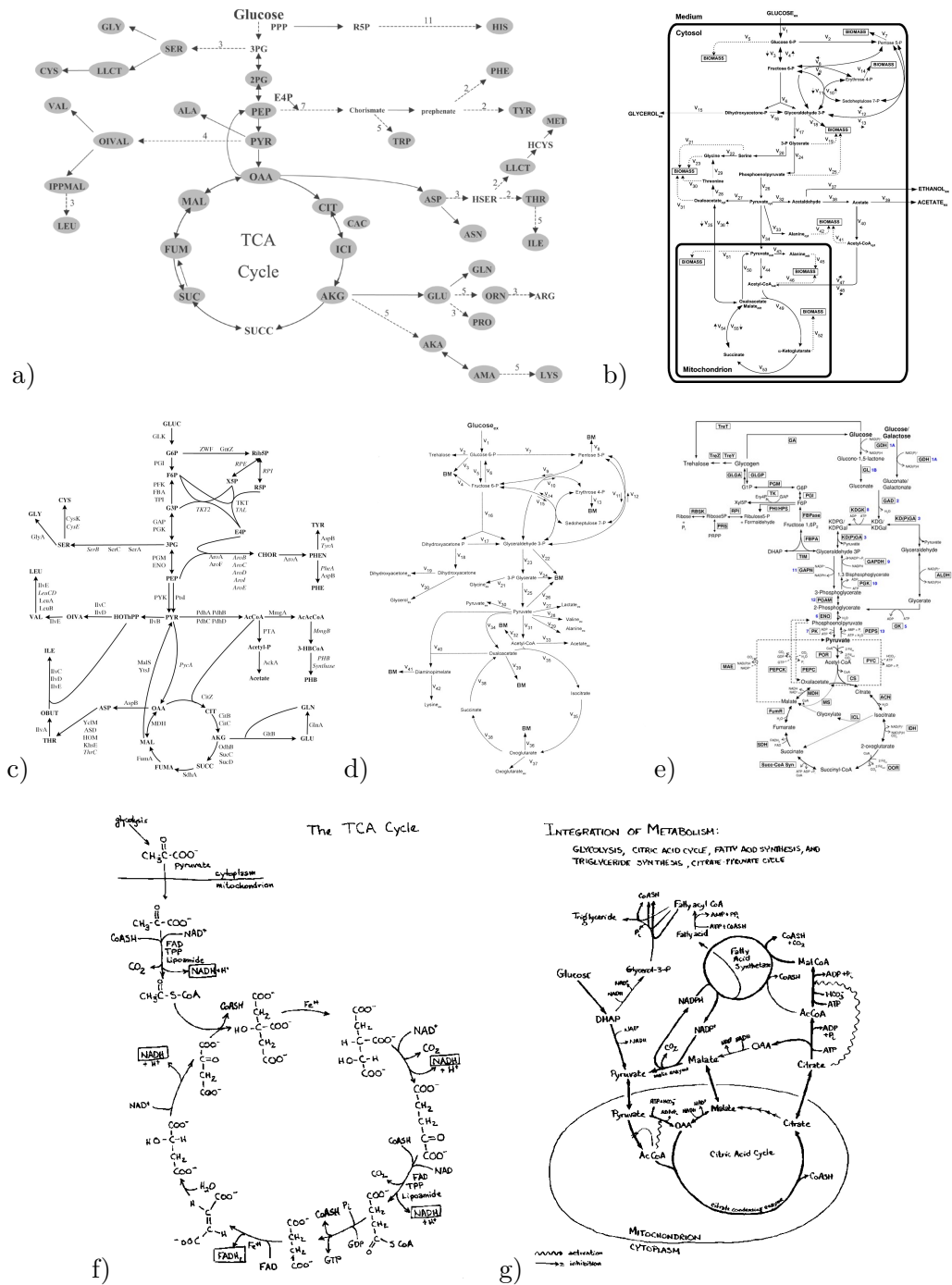


Figure 4.1.: Examples for metabolic networks from literature and Internet: (a) taken from Villas-Bôas et al. 2005, (b) Frick and Wittmann 2005, (c) Wang et al. 2005, (d) Kiefer et al. 2004, (e) Zaparty 2010, (f) and (g) hand-drawn network diagrams taken from White 2001.

## 4.2. Automatic Versus Manual Approaches

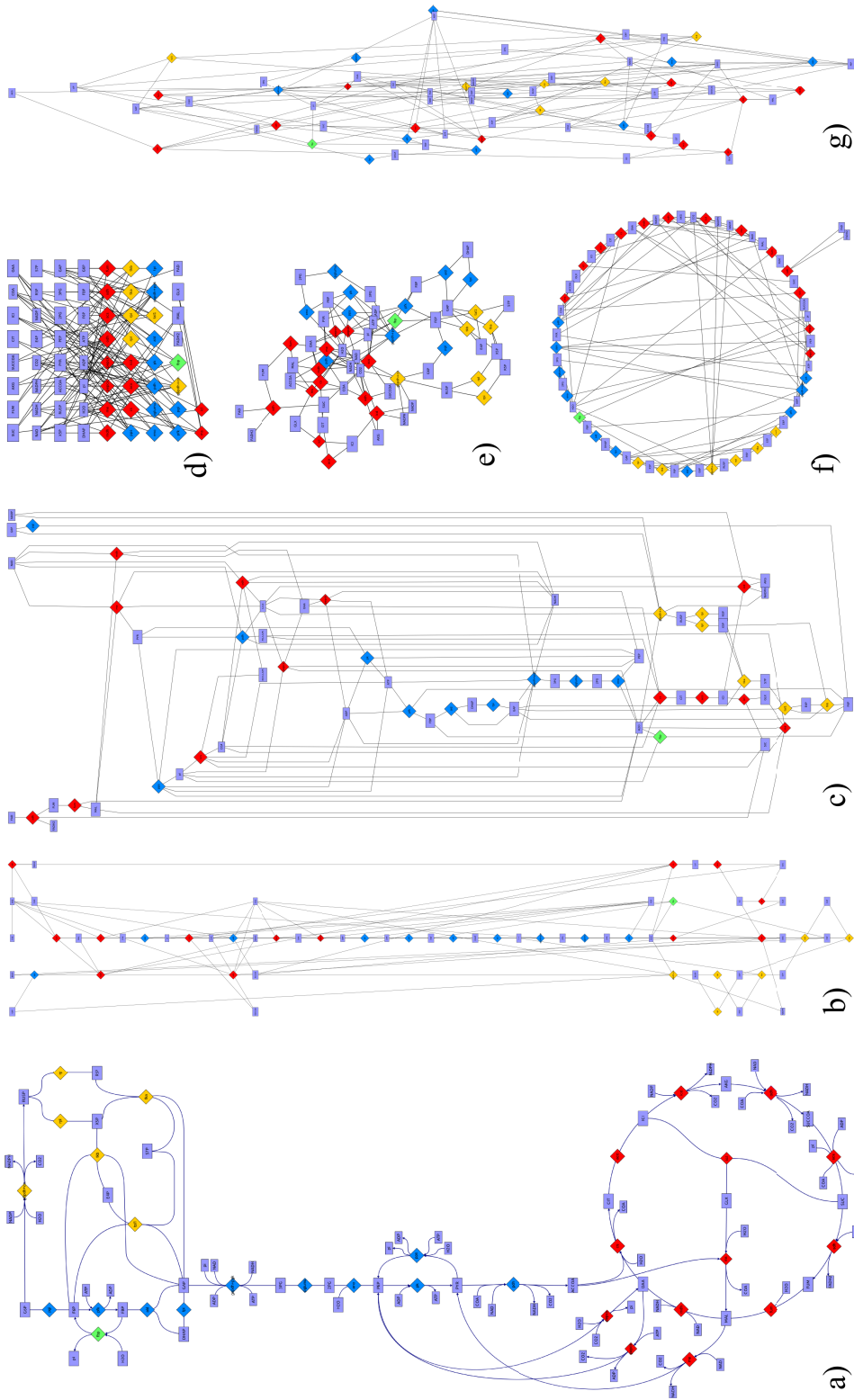


Figure 4.2.: Layout of the central metabolic pathways: a) manually drawn according to well-established layout conventions; b) to g) automatic computed graph layouts with several algorithms provided by *CytoScape*.

“standard” text book layout. Fig. 4.2 b) to g) display the same network but are designed by means of six different automatic layout algorithms performed by *CytoScape* (Shannon et al. 2003): Sugiyama algorithm (b), hierarchical network layout (c), grid layout (d), force-directed placement (e), circle layout (f) and tree layout (g).

Even an untrained biologist is immediately able to identify all the essential metabolic pathways shown in Fig. 4.2 a). It is quite obvious, however, that the automatic computed diagrams are hard to read because even the prominent paths and cycles cannot be recognized in the automatically computed arrangement (see Fig. 4.2 b to g). These diagrams are inadequate for information visualization where orientation and fast identifiability is an indispensable prerequisite as for metabolic networks.

As stated by Suderman and Hallett (2007), most tools in the context of biochemical network visualization support a fully automatic network drawing mode. Here, the usual approaches of automatic graph drawing, as for instance hierarchical (Sugiyama et al. 1981) or force-directed (Eades 1984, Fruchterman and Reingold 1991) methods are adapted and improved according to the requirements of the biochemical application field by, for example, clustering or grid-based approaches (Li and Kurata 2005, Bourqui et al. 2006, Rohrschneider et al. 2009). However, Saraiya et al. (2005) wrote that the utility and impact of those approaches for visualization of metabolic networks is yet unclear. Even these adapted automatic solutions rarely lead to well-accepted network layouts. For instance, even the KEGG reaction database (Kanehisa and Goto 2000) uses manually drawn diagrams for the visual representation of metabolic pathways.

This thesis does not argue against automatic graph drawing in general. Certainly, these techniques are successfully applied in other types of biochemical networks, for instance, protein-protein interaction networks (Schwikowski et al. 2000). In such application fields the arrangement of nodes and edges in the diagram has no effect on the identifiability of single components because there are no standardized layout rules. Hence, an automatic layout of network components does not affect the gain of insight from the visualization.

Automatic graph drawing is feasible if no commonly accepted layout conventions are available, or, graphs are generated for getting a qualitative impression by means of connection-centric approaches. Furthermore, automatic network layout is undoubtedly valuable where the size of the networks exceeds several ten thousands of nodes even in the context of metabolic networks (Batagelj and Mrvar 2002, Shannon et al. 2003). Gehlenborg et al. (2010) state “Although these specialized automated layout methods are useful, they are usually of low quality compared to manually laid out pathways created by human experts and often require manual editing in addition”. Human intervention is necessary in order to adapt the automatically computed layout to the requirements of the scientific application field (Villéger et al. 2010). This work focus on smaller networks ranging from one hundred up to two thousand nodes.

## 4.3. Circular Arrangements

Prevailing automatic graph layout algorithms are not able to arrange cycles as they are commonly drawn in metabolic networks. From a graph theoretical point of view, metabolic networks usually contain several potentially nested cycles as depicted in Fig. 4.3. Here, a subsection of a network from central metabolism is shown with four exemplary graph cycles highlighted in the network. Not all graph cycles, however, are actually drawn as circles in a network diagram by convention. In the example from the central metabolism, only the citric acid cycle (cf. Fig. 4.3 d) is arranged in a circle as it was initiated by the pioneers of molecular biology. Obviously, graph analysis algorithms are not able to identify the one network path conventionally drawn as circle.

A frequently applied automatic layout scheme is the arrangement of all network nodes on a circumference interconnected by straight line edges (Breitkreutz et al. 2003) as shown in Fig. 4.2 f). However, a circular arrangement of cyclic network paths as part of a hierarchical network layout has not yet been applied successfully.

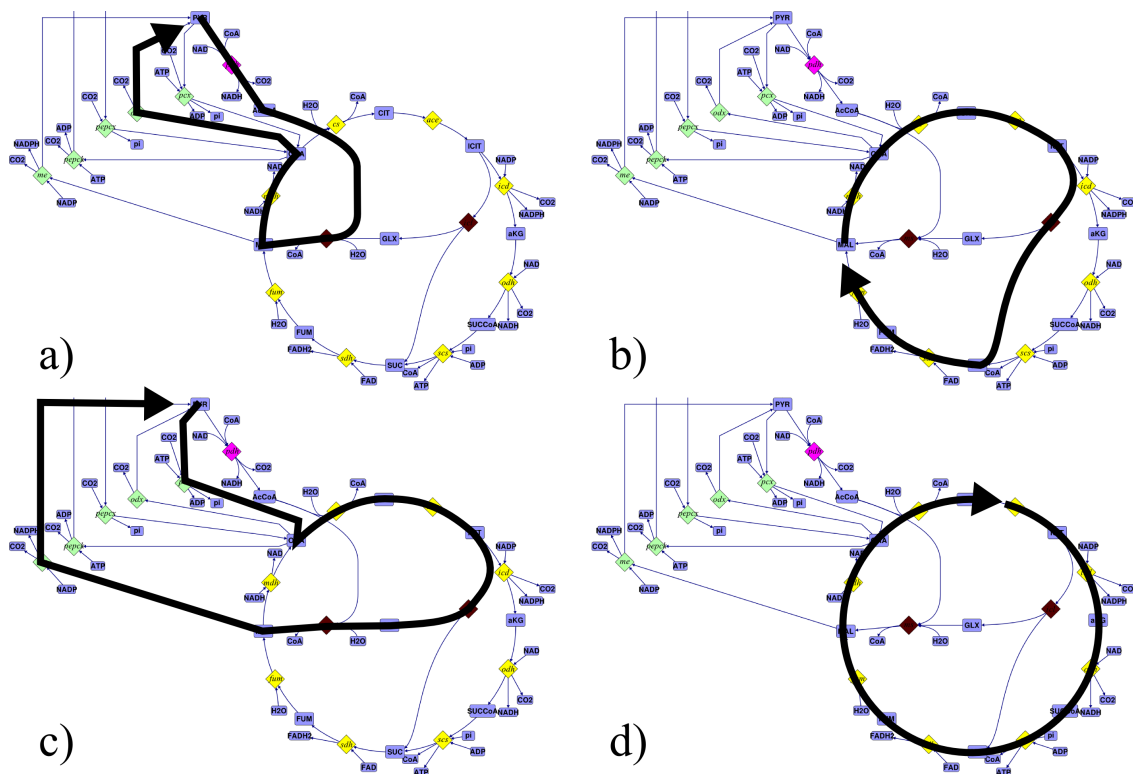


Figure 4.3.: Examples for cyclic paths in a subnetwork from central metabolism.

## 4.4. Layout Requirements of Metabolic Networks

In the context of data visualization the design process of metabolic network diagrams is governed by conserving *recognition value* which means rapid orientation in the diagram and immediate identifiability of “familiar” network sections. Recognition value is very important in order to generate network-integrated data visualizations giving fast impressions of the embedded information.

Another closely related aspect in drawing metabolic networks is *aesthetics*. Here aesthetics does not only and not necessarily include a minimized number of edge crossings and bends, symmetry, and maximal angles between edges leaving a node (Purchase 1997). Here, a more general concept of the term *aesthetics* is propagated including the shape of edges and proportions of nodes. Fig. 4.2 a) shows differently formed edge shapes leading to an aesthetically pleasing diagram. Here, edges between circular arranged nodes are shaped as arcs. Other edges have linear or curved characteristics adapted to the arrangement of nodes. The shape of edges is a significant aesthetic criterion for metabolic network diagrams.

A third aspect that becomes increasingly important for larger networks is *efficiency*. Drawing networks is very time-consuming. Saving time by supporting the human activities is a necessary capability of a drawing software for metabolic networks. The efficiency aspect includes an intuitive user guidance as well as the option of automating repetitive actions.

These three requirements – recognition value, aesthetics and efficiency – must be supported by a network drawing tool. It is nearly impossible to perform all these requirements for metabolic networks with automatic layout algorithms. Clearly, the automatic placement of network elements is time-efficient but the designed network diagrams do neither satisfy common aesthetics nor have any recognition value (see Fig. 4.2 on page 47). In the context of metabolic networks, aesthetics and recognition value can only be sufficiently preserved by the human factor but manual graph drawing is time-intensive.

Here, a novel semi-automatic graph drawing approach is introduced that aims at assisting the manual drawing process by a small set of automatisms leading to satisfying network drawings and acceptable time factors. This approach consists of the so-called *layout pattern*, which is described in Section 5, and the *motif stamps* introduced in Section 6. In order to give an impression of the strength of the contribution Section 7 discusses a work flow for applying the design and layout techniques presented here “in real life”. In Appendix B a user case study is given evaluating the effectiveness of the here presented semi-automatic layout approach against the requirements recognition value, aesthetics and efficiency.



## Chapter 5.

# The Layout Pattern: Shaping the Coarse Network Structure

The drawing area of the diagram editor can be pre-structured with a so-called *layout pattern*. A layout pattern is a skeleton on which series of nodes and edges can be arranged (see Fig. 5.1). In the process of arranging nodes on the drawing area, the layout pattern supports the user in finding the correct node sequences in a heterogeneously connected graph. To a certain extent, the layout pattern is a meta-component of the diagram imprinting an overall structure to the network. In the following, this novel method of semi-automatic network layout is introduced in more detail. The usage of the layout pattern for the arrangement of nodes and edges is illustrated and the challenges arising from the combination of interaction and automation are stressed.

The layout pattern is an arbitrarily complex figure consisting of several spline segments, the so-called *pattern sections*, which can be versatily assembled. Fig. 5.1 a) shows a pattern figure for the central metabolism. Every pattern section is an arbitrarily shapeable spline between two branch points as illustrated in Fig. 5.1 b). On a pattern section a sequence of nodes and edges is arranged according to the geometry of the path curve (see Fig. 5.1 c). When the shape of a pattern section is subsequently changed the position of all its nodes and the shape of all the edges are immediately adapted to the new shape of the pattern. A node on a pattern section can have two neighbors; a node on a branch point can have as many neighbors as pattern sections are connected to the point.

Pattern sections can be added to the drawing area. Here, user guidance is oriented to the Bézier spline handling of common graphics software tools. Curvature can be edited with control points, smoothness can be enforced and curve sections can be joined or split. In this way, the diagram layout can be prepared. Furthermore, the layout pattern figure is exchangeable between different network documents by an import/export functionality.

Network editing and layout pattern editing work in two different layers. When the pattern is edited, it becomes a top level item whereas all components of the network diagram are inaccessible and semi-transparently displayed in the background (cf. Fig. 5.1 b). Vice versa, the layout pattern is semi-transparent in the background in network editing mode (Fig. 5.1 c). When the diagram is used for visualization, printout etc. the layout pattern is invisible.

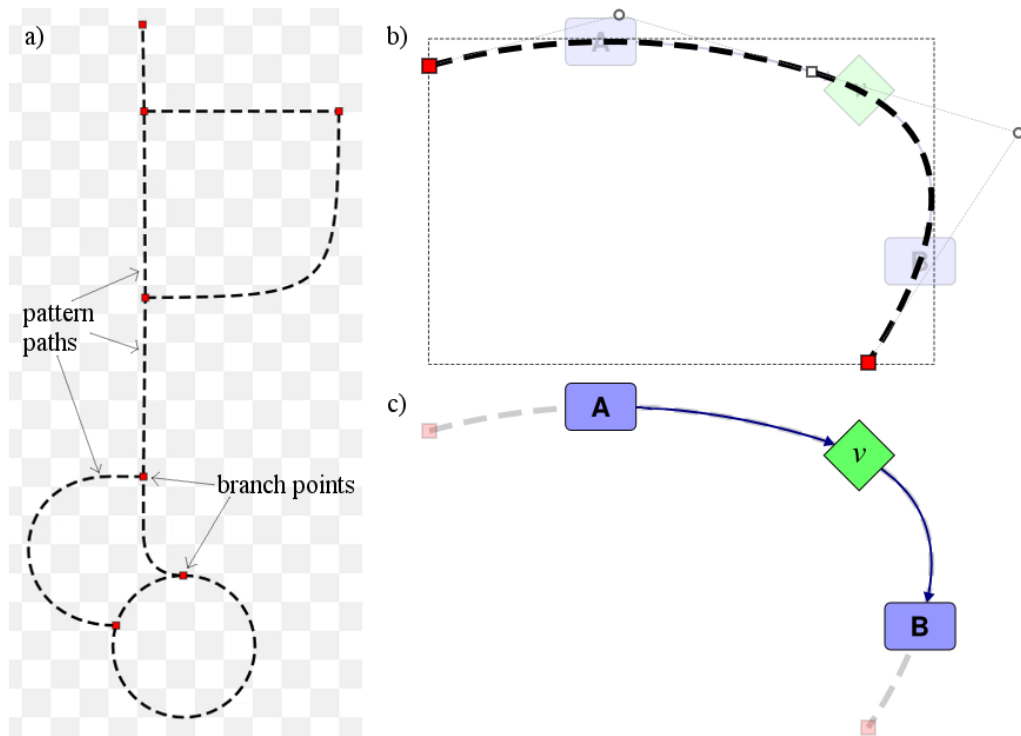


Figure 5.1.: Layout pattern consisting of several pattern sections connected by branch points (a). Pattern sections formed as a sequence of Bézier curves (b). Arrangement of nodes and edges on a pattern section (c).

## 5.1. Using the Layout Pattern

This subsection gives an impression of the work flow of using the layout pattern. As mentioned above, nodes and edges can be placed on the layout pattern. This is exemplified in the image series in Fig. 5.2 on the next page where a chaotic network is being arranged by a user-defined pattern section (see Fig. 5.2 a). Using the layout pattern is done by positioning nodes either above a pattern section or a branch point. In Fig. 5.2 b) a reaction node has been moved onto a position on a circular pattern section. When a node  $v_0 \in V$  is added to the layout pattern in this way, the layout pattern searches for paths in the network that interconnect the added node  $v_0$  to its neighboring nodes  $v_1 \dots v_n$ ,  $n > 1$ . The path search requires human interaction because there can be many paths between two nodes in a network. The challenges and requirements of this path search are discussed below.

Since the pattern section in the example shown in Fig. 5.2 is a circle, a cyclic path must be specified by the user to be arranged on the pattern. This path starts and ends with the reaction node recently added to the layout pattern. After selecting the preferred path, its

## 5.2. Challenges and Requirements

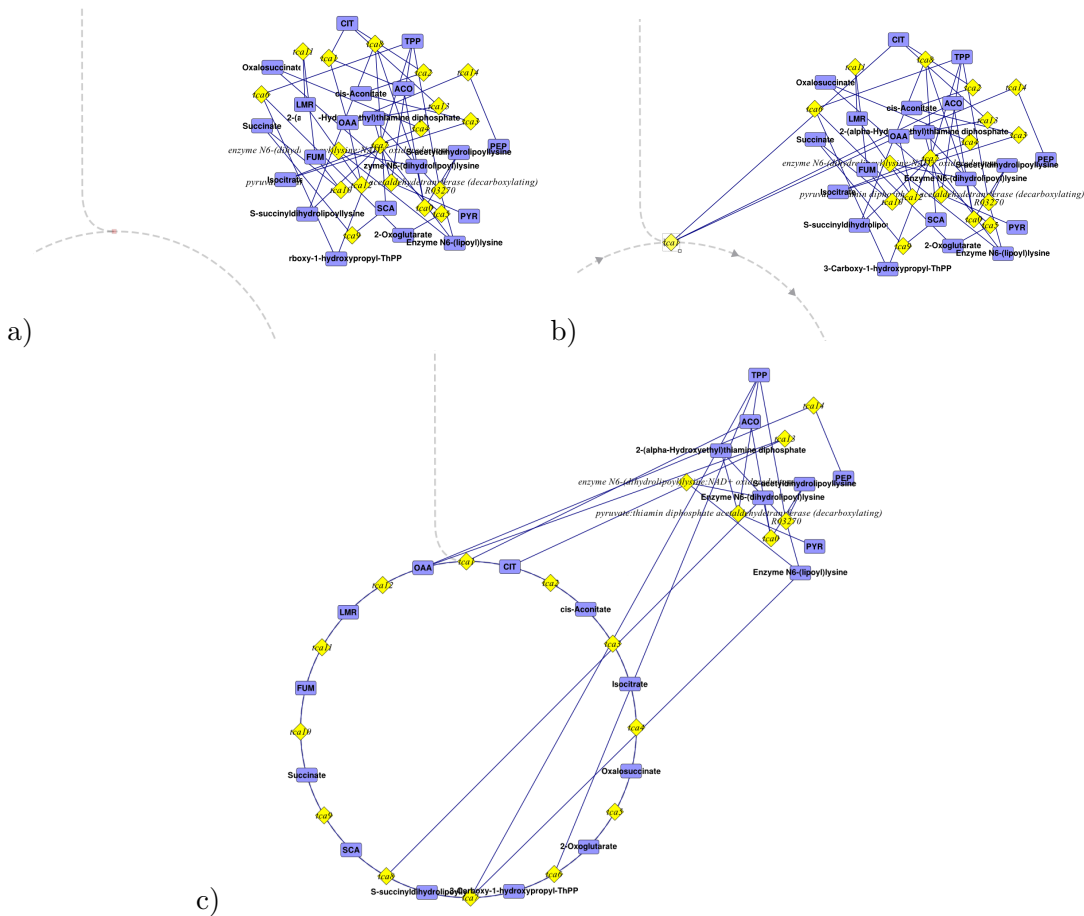


Figure 5.2.: Using the layout pattern to design a sequence of nodes and edges: (a) initial situation, (b) starting the layout step by moving a node above the pattern, (c) finished layout step after interactive search of matched node sequence.

node sequence is finally added to the pattern by arranging the nodes equidistantly and reshaping the edges according to the form of the underlying pattern section as depicted by Fig. 5.2 c).

Subsequently, further pattern paths can be inserted for the placement of the other node sequences in the network. Thus, the layout pattern can be used to successively arrange metabolic pathways in the diagram.

## 5.2. Challenges and Requirements

Human interaction is required to decide which network path is to be arranged between two nodes  $v_0, v_n$  in the network positioned on the layout pattern. The user has to select

the preferred path out of all possible network paths. A prior path search on the graph is necessary in order to compute all possibilities. Here, the following conditions have to be taken into account:

1. The directed metabolic network has to be regarded as an undirected graph because the pattern sections are non-directional. This convention enables the user to arrange sequences of opposed edges on the layout pattern.
2. A node may not occur twice in a network path except for it is a start as well as an end node on a circular pattern section.
3. Only those nodes that are not already positioned on a pattern section may occur in a network path.

The number of network paths between two nodes is  $O(n!)$  with  $n = |V|$  in the worst case of a complete graph. Considering a maximal number  $k$  of edges connected to a node, the complexity is diminished to  $O(n^k)$  which nevertheless leads to two non-trivial problems:

1. Calculation of all possible paths with standard search algorithms like depth or breadth search leads to computation times which are unacceptable for human interaction even for small networks.
2. Selecting the preferred path from an exhaustive list of all possible ones typically requires distinguishing between a huge number of candidates. This is nearly impossible even for experienced users.

Due to both of these reasons, fully automatic network analysis is unsuitable. Constraining the analysis, for example, by a limited depth search certainly reduces the computation time. However, a limited depth search leads to an incomplete set of result paths. Due to these runtime and completeness problems, a novel concept of network path search that assists the user in selecting the preferred path in an efficient and interactive way is introduced here.

### 5.3. Semi-automatic Path Search

In order to deal with the mentioned runtime and completeness problem, the node path search of the layout pattern is implemented in a semi-automatic manner. Here, the core idea is a user-performed manual depth search which is software-supported by a pre-viewing and pre-selecting search algorithm.

#### 5.3.1. Human-Interactive Path Search

The human interaction interface is a tree view (see Fig. 5.3 on the facing page) representing the graph in the form of a tree whose root element is the node  $v_0$  currently moved above the layout pattern. All nodes connected to the node  $v_0$  are child elements of the node's

### 5.3. Semi-automatic Path Search

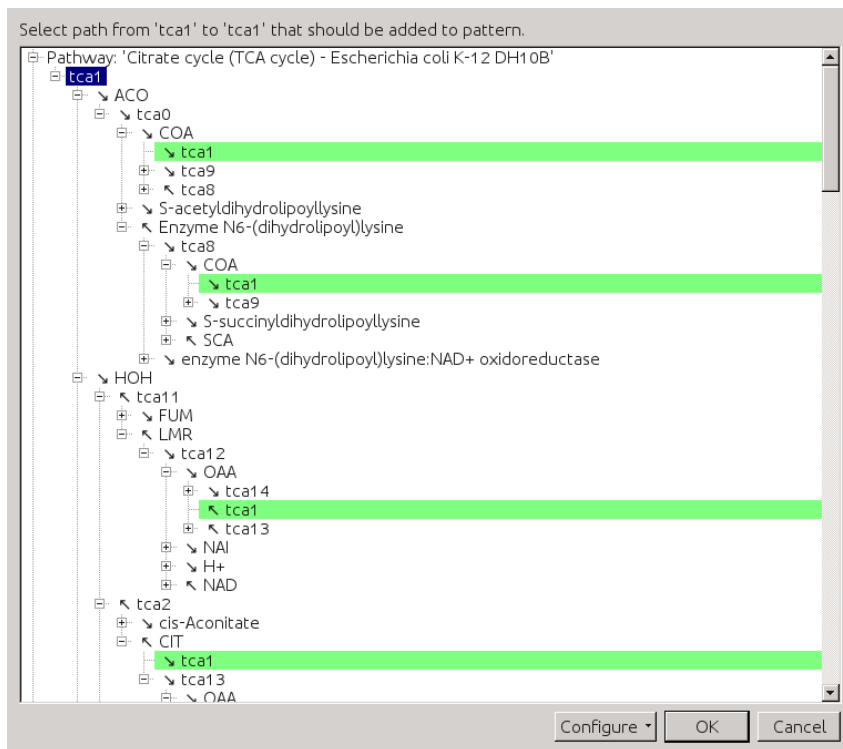


Figure 5.3.: Dialog window of the interactive node path search. In the scenario shown, the user is asked to select a cyclic node path containing the node “tca1”. Initially, the internal search algorithm has already found several valid node paths. These paths are expanded and highlighted in the tree view. The arrow symbols indicate the direction of the interconnecting edges.

tree element. The direction and type of an edge between two nodes is displayed by an arrow symbol preceding the node name as shown in Fig. 5.3.

Initially, only the root element and its first nesting level elements are visible. The user has to traverse the tree by expanding its elements until the target node  $v_n$  occurs as a leaf element in the tree. A leaf element containing  $v_n$  represents a valid path which can be added to the layout pattern. For a fast perception, the target leaf elements are highlighted in green as pictured in Fig. 5.3. By selecting a leaf element, the corresponding node path is designated for automatic arrangement on the layout pattern.

#### 5.3.2. Software-Assisted Pre-Selection and Exclusion of Network Paths

When the tree is initialized and, likewise, every time the user expands a new tree element a search algorithm traverses the graph. Triggered by expanding a tree element, the algorithm is initialized with the currently expanded element. Thus, the software always

pre-caches the network topology several nesting levels in advance of the user's manual search activity. Here, a depth-limited search algorithm is applied with an initial recursion limit  $d = 15$  whereas the limit is decreased in each recursion step depending on the connectivity  $|E(v_i)|$  of the currently traversed node  $v_i$ :

$$d_{next} = d_{current} - |E(v_i)| \text{ with } v_i \in V \quad (5.1)$$

In particular, in the presence of highly connected nodes the context-dependent adjustment of recursion depth avoids the impairment of the human-computer interaction.

The software search is conducive to the simplification and acceleration of the user search. A valid network path ends with the target node  $v_n$ . If  $v_n$  is found during the search procedure the complete branch path leading to the node is automatically expanded. The user can inspect and optionally select the complete path immediately.

When a node occurs twice in a network path it is invalid. If the depth search algorithm detects an invalid path the corresponding branch is eliminated from the tree. If an eliminated branch was the only child element of its parent, the parent element is also eliminated. This continues until a parent element has at least one child element left. If the depth search does not lead to an assertion about the validity of a path, the corresponding tree element remains unexpanded in the tree and may be expanded manually by the user on demand.

Eliminating elements representing invalid paths prevents the user from traversing dead branches and from selecting paths which contain loops. If no path exists between two nodes in the network, the software search will successively traverse the complete graph and eliminate all elements in the tree view. In this case, no further nodes and edges are arranged on the layout pattern.

In case of short network paths between the nodes  $v_0$  and  $v_n$ , the computer-performed search can find the preferred path immediately during the initial traversing and offer it for user selection. Furthermore, the user can successively trigger the complete analysis of the network leading to the elimination of all invalid paths in the tree view.

### 5.3.3. Pathway-Oriented Layout

As mentioned in Section 3.2.2, metabolic pathways lead to a biologically motivated break down of the main graph  $G$  into smaller subgraphs  $G_P$ . The classification of reaction sequences into metabolic pathways provides important meta-information about the network structure. It can help to compute network layouts because these pathway subgraphs consist usually, but not exclusively, of linear or circular node sequences surrounded by short branches.

If both nodes  $v_0$  and  $v_n$  added to the layout pattern are part of the same pathway subgraph  $G_P$ , the interactive node path search is optionally performed on  $G_P$  in addition to the search on the complete graph  $G$ . This, additionally, accelerates the interactive path search process.

## Chapter 6.

# Motif Stamps: Cloning Small Reaction Patterns

Whereas the layout pattern concept helps to roughly arrange network diagrams, the second novel technique of the semi-automatic graph layout introduced here is concerned with a more detailed view: the so-called *motif stamps* are intended to speed up the drawing of similar and multiply occurring compositions in a network diagram. The term “motif stamp” is derived from similar concepts in pixel-oriented graphics software tools.

The concept of motif stamps aims at simplifying similar actions during the drawing process of a metabolic network. A motif, in this sense, is a template periphery  $\Pi_r$  (see Section 3.2.2) surrounding a pseudo-reaction which can be appended (“stamped”) to other reactions in the network diagram. If a motif is appended to a reaction, all metabolites and edges are inserted according to the motif.

The concept of motif stamps is motivated by the observation that certain metabolic reactions are similarly composed and connected metabolites and edges are frequently arranged in a similar manner. Fig. 6.1 shows a subsection of a large network diagram. The cofactors ATP and ADP are involved in many reactions all over the network as highlighted in the figure. Many other similar cofactor pairs are frequently involved all over a metabolic network like NADP and NADPH, FAD and FADH etc. (Becker et al. 2006). Other frequently occurring reaction partners are e.g. H<sub>2</sub>O, CO<sub>2</sub> and ions. It is a well-established practice to display cofactors by one node duplication per occurrence with similar edge shapes (cf. Section 3.2.1). This leads to many similar actions the user has to make during the drawing process, which includes duplicating and arranging nodes, inserting edges, adapting the shape and visual properties of edges and so on.

Fig. 6.2 shows a schematic diagram of the motif stamp concept. Metabolites and edges are arranged around a pseudo-reaction composing a motif stamp as shown in Fig. 6.2 a). The pseudo-reaction is not part of the drawing but template for any reaction, the motif is later appended to. By appending the motif to a reaction (Fig. 6.2 b) the metabolites and edges are arranged around the reaction node as defined in the motif stamp (see Fig. 6.2 c).

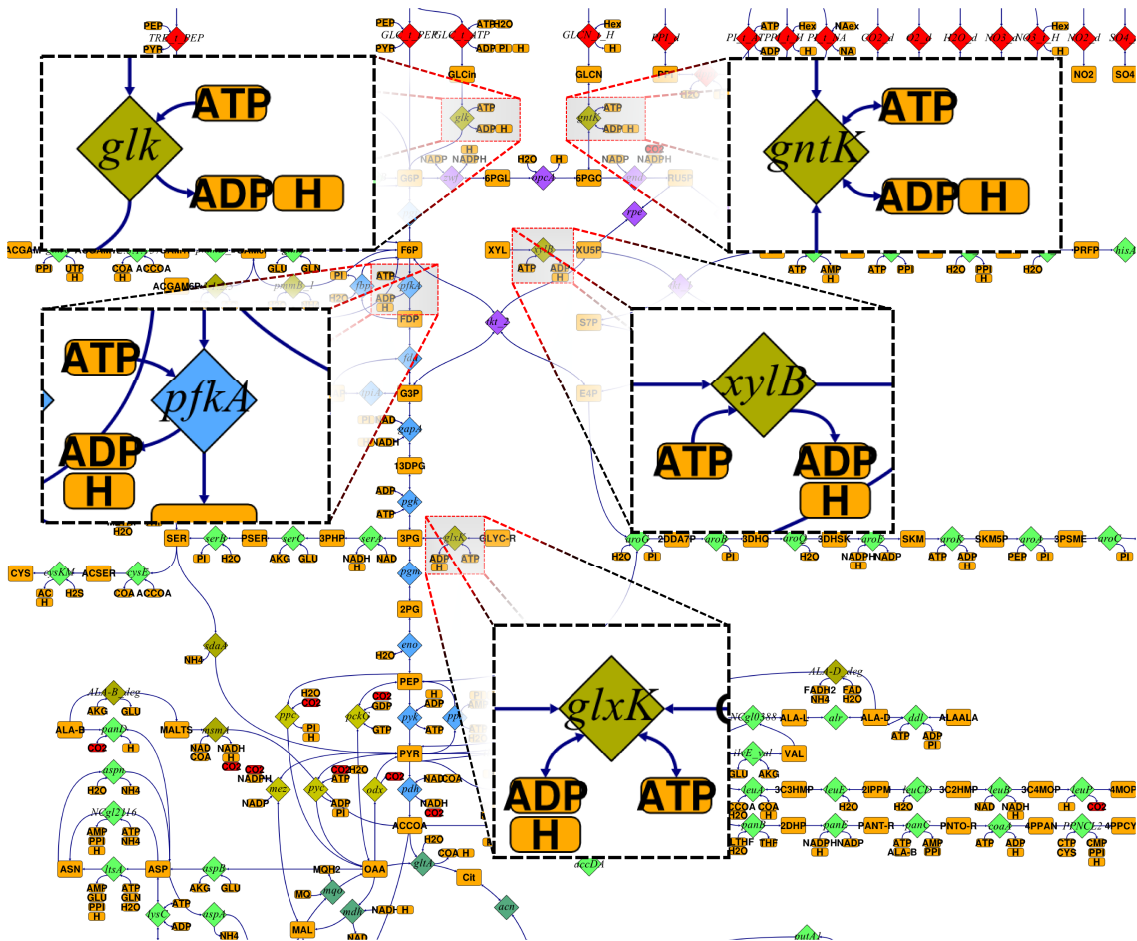


Figure 6.1.: Multiple occurring reaction partners ADP, ATP and H.

## 6.1. Creating Motif Stamps

The motif can be designed according to individual aesthetic aspects. A motif stamp defines the connected metabolites of a reaction, their position relative to the reaction node and the shape of the connecting edges. Furthermore, visual properties like color, bounds or text font of the nodes and edges are also inherent in a motif stamp.

Motif stamps can basically be created in two ways:

1. The motif can be drawn step by step. Here, metabolites and edges are created and connected to the pseudo-reaction node. Any number of nodes can be inserted but there should be at least one edge between every node and the pseudo-reaction. Basically, not only cofactor edges can be part of the motif stamp but also normal flux edges and even effector edges.



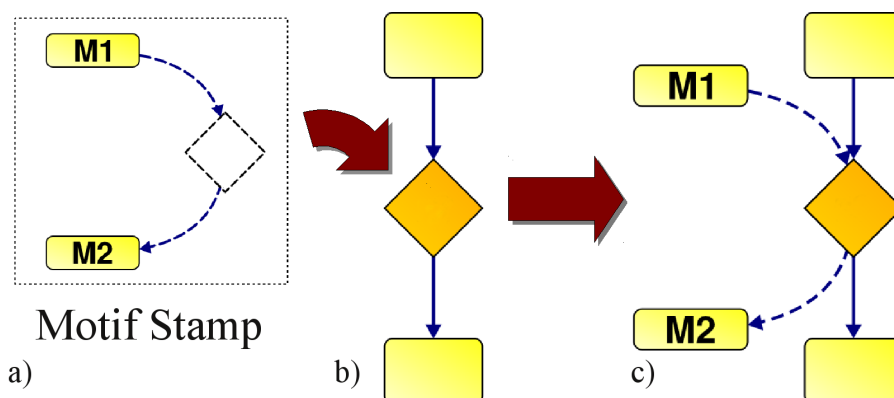


Figure 6.2.: Schematic diagram of the use of motif stamps in a drawing process.

2. The composition of a motif can be captured from the periphery of an existing reaction in the network diagram. After selecting a reaction as a template for a new motif, the user has to specify which connected metabolites should be part of the motif stamp and which are dispensable components.

After naming the motif stamp, the definition process is finished. In this way, multiple motif stamps can be defined containing certain constellations and graphical appearances of reaction partners. Furthermore, motif stamps are exchangeable between different network documents by an import functionality.

## 6.2. Appending Motifs to Reactions

Motifs can be appended in three steps: selecting a preferred motif stamp, selecting a reaction for appending the motif and finally adjust the motif's orientation. A motif stamp can basically be rotated, scaled and inverted in order to adjust it to the environment of the reaction it is appended to. These transformations are illustrated in Fig. 6.3 showing the semi-transparent silhouette of the appended motif around a target reaction node. The user can define the angle for the placement of the motif (Fig. 6.3 a). Fig. 6.3 b) shows a scaled version of the motif stamp. In Fig. 6.3 c) the motif is mirrored and aligned with the reaction's connections. After confirming the alignment angle, the motif is finally appended to the selected reaction node (Fig. 6.3 d).

In order to aid aesthetics, the software optionally helps to align the motif stamp in a smooth manner with respect to the already existing periphery of the reaction. Therefore, an axis is computed for the appended reaction node. The axis is the average of the tangents of all edges at the docking position of a reaction. Likewise, motif stamps have an axis which is indicated by a hairline (see Fig. 6.3 a to c). During the motif rotation by the mouse, the user can choose to automatically align the motif's axis to the axis of the reaction node (cf. Fig. 6.3 c).

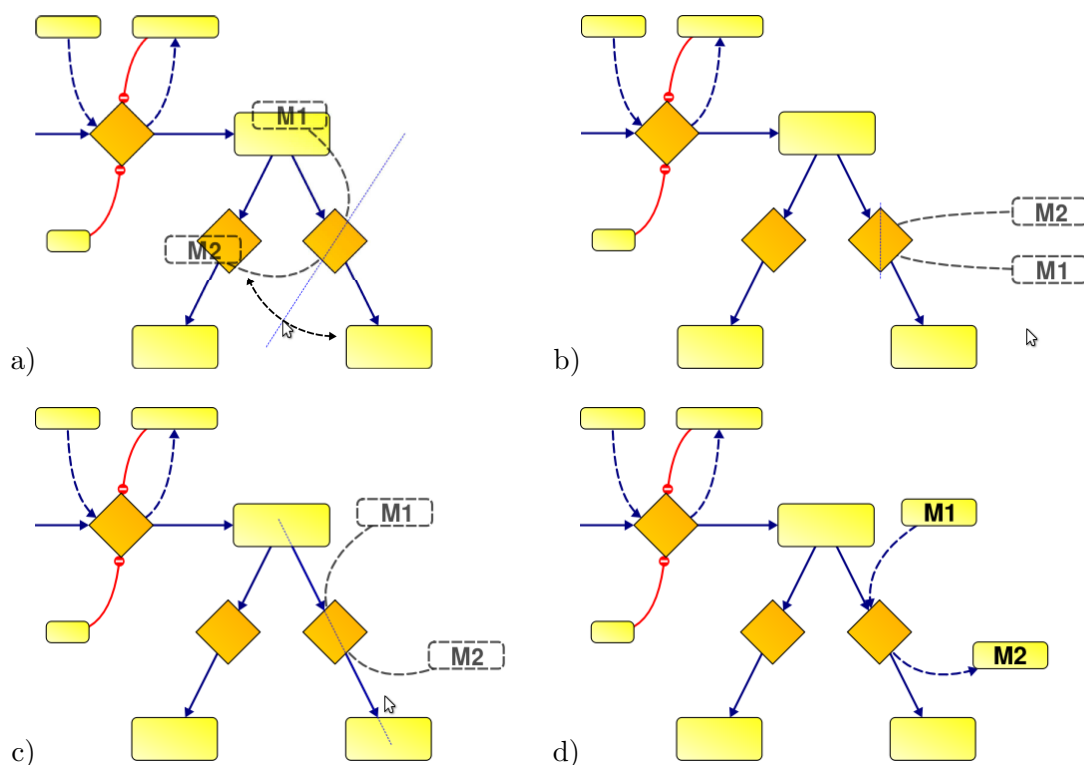


Figure 6.3.: Different transformations available when a motif stamp is appended on a reaction: (a) rotation, (b) scaling, (c) automatic “snapping” to a smooth arrangement, (d) final result.

When a motif is appended to a reaction, three different cases can occur and have to be taken into account (cf. Fig. 6.4 on the next page):

- The target reaction is not yet connected to a metabolite  $\mathbf{M}$  defined in the motif stamp — In this case, the metabolite  $\mathbf{M}$  is inserted and connected according to the motif stamp.
- The target reaction is connected to the metabolite  $\mathbf{M}$  and the metabolite node has further connections — Here, a further duplicate of the metabolite  $\mathbf{M}$  is inserted. The connecting edge is switched to the new copy and arranged as defined in the motif stamp. If the metabolite node does not yet have any layout information (see Section 3.6) the node and its edge is automatically inserted into the network diagram.
- The target reaction is already connected to the metabolite  $\mathbf{M}$ , which has no further connections — Only the arrangement of the node and edge has to be adapted. This also holds if the node has no layout information.

These three cases apply on every single metabolite connection defined in a motif stamp.

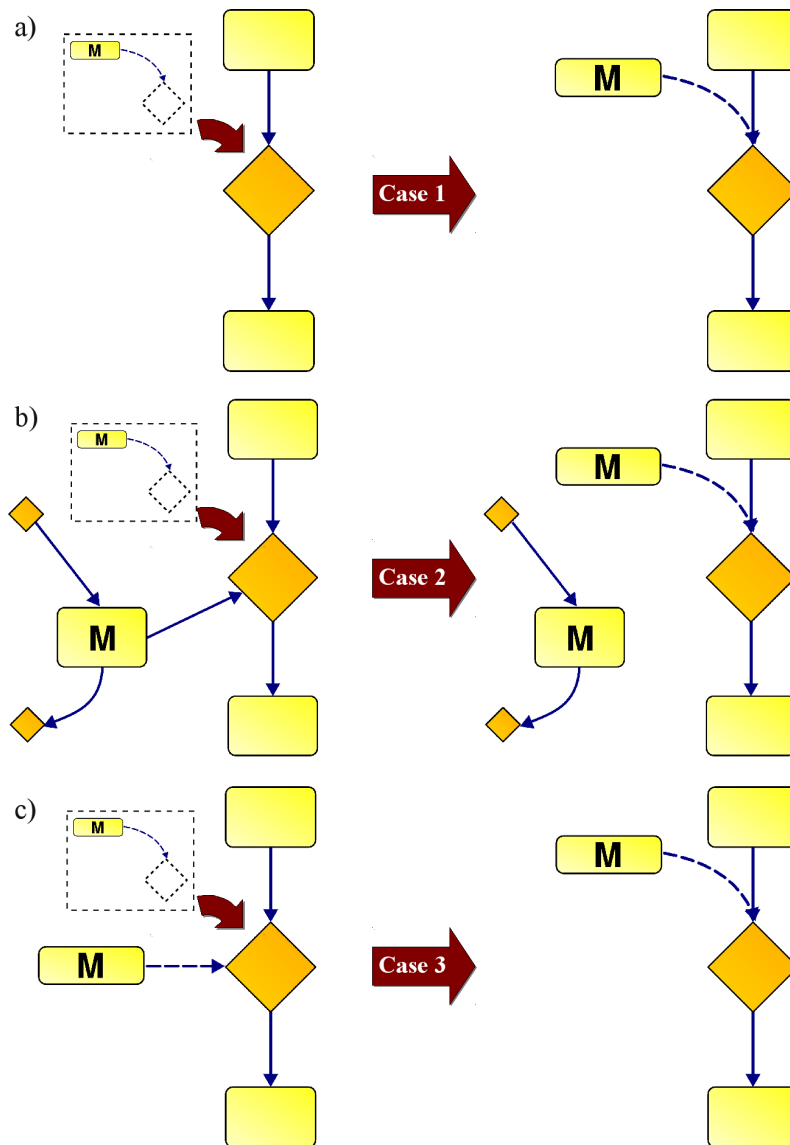


Figure 6.4.: The motif stamp defines the connection between a metabolite **M** and a reaction including curve shape, size and other visual properties. When a motif stamp (dashed box) is appended to a reaction, three cases arise which differ with respect to whether and in what way **M** is already connected to the appended reaction: (a) not connected, (b) connected amongst other connections, (c) solely connected.

## Chapter 6. Motif Stamps: Cloning Small Reaction Patterns

---

The motif stamp concept allows multiple motifs to be successively appended to a reaction node. Hence, a more complex constitution can be assembled from several motif stamps. The reaction periphery composed in this way can furthermore be a template for a new, larger motif stamp. Summarizing, it has become clear that motif stamps can be used to initialize new connections as well as to arrange existing connections between one or more metabolites and a set of reactions in the diagram. The usage of motif stamps can accelerate the drawing process considerably because metabolites do not have to be cloned manually and edges do not have to be drawn and reshaped every time a certain repeating configuration is needed. The motif stamps are a beneficial utility when multiply occurring cofactor pairs, side metabolites or effectors have to be drawn. The semi-automatic layout techniques introduced here are evaluated in a case study presented in Appendix B.

# Chapter 7.

## Exemplary Work Flow

This section describes a work flow of drawing a biochemical network diagram and, hereby, shows in which way the techniques of semi-automatic layout introduced above accelerate the drawing process. In this example, a metabolic network is given in the form of a model description file (e.g. in SBML Hucka et al. 2003) without layout information which is to be drawn with the network drawing software *Omix*.

After importing the network, all reactions and metabolites appear in the component view from where they are added to the diagram (cf. Chapter 3.6). An important precondition for the semi-automatic network drawing solutions as presented here is knowledge of the semantics of the network topology on the user side.

### 7.1. Metabolic Pathways

Before the network diagram is drawn it is recommended that reactions should be grouped into metabolic pathways (cf. Section 5.3.3). The Systems Biology Markup Language (SBML) format, for instance, does not directly support this kind of information; hence, it is not available after model import by default.

The user can add pathways to the network model and assign reactions to each of them (cf. Chapter 3.2.2). This optional task leads to the decomposition of the complex graph into smaller, less complex subgraphs.

### 7.2. Using the Layout Pattern

The arrangement of nodes and edges in the diagram can now be performed by applying the layout pattern concept. It is recommended to start with one of the central parts in the network. This is a common practice in drawing metabolic networks. In the sketched example work flow, the citric acid cycle (TCA) is chosen as a prominent example.

The next step is to draw a circular pattern section in the pattern editor (cf. Fig. 5.1 a), which will later carry the nodes of the TCA. After dragging a metabolite or reaction node from the component view and dropping it onto the circular pattern section the user is asked to select the network path representing the TCA in the network. This can be performed fast because the semi-automatic path search uses the above-mentioned classification into pathways as structural information about the network and, hence, offers the correct network path immediately.

After arranging the first pathway, the pattern can be enlarged by further sections in order to later arrange directly connected pathways. In this way, a structural backbone of the network diagram is drawn step by step.

### 7.3. Using Motif Stamps

After creating the structural backbone of the diagram motif stamps are used to arrange cofactors and connecting edges in the environment of reactions already inserted. For this purpose, several motif stamps can be drawn or imported from other network files. As an example, the metabolite CO<sub>2</sub> may be a product of several dozen reactions of the imported network. When the CO<sub>2</sub> node is inserted in the diagram all edges appear automatically, connecting the new node to the reactions in which CO<sub>2</sub> is involved. This results in many edge crossings.

Hence, a new motif stamp is created to duplicate the CO<sub>2</sub> node. A metabolite node CO<sub>2</sub> is inserted into the motif stamp and connected to the pseudo-reaction. The connecting edge can be reshaped as preferred. After creation, the motif stamp can be appended to all reactions of the network, particularly to all reactions which are already connected to CO<sub>2</sub> (cf. Section 6.2). Every CO<sub>2</sub> connection is visually designed according to the arrangement in the motif stamp. The isomorph design of similar parts of the network leads to an aesthetically pleasing diagram. In this way, all cofactors, and other secondary reactants can be inserted in the network diagram.

### 7.4. Completing the Drawing Process

Finally, all remaining network components which have not yet been inserted by the layout pattern or a motif stamp indicated in red in the component view are inserted manually. After finishing the arrangement of all imported network components, the visual style of the network can be adapted to individual preferences and requirements. The resulting designed metabolic network diagram can subsequently be used for a wide variety of visualization purposes. For this, *Omix* is equipped with extensive data visualization features as introduced in Part III.

**Part III.**

# **Visualization**





## Chapter 8.

# Introduction to Customizable Visualization

*“The purpose of information visualization is to amplify cognitive performance, not just to create interesting pictures. Information visualizations should do for the mind what automobiles do for the feet.” (Card 2007)*

### 8.1. State-of-the-Art in Multi-Omics Data Visualization

As introduced in Chapter 1 information visualization is an important tool for the interpretation of multi-omics data in systems biology. A wide variety of large-scale data arises incessantly from high-throughput experimental studies and computer simulations. In most cases, the data is directly related to nodes and edges of the metabolic network. This makes the visual representation of data in network diagrams the visualization technique of choice. By this network-integrated visualization, an overall view of data corresponding to the components of a network can be achieved and deeper insights into complex interrelations between the components is enforced. Multiple tools have been developed in recent years implementing a network-integrated data visualization approach. Section 1.9 has shown, that these visualization tools come up with several restrictions especially regarding adaptability. All known tools rely on one single or only few, hard-coded and mostly specialized visualization method. In the following, two visualization approaches are discussed that represent examples for very low and a very high customizability.

*ProMeTra* (Neuweger et al. 2009) is a web service that uses static network diagrams given in SVG file format to visualize metabolome, transcriptome and/or proteome data. The manner data is visualized in *ProMeTra* is hardly customizable and requires specific input formats. The user must submit their data in a specific arrangement and labeling as spread sheet file (Excel or CSV). Likewise, the drawing of the network must be given as SVG file. This file can be created in a normal graphics program but must specifically be prepared to be usable by the web service. The data is visualized by color-coding the network nodes. The user has the choice between eight different color codes (e.g. green to red, white to blue etc.) and fourteen value ranges (e.g.  $-1.0$  to  $1.0$ ,  $0.0$  to  $100.0$  etc.). Time-dependent data is displayed by subdividing the nodes into different parts each color-coded according the represented time point.

A very customizable example for omics data visualization is *Cytoscape* (Shannon et al. 2003). Here, the user can add custom attributes to the nodes and edges. For example,

an attribute “concentration” can be defined for all nodes. The user must specify the data type of the new attribute. Four data types are possible: text, boolean values as well as integer and real numbers. After defining, these attributes can be filled with concrete values by manual editing or by loading data from spread sheet files. The visualization of data is performed in a side window called “VizMapper™”. Here, the user can bind visual properties of nodes and edges to custom attributes. By this, multi-omics data is mapped to the appearance of the network diagram. The data mapping configurations can be exported as a separate file and is hereby exchangeable between different documents.

*Cytoscape* realizes a very flexible visualization approach that is adaptable do various application fields. The visual mapping of data is completely performed by the user and can be designed toward individual requirements and preferences. All known approaches to visualizing data in metabolic networks can be classified between the specificity of *ProMeTra* and the flexibility of *Cytoscape*. Nevertheless, the visual mapping method of *Cytoscape* has restrictions that motivate a new approach. *Cytoscape* does not support the animated visualization of time-dependent omics data. Furthermore, data can only be represented by the node and edge symbols themselves. Visualizations can easily become confusing if too many visual properties are used for data mapping (Saraiya et al. 2005). Hence, only a few types of data can be visualized simultaneously. There is no way to annotate the nodes and edges with further information carriers like labels or images.

“Life sciences research is, by nature, borderline chaotic” (Killcoyne and Boyle 2009) meaning that: “Research mechanisms constantly change; researchers are continually introducing new technologies and refining older technologies.” (ibid.) This highly-fluctuating application fields for visualization software requires high adaptability and customizability. The software *Omix* must meet these requirements in the underlying visualization approach.

## 8.2. Network-Integrated Visualization of Data

Omics-data can basically be visualized in two different ways inside a network diagram. The first way is by annotating the network components with concrete numerical values from the dataset as shown in Fig. 8.1 a). This method provides an exact, quantitative representation of data in the network diagram. In this way, the human observer can inspect concrete quantities but can hardly get a comprehensive overview of the dataset in combination with the complete diagram. Only two or three types of information should be visualized simultaneously in this manner, otherwise, the diagram would be overloaded which hampers the value of a visualization. Hence, a qualitative visualization of data in a diagram is more appropriate in many situations.

Qualitative data visualization can be performed by mapping the data to visual properties of the network components as exemplified in Fig. 8.1 b) to f). These appearance properties can be, for instance, the color fill level of network nodes (see Fig. 8.1 b) or the line thickness of edges (Fig. 8.1 c). Another method is to use color interpolation in order to indicate lower and higher numerical values (cf. Fig. 8.1 d). Categorical information

### 8.3. Basic Concept of the Visualization Approach

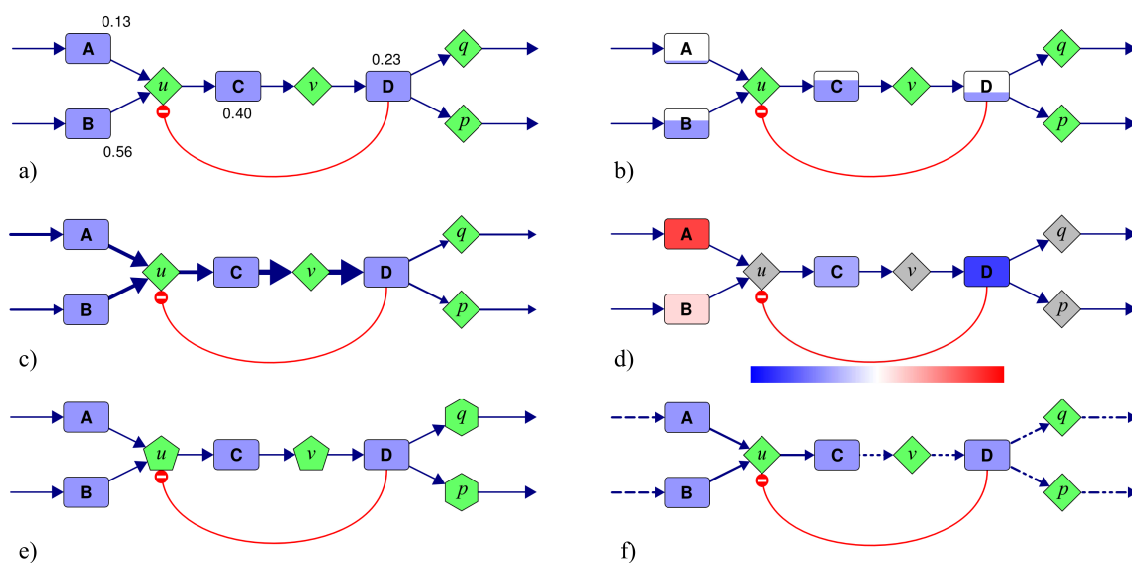


Figure 8.1.: Visualizing data in networks can be done by annotating nodes and edges (a) or by mapping the data to the visual properties of nodes and edges: (b) fill level size, (c) line thickness, (d) color, (e) node shape, (f) edge style.

such as states and roles can furthermore be visualized by using different node shapes or edge styles as shown in Fig. 8.1 e) and f). Of course the here mentioned set of visual properties suitable to represent data is not exhaustive. A combination of both visualization techniques is the annotation of network components with graphical items that themselves visualize information by their visual properties.

### 8.3. Basic Concept of the Visualization Approach

The core idea of the visualization approach introduced in this work emerged from the observation of typical user's actions during a manual preparation of a data visualization as it can be done with a graphics design software. For demonstration purpose, this situation is sketched here.

- Given: A dataset of metabolite concentrations available in a spreadsheet file and a network diagram showing all metabolic reactions of interest.
- Aim: The concentration data is to be indicated by the fill level of the metabolite nodes.
- Actions: The user selects a metabolite node in the diagram, searches for the corresponding value in the table and changes the fill level of the selected node. This is done until all metabolites in the diagram have been changed.

## Chapter 8. Introduction to Customizable Visualization

---

Considered from the view point of computer science, the user's actions sketched are an algorithm that can be automated. Listing 8.1 shows a pseudo program realizing the algorithm. Here, a value is read from the spreadsheet table identifiable by the metabolite name and assigned to the fill level size of the corresponding metabolite. This step is done for all metabolites in the diagram.

Listing 8.1: Pseudo code

```
for each m in metabolites
do
  m.fillsize :=read_value_from_table(m.name)
end
```

This small example illustrates the core idea for a highly-customizable visualization method, i.e. to allow the user automating own drawing activities by programming.

# Chapter 9.

## Script-based Visualization

Customizability is very important, because biotechnological research rapidly develops new visualization demands.\* The novel approach of network-integrated data visualization introduced here is to make network diagrams programmable in order to allow custom ways of mapping data to the appearance of the network components. Programming a network takes place in a particular scripting language that was developed for the visualization tool *Omix* called *Omix Visualization Language* (OVL). The user can easily write small scripts in OVL that manage the visualization of omics-data in network diagrams.

Fig. 9.1 shows the principle of OVL programming. The scripting language allows, for instance, defining a button in the main window of the tool. This button is connected to a function. Every time the button is pressed the function is executed. Such a function, for example, might load a time series of data that is mapped to the appearance of nodes and edges as illustrated in the pseudo code in Listing 8.1. The visualization of time-dependent data furthermore can be controlled by a media-player-like control. All these features can be programmed individually by the user. OVL offers many degrees of freedom to visualize data on biochemical networks by giving access to the visual properties of network components listed in Chapter 3.3.

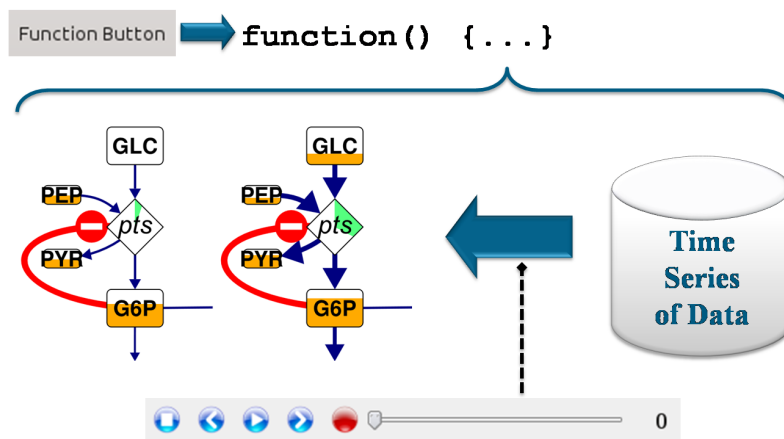


Figure 9.1.: Schematic diagram of an example for OVL programming

\*Excerpts of this Chapter have been published in (Droste et al. 2009a; 2010b).

In OVL visualization of omics-data can be realized in various ways. For the mapping of data to the appearance of network components OVL allows to change all visual properties of nodes and edges in the network diagram (cf. Fig. 3.6 on page 34). OVL comes up with an extensive library for data visualization containing methods for loading data from various sources and types, searching in datasets and scaling of data (refer Droste 2008b; 2010).

For the visualization of data by annotating network components, OVL allows to define additional information carriers in the diagram. Here, the capabilities reach from simple text annotations to comprehensive graphics items. Furthermore, interactive components can be defined in a simplified manner. In this way, for instance, interaction with single network components or the animation of time-dependent datasets can be realized. With OVL, the user can extend network components with further custom properties and, by this, augment a network with arbitrary information.

### 9.1. The Omix Visualization Language

The *Omix Visualization Language* (OVL) is a novel object-oriented scripting language derived from the programming language Java (Gosling et al. 2005, Oracle Corporation 2011). The OVL execution engine realized by *Omix* is based upon the *ANTLR* parser technology (Parr and Quong 1995). The main reason for creating a new scripting language for visualization is to realize a highly customizable visualization approach combined with the power of the Java programming language and the abundance of the Java runtime library. A lightweight introduction into visualization with OVL is given in the *Omix User Manual* (Droste 2008a). An expert reading is the *OVL Technical Manual* (Droste 2008b) describing the visualization abilities of OVL in detail and giving an exhaustive definition of the *Omix Visualization Language*. This section introduces the basic concepts of OVL including special syntactical constructs to simplify the programming of interaction and visualization. Some technical details shall demonstrate the elaborateness of the OVL scripting language.

#### 9.1.1. Extending Network Components

As introduced in Chapter 3.2, network diagrams consist of clearly demarcated, biologically meaningful network symbols such as metabolites and reactions, flux edges, effector edges, pathways and compartments. Each of these symbol types is defined with a set of properties like color, stroke width, arrow size, text font etc. In OVL this default definition of the network symbols can be extended. Therefore, the language introduces the keyword `extend`.

In Java the keyword `extends` is used for subtyping, i.e. deriving a class from another superclass. As a related concept, `extend` in OVL means to augment the definition of a class with further variables and methods. Listing 9.1 shows the principle setup of an extension block in OVL.

Listing 9.1: Type extension block in OVL.

---

```
1 extend «Type Name» {  
2     // contents...  
3 }
```

---

All network symbol types exist as OVL type definition, for instance, `Reaction`, `Metabolite`, `FluxEdge`, `Compartment` and others. In particular, the network diagram itself is also defined by an own type: `MetabolicNetwork`. Listing 9.2 shows an extension of the `Reaction` type with an additional attribute `database_ID` and some functionality implemented in the method `connectDB()`. After programming a network with the OVL script of Listing 9.2 all reactions are augmented with a further property. The property can be inspected and edited as other, predefined properties of the network items, e.g. the reaction name. Without defining any kind of interactivity with the network, as later introduced, the method `connectDB()` remains hidden and inaccessible for the user.

Listing 9.2: Extension of the `Reaction` type.

---

```
1 extend Reaction {  
2     String database_ID;  
3  
4     void connectDB(){  
5         // method realization...  
6     }  
7 }
```

---

The network symbol types have several methods allowing access to the network structure. For example, the type `MetabolicNetwork` contains methods like `metabolites()`, `reactions()`, `compartments()` and others returning lists of the corresponding network components in the diagram. The `Metabolite` and `Reaction` types allow access to their incoming and outgoing flux edges via `inEdges()` and `outEdges()` and the `FluxEdge` type has methods `src()` and `dst()` returning the source and destination node of an edge. By this, functionality can be realized as an extension to `MetabolicNetwork` that changes visual properties all over the network.

### 9.1.2. Simplified Access

The basic design target of OVL is to provide easy and programmable access to the visual properties of the network components. In OVL, the appearance of network components in the diagram can be changed by assignments. By this, many internal operations are immensely simplified. For example, in Listing 9.3 the width of an edge line is set to a new value. This simple assignment in OVL wraps an amount of polynomial computations and an update of the drawing area that is done when the line is executed and the edge shape is changed.

Listing 9.3: Assignment: Changing the line width of an edge.

---

```
1 lineWidth = 5;
```

---

All visual properties are available in OVL as variables of the corresponding network component type. They can be changed by assigning new values to the variable. In contrast to Java, an assignment in OVL causes notification of change. In a normal Java program, an object cannot recognize and react on changes of its member variables that are performed from outside the object code. Therefore, variables are usually declared `private` and changes are only allowed by using a mutator (“setter”) method. This concept is called *encapsulation* (Scott 2009). Inside of the mutator method, the program can realize any kind of response to the changed value. In OVL, member variables are an abstraction of the underlying complex programming structures including the graphical effects of variable changes. Assignments are much easier to read and to understand than the Java-typical “setter” methods.

### 9.1.3. Data Types in OVL

Similar to Java, OVL is type sensitive. This means that methods, parameters and variables are declared with a data type, and the actual value returned by a method, stored in a variable and submitted as parameter, respectively, must correspond to the declared data type. OVL has exactly the same type system as Java including primitive types like `int` and `double`, array types, object types and the type `void` for methods.

All classes available in Java can be used in OVL as object types. Furthermore, own classes can be defined in OVL code. OVL supports inheritance as well as information hiding. Unlike normal hard-coded Java classes, OVL classes can be created and modified dynamically at run time. Furthermore, the visualization tool is able to inspect objects of OVL classes, display all attributes, and even respond to attribute changes. An example for OVL classes is given in Section 9.3.2.

### 9.1.4. Interactive Components

OVL allows the definition of interactive components in the main window. These so-called *triggers* can be used to access a certain functionality implemented in the OVL code, for instance, starting a visualization or controlling an animation. OVL provides different trigger types with different properties, for example, stateless push buttons, check buttons with a boolean state (checked/unchecked), sliders and spin boxes with a number value, and players as media-control-like triggers. Triggers are always associated with a component of the network diagram. Therefore, the definition of triggers takes place inside of extension blocks. By this, the network components can be equipped with interactive components. If a trigger is defined for the `MetabolicNetwork` the interactive component appears on a separate toolbar and in the main window (see Fig. 9.2 a). By equipping other network symbol types with triggers, the interactive elements appear as accessory of



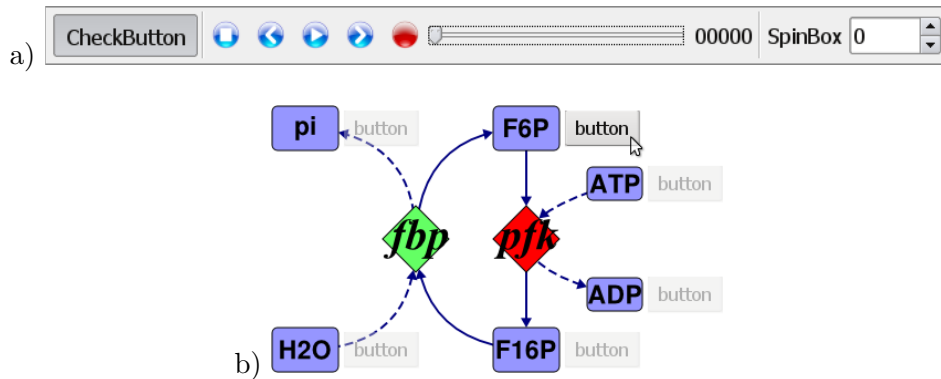


Figure 9.2.: Triggers as additives of the network: (a) three triggers (check button, player and spin box) on a toolbar, (b) push button as equipment of metabolite symbols in the diagram.

the corresponding graphical items in the diagram as depicted in Fig. 9.2 b). In this way, interaction with the single network components can be realized.

### Trigger Syntax

For the definition of triggers a special syntax has been developed which is simple and avoids programming errors. A trigger definition starts with the trigger type and a variable name. The name is followed by a colon and a method signature:

«Trigger Type»«Variable Name»:«Method Signature»;

OVL provides eight different trigger types (for a detailed description refer Droste 2008b). The method signature consists of a method name and the types of the method parameters in parentheses whereas the signature must represent an existing method of the item extension. The colon operator connects the interactive component with the method. Every time the trigger is activated by the user the connected method is invoked. Since triggers have a state the method parameters must comply with the kind of trigger state.

The relation between the trigger states and the method parameters is exemplified in Listing 9.4. For instance, a `PushButton` is stateless and consequently connected to a parameterless method (line 1). A `CheckedButton` can be connected to a method with a `boolean` parameter (line 2). The checked state is correspondingly submitted at user interaction. A `SpinBox` is an editor component for integer numbers. Hence, the trigger can be connected to a method with an `int` parameter (line 3).

Listing 9.4: Exemplary trigger definitions

```

1 PushButton pushButton : pushButtonPressed();
2 CheckButton checkButton : checkedChanged(boolean);
3 SpinBox spinBox : valueChanged(int);

```

### Trigger Properties

The single trigger types contain properties concerning their visual and functional attributes analogue to the network item types. For instance, the `SpinBox` has a `minimum` and `maximum` variable defining the value range of the integer number editor. These property variables can be accessed in OVL and simply changed by assignments as introduced in Section 9.1.2. Beside changing the trigger properties from inside a method, OVL allows to predefine the property values during the trigger definition. Therefore, the trigger variable name can be followed by a number of parameter assignments enclosed in parentheses:

«Trigger Type»«Variable Name»( «Assignments» ) :«Method Signature»;

This is exemplified in Listing 9.5 as a modification of Listing 9.4.

Listing 9.5: Exemplary trigger definitions with initial property assignments

---

```
1 PushButton pushButton(enabled=false) : pushButtonPressed();
2 CheckButton checkButton(checked=true) : checkedChanged(boolean);
3 SpinBox spinBox(minimum=0, maximum=100) : valueChanged(int);
```

---

### Design Motivation

The special syntax of triggers is favorable due to four reasons:

1. Simplification – Programming of interactive components in normal Java development is much more complicated. Listing 9.6 shows an example implementation of Listing 9.5 line 1 with Java. OVL is easier to read and to understand.

Listing 9.6: Implementation of user interaction in Java

---

```
1 JButton pushButton = new JButton("pushButton");
2 pushButton.setEnabled(false);
3 pushButton.addActionListener(new ActionListener(){
4     void actionPerformed(ActionEvent e){
5         pushButtonPressed();
6     }
7 });
```

---

2. Restrictions of Java – In Java there is no way to create a compile time constant reference of a method. A reference to a method can only be created by using the Java Reflection (Forman et al. 2004). Here, the Java compiler is not able to check if the referenced method actually exists. Hence, invalid references cause exceptions during the program execution.

Listing 9.7: Using the Java Reflection to reference a method.

---

```
1 try{
2     Class<?> myClass = this.getClass();
3     Method checkedChanged = myClass.getMethod(
4         "checkedChanged", boolean.class);
5     // when method does not exist exception is thrown
6 }catch(NoSuchMethodException e){
7     e.printStackTrace();
8 }
```

---

OVL introduces a new syntactical construct for the compile time sensitive referencing of methods in the program code, i.e. the compiler checks if the method signature of the trigger definition references an existing method in the code.

3. Definition equals construction – The definition of a trigger variable causes the construction of the trigger. Constructor calls like in Java are not necessary in combination with triggers.
4. Definite composition – A trigger can never be uncoupled from its owner or deleted. It is not allowed to change a trigger variable. By this, a trigger belongs unambiguously to the network component it has been defined in. The situation would be not clear if normal Java-like constructor calls and assignments was allowed in combination with triggers as sketched in Listing 9.8.

Listing 9.8: Example scenario of trigger construction and assignment.

---

```
1 CheckButton button = new CheckButton();
2 for(Reaction reaction : reactions()){
3     reaction.checkedButton = button;
4 }
5 // who actually owns the button?
```

---

### Example OVL Script

Listing 9.9 implements a simple example as extension to the `MetabolicNetwork` type. Here, a button is defined that invokes a method when pressed by the user. Inside of the method the color fill level of all metabolite symbols in the network is changed to a random value as seen in Fig. 9.3.

The impact of the simple code example in Listing 9.9 is that a button occurs on the toolbar of the visualization tool. When this button is pressed, the described changes in the diagram are performed. In this example, the assignment of a random value was chosen for the sake of brevity. Of course, methods in OVL can implement data computation as well as file and database in- and output as source for omics data.

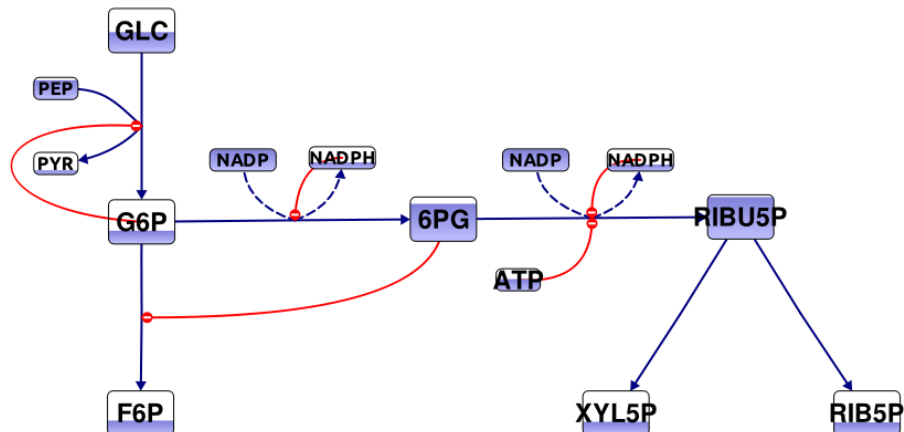


Figure 9.3.: Result of the OVL example in Listing 9.3.

Listing 9.9: OVL code example: Changing the color fill level after button click.

```

1  extend MetabolicNetwork {
2      PushButton change : changeLevels();
3
4      void changeLevels() {
5          for(Metabolite m : metabolites()){
6              m.fillLevel = Math.random();
7          }
8      }
9  }

```

### 9.1.5. Additional Information Carriers

OVL allows to equip the network diagram with additional information carriers. These can be textual annotations, images and even comprehensive graphical items. In OVL these information carriers are called *data annotations*. Both concepts, data annotations and triggers compose the group of *accessories*. Accessories are additives to the network diagram and its nodes and edges.

Data annotations are defined with the same syntax as used for triggers. In contrast to triggers, data annotations are not interactive and consequently cannot invoke methods. Hence, the method signature is simply exchanged by the keyword `void`:

```
«Data Annotation Type»«Variable Name»( «Assignments» ) : void;
```

OVL provides seven different data annotation types (cf. Droste 2008b). Fig. 9.4 a) shows a metabolite equipped with an `ImageField` and a `TextField` annotation. The data annotation type `SubNode` has the same set of visual properties as reaction and metabolite symbols and, thus, can be used to visual data mapping (cf. Fig. 9.4 b).

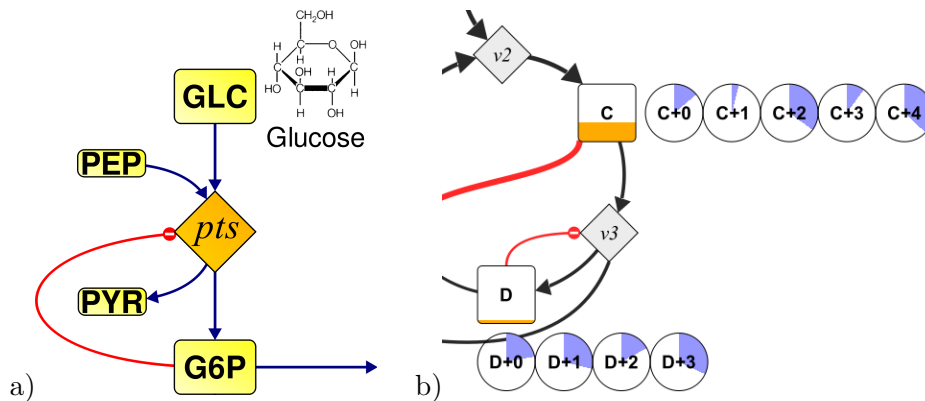


Figure 9.4.: Examples for data annotations as additives of metabolites: (a) ImageField and TextField, (b) arrays of data annotations (SubNode)

### 9.1.6. Accessory Arrays

Beside extending network components with single triggers and data annotations, OVL allows to specify arrays of accessories with flexible size. An example is given in Fig. 9.4 b). The particular accessory type is called `Array`. The type of the accessories stored in the array is specified by a generic type parameter, for instance, line 1 of Listing 9.10 defines an array of text annotations. An accessory array definition can also define properties for the inner accessories. These properties must be separated from the array properties by a semicolon (see line 2). It is also possible to define multiply nested arrays (line 3). Furthermore, it is possible to define arrays of triggers. In this case, the connected method must contain a leading `int` parameter as shown in line 4 of Listing 9.10. When the method is called by trigger activity, the `int` parameter contains the index of the calling trigger.

Listing 9.10: Arrays of data annotations

```

1 Array<TextField> textFields(size=5) : void;
2 Array<SubNode> subNodes(size=2; width=20, height=30) : void;
3 Array<Array<IntegerField>> arrays : void;
4 Array<CheckBox> checkButtons : checkedChanged(int, boolean);

```

### 9.1.7. Meta-Information

Variables in OVL scripts are not only visualized in the network but are also visible and editable in the property editor sidebar and in other window components of the visualization tool. The presentation of these variables in window components and likewise the appearance of triggers can be supplemented with meta-information by using the code annotation concept of Java. In Java, variables, methods and other programming constructs can be annotated by special annotation expressions consisting of an '@' character, a type name and optional parameters. This concept is adopted in OVL. Particularly, OVL provides a set of annotation types that take effect on the appearance of variables and accessories.

The code example in Listing 9.11 provides meta-information for a variable and a trigger. The `concentration` variable in line 1 is supplemented with a physical unit (mole) in order to be correctly interpreted. Here, the annotation type `Unit` causes suffixing of the displayed value with the unit symbol wherever it is displayed in windows or in the diagram. Line 2 changes the label of the push button `changeLabels`. By default, triggers are labeled with the name of the trigger variable. However, variable names are restricted to a limited character set. Hence, the `Label` annotation type is useful for overriding variable names with more general labels. Refer the *OVL Technical Manual* (Droste 2008b) for further annotation types provided by OVL.

Listing 9.11: Semantic code annotation in OVL

---

```
1    @Unit("mol") double concentration;
2    @Label("Change Levels")
3    PushButton changeLabels : changeLevels();
```

---

## 9.2. OVL Development

OVL scripts can be developed as internal part of a network document or as standalone text file. An OVL program can only be applied as embedded part of a network document. Standalone OVL files cannot be executed, but, can be loaded into network documents. The import/export functionality of the OVL editor allows that OVL solutions can quickly be exchanged between different network documents.

The development of OVL scripts is supported by a complete integrated development environment (IDE) for OVL realized as part of the visualization tool. The IDE contains a syntax-highlighting editor and a parser that returns comprehensible messages in case of programming errors. An overview table displays the structure of the OVL program for a fast navigation in the code. A search engine helps to find and inspect available resources for the OVL development. Fig. 9.5 on the next page shows the main window of the IDE for OVL development.

An outstanding feature of the IDE is a debugger for line by line execution of OVL code and for the inspection and manipulation of local variables during the program execution. The debugger allows to verify the program and helps to understand OVL programming by observation of the internal states caused by the program code. In Fig. 9.5 on the facing page the debugger currently breaks at a certain line of the OVL code. The table on the right side lists all local variables. Besides, object types can be expanded in order to inspect the values of their internal variables. The variables can be edited in order to manipulate the program flow.

OVL programs can use the complete Java runtime library that provides useful features like data collections, file management, XML parsing etc. Moreover, OVL programming is supported by a special OVL library containing classes and interface definitions that represent network components, interaction components, properties and diverse utilities for the network-integrated data visualization. For instance, the OVL library offers several Java

### 9.3. Visualization on Demand

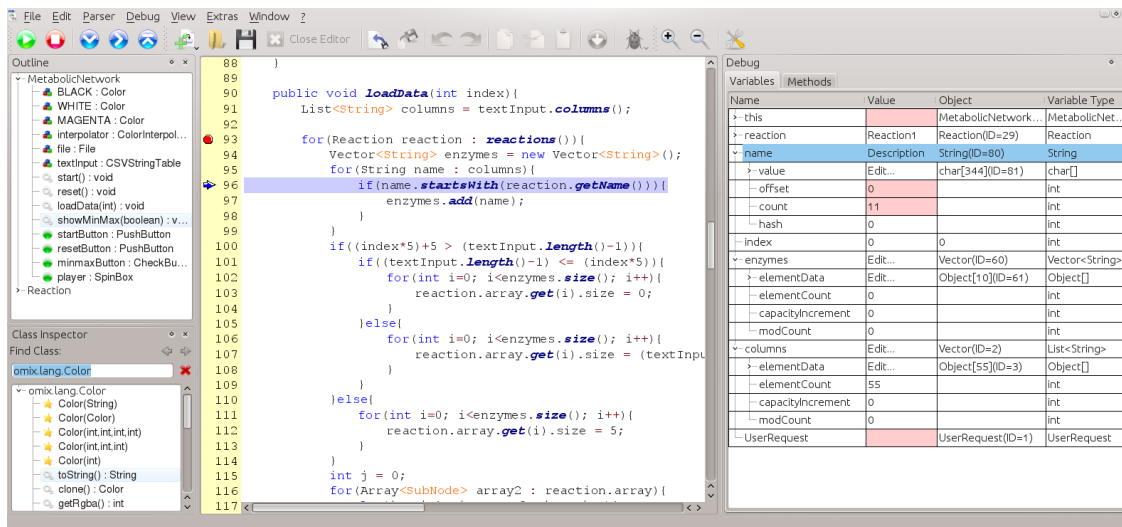


Figure 9.5.: Screen shot of the OVL development environment. The center shows the syntax-highlighting editor. The left side windows contain information about the program structure. The right side of the window shows the local variables of the program during a debug operation.

classes in a simple programming interface that are specialized for extracting information from CSV files. The reference documentation of the OVL library is publicly available (Droste 2010).

Advanced developers can include any Java and even native libraries (via Java Native Interface, JNI) in OVL scripts. Hence, custom Java classes can be developed implementing arbitrary functionality as supplement to the OVL visualization solution. In particular, the OVL execution environment allows accessing OVL code from inside Java code. Thus, a user written Java class can, for instance, read OVL fields, change the appearance and states of network components and accessories and even call OVL methods.

### 9.3. Visualization on Demand

Network-integrated visualization faces the challenge that each individual piece of information is related to a specific node or edge of the network, and that complex interactions between hundreds and thousands of different compounds must be assembled and represented even in mid-scale network models. As introduced in Chapter 8.2, data can be visualized in association with a biochemical network by annotating the network components with data. However, annotation of nodes with more than a few pieces of information causes information overload, particularly in large networks. Moreover, inherent layout problems occur when annotations are to be placed in the neighborhood of related network components without collisions with other nodes, edges or annotations. Another suggested technique

was to map the data to the visual appearance of network components. Unfortunately, this concept is inherently limited by the human ability to capture several information types from one diagram or animation.

This section addresses the visualization of manifold and extensive information in association with nodes and edges of large biochemical networks, especially when the information pieces are hierarchically organized with variable depth. The simultaneous visualization of manifold and extensive information over entire diagrams of realistically sized biochemical networks is practically impossible. Therefore, a novel *Visualization on Demand* approach has been developed that solves this dilemma with interactive diagrams.

### 9.3.1. Interactive Visualization of Custom Properties

The cornerstone of the *Visualization on Demand* approach is the ability to extend the network components with further variables of arbitrary data types (cf. Section 9.1.1) and to define own class types in OVL as introduced in Section 9.1.3. OVL classes can define member variables of complex data types that themselves have own member variables of various types. This enables the user to compose hierarchies of information in any manner and depth.

*Visualization on Demand* (VoD) is an optional network viewing mode in the visualization tool. The VoD mode facilitates user controlled displaying of supplementary information that is related to metabolite and reaction nodes, and to their interconnecting edges in arbitrary levels of detail. In VoD mode, simple mouse hovering over network elements activates displaying of all in OVL defined variables of the respective node or edge as illustrated in Fig. 9.6 on the next page. The user can open further levels of detail by hovering over the respective displayed variable. Layout problems and information-overload are effectively avoided by showing properties of only one network component and zooming in the details of only one property at a time.

Global representation of the network is hardly affected by the details of single components, because user perception is temporarily directed to individual pieces of information in their local context. The nodes and edges of the biochemical network are globally arranged in a macro layout, whereas the hierarchically organized details are locally arranged in a dynamic tree-structured micro layout at the corresponding network components. When the mouse hovers over a node or edge, the associated properties are itemized to the left and right of this network component. In addition, the individual items are connected with the corresponding node or edge by a dashed line.

The VoD approach is similar to a contextual menu. Context menus are a standard interface component in nearly all desktop applications providing context-dependent functionality in arbitrarily nested menu entries. In opposition to standard context menus, the appearing VoD items do not represent actions but information. They are an integrated component of the graphics scene. Thus, VoD can be captured by the image and movie export feature of *Omix*. A similar approach is realized in *CellPublisher* (Flórez et al. 2010) by using the *Google Maps JavaScript API* (Google Inc. 2011) to interactively visualize background information on a network map. The strength of the here presented



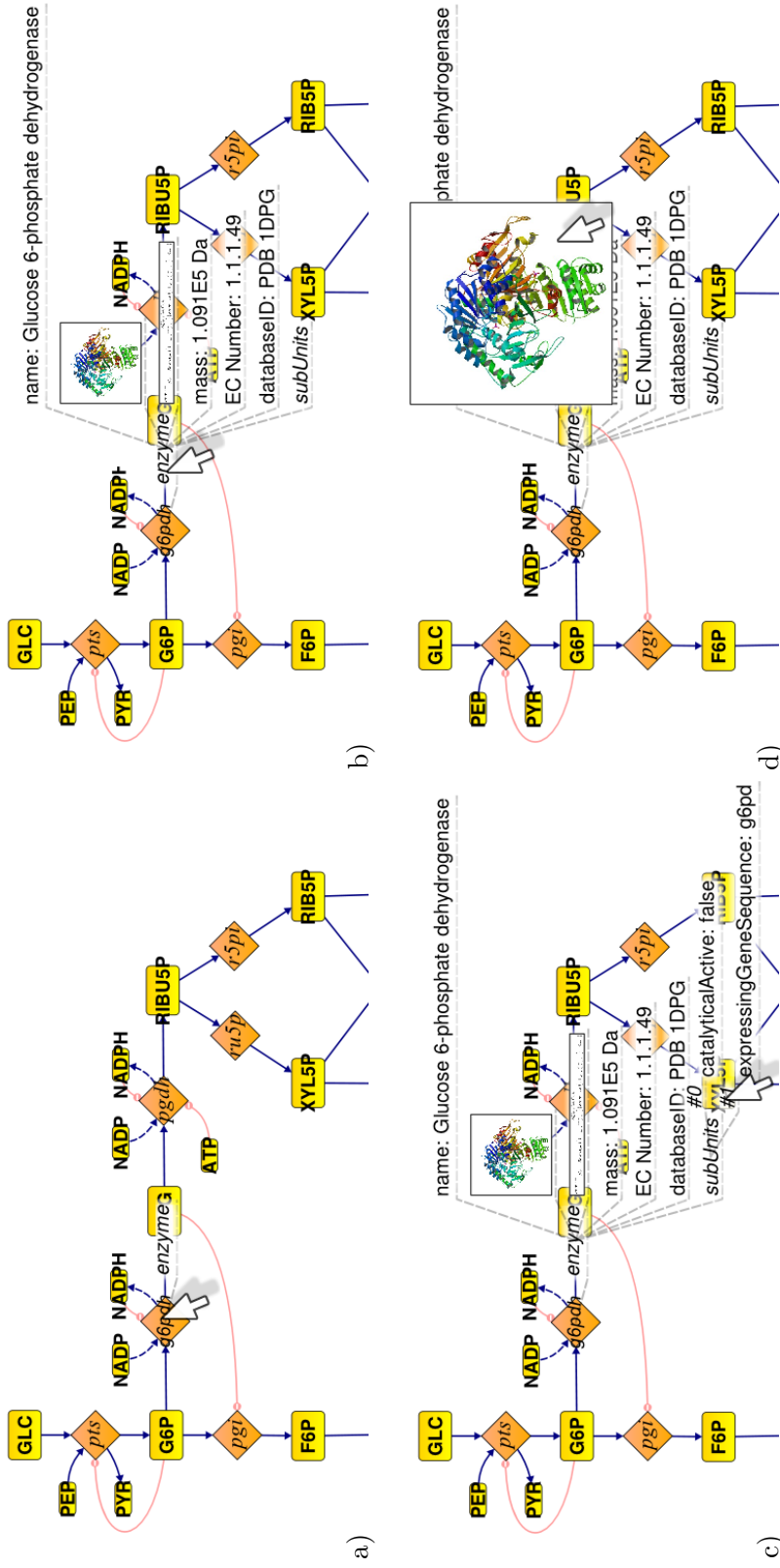


Figure 9.6.: Visualization of the "enzyme" property in several levels of detail: (a) the "enzyme" property appears after hovering the node, (b) all information in the "enzyme" property is shown composed in the "enzyme" property, (c) further information is given, (d) enlarging an image by mouse wheel.

VoD feature is, the manner information is composed and organized in the network can completely be programmed by the user.

### 9.3.2. Custom Data Types

Listing 9.12 shows how to use OVL for equipping biochemical networks with various hierarchically organized information. In lines 1 to 3, an extension of `Reaction` is defined containing a variable that stores information about the reaction's enzyme. The class `Enzyme` in lines 4 to 12 encapsulates not only the enzyme name, but also a structural image of the molecule shape, a MathML formula of the enzyme kinetics, the atomic mass, the Enzyme Commission (EC) number, the enzyme ID in a reference database, and a list of subunits that are stored in an array.

Listing 9.12: Implementation of a custom data type.

---

```
1  extend Reaction {
2      Enzyme enzyme;
3  }
4
5  class Enzyme{
6      String name;
7      Image image;
8      MML kinetics; // as MathML formula
9      double mass;
10     String ecNumber;
11     String databaseID;
12     SubUnit[] subUnits;
13 }
```

---

The variable types `String` (line 6 of Listing 9.12), `Image` (7) and `MML` (8) used for defining the example `Enzyme` class are native Java classes, and provided by the Java runtime library and other sources. The data type for describing the enzyme subunits (line 11) is an OVL class that combines information on the activity of the subunit with information on the expressing gene that could be a further custom data type. Listing 9.13 shows a possible implementation of `SubUnit`.

Listing 9.13: Implementation of a custom data type.

---

```
14 class SubUnit{
15     boolean catalyticActive;
16     Gene expressingGene;
17     //...further properties
18 }
```

---

The example illustrates in what manner the script-based visualization approach enables the user in defining the organization of large data amounts in hierarchies, relations and aggregations along the nodes and edges of biochemical networks. Individual networks components can be augmented with many different information types that evolve from or

refer to experiments, simulations and databases. Depending on the nature of the supplementary information the definition of composed OVL class types can help to subclassify complex data into different levels of detail.

### 9.3.3. Information Management

Custom properties of network objects are initially empty or set to default values. In order to insert data into network diagrams the visualization tool offers two ways of loading property variables with information: First, an OVL script can be programmed that imports data from files or databases and automatically assigns this data to properties of network components (cf. Section 9.2). Second, data can be inserted into networks by manually changing the properties of all network components in the property editor (cf. Section 3.3). The window collection of the tool provides editor components for a multitude of Java types which are frequently used in OVL, for example, floating point numbers, boolean values, plain text, colors, lists and arrays or file references, and many more. Generally, objects of all kinds of Java classes can be constructed via dynamic access to their class definitions in a set of dialog windows as shown in Fig. 9.7 on the following page.

### 9.3.4. VoD Example

Fig. 9.6 on page 83 shows four snapshots of a typical interactive visualization example for the `Enzyme` property of a reaction from Listing 9.12 on the facing page with data from the Protein Data Bank (PDB) (Berman et al. 2000): In Fig. 9.6 a), the mouse hovers over the reaction node, and the property `Enzyme` appears besides the node. The appearance of such property items is animated in order to assist the user with distinguishing between underlying network elements and supplementary information. If the displayed variables contain complex values, for example arrays, lists, or objects of custom OVL classes, the representing graphical items can be further expanded for displaying the next level of subordinate variables. The display of subordinate information is also triggered by mouse hovering over expandable property items as illustrated in Fig. 9.6 b) and c).

In Fig. 9.6 b) the mouse hovers over the `Enzyme` property causing its expansion. The name, image representation, kinetic equation, mass, EC number, database ID and sub-units of the enzyme become visible. In Fig. 9.6 c) the `SubUnit` property of the enzyme is additionally unfolded. Images that are elements of itemized properties are initially displayed with a rather small default size, in order to avoid overloading of the network diagram. On demand, such images can be magnified up to their original size by mouse wheel interaction as illustrated in Fig. 9.6 d). This way, the user can visualize arbitrary levels of detail on demand.

## 9.4. Advantages of OVL

The biggest advantage of OVL is: the manner of data visualization in network diagrams is completely customizable and can be adapted to any preferences and requirement of the

## Chapter 9. Script-based Visualization

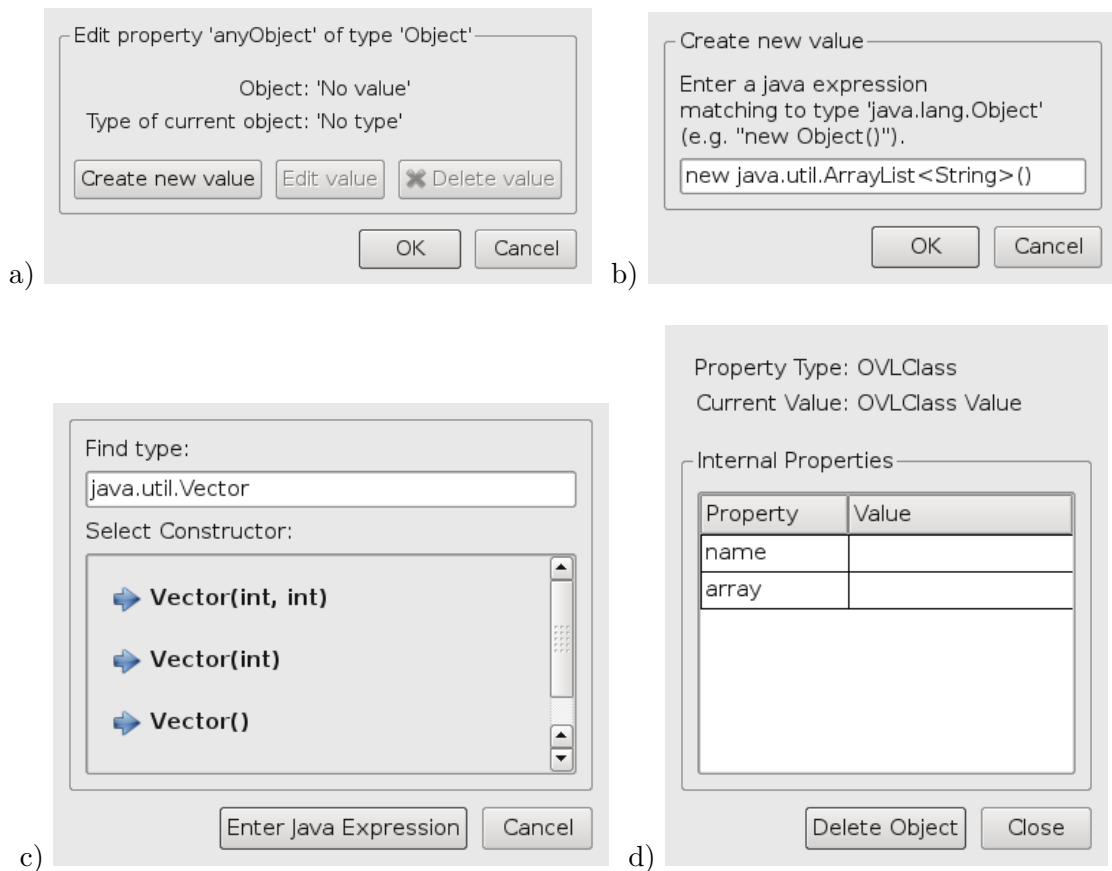


Figure 9.7.: Dialog windows that offer dynamic access for constructing objects from arbitrary Java classes and custom OVL types: a) object creation and deletion, b) input of Java expressions for dynamic object construction, c) constructor selection for creating objects of unspecified type, d) editor dialog for values of OVL defined data type.

single user. Inasmuch as OVL is based on Java, the complete power of the Java programming language and Java runtime library is available in OVL. This allows comprehensive, object-oriented solutions to be developed in OVL but, nevertheless, does not hamper the usability of OVL.

OVL simplifies many internal operations. Thus, big effects can be achieved in only a few lines of code. This makes visualization with OVL easy to learn. The slenderness of the OVL example code given in the tutorials (cf. Droste 2008a;b) allows a rough understanding. These examples can be adapted to solve personal visualization tasks even by users without skills and expertise in programming. An OVL script can be applied on multiple networks since it can be freely exchanged between different documents and

different users. In this way, a pool of existing solutions for different visualization issues can arise, from which all members of a community can avail themselves of.

In recent studies many OVL solutions have been developed dealing with the visualization of metabolome, fluxome and transcriptome data, regulatory effects, thermodynamic data, literature and database information, isotopic mass distributions, comparative analysis of several datasets and more (cf. Chapter 10). In any case, the animation of time-dependent data is possible in OVL.



# Chapter 10.

## Application Examples

During the last years collecting experiences with the script-based visualization approach many visualization solutions have been developed in OVL. This Chapter presents a set of application examples pointing out the flexibility and impact of visualization with OVL. The interdisciplinary authorship of the presented visualization examples demonstrates the acceptance of script-based visualization under life scientists.\*

### 10.1. Comparative Visualization of Metabolome Data

In a typical metabolic fingerprinting approach intracellular concentrations from a wide range of metabolites are detected with different analytical devices (Oldiges et al. 2007, Kanani et al. 2008, Luo et al. 2007) and interpreted with respect to correlations between relative changes of metabolite abundances between alike samples. This includes the comparison of metabolite levels under different experimental conditions or different strains of the same organism. The main goal of comparative metabolome studies is the determination of significantly changed metabolite concentrations which are expressed in ratios relative to a reference dataset. Due to the wide range – intracellular metabolic concentrations vary between some *nmol* up to several *mmol* – and the quantity of metabolites – several hundred per experiment – a simple comparison from peak areas or heights of mass spectra is not practicable. Thus, a conversion into comparable values, i.e. concentrations, has to be enforced by peak area integration. Finally, visualization is a key for getting an overall impression about the dataset and towards understanding the concerted changes of metabolite pools within the cell in relation to the metabolic network. In order to highlight the main changes in comparative analyses a specific visualization solution has been developed in OVL.

In Fig. 10.1 on the next page the central carbon metabolism of *Corynebacterium glutamicum* including simplified biosynthesis pathways for amino acids is shown. Metabolites are indicated by orange-colored ellipses. Reaction symbols are hidden in the view, because they are not of primary interest. Instead, the flux edges are joined at the reaction node's center and labeled by the reaction name (cf. Chapter 3.4). Visual attributes of metabolite nodes like color, size, or shape can be directly used to visualize metabolite concentrations.

---

\*Excerpts of this Chapter originate from (Droste et al. 2009a; 2011a).

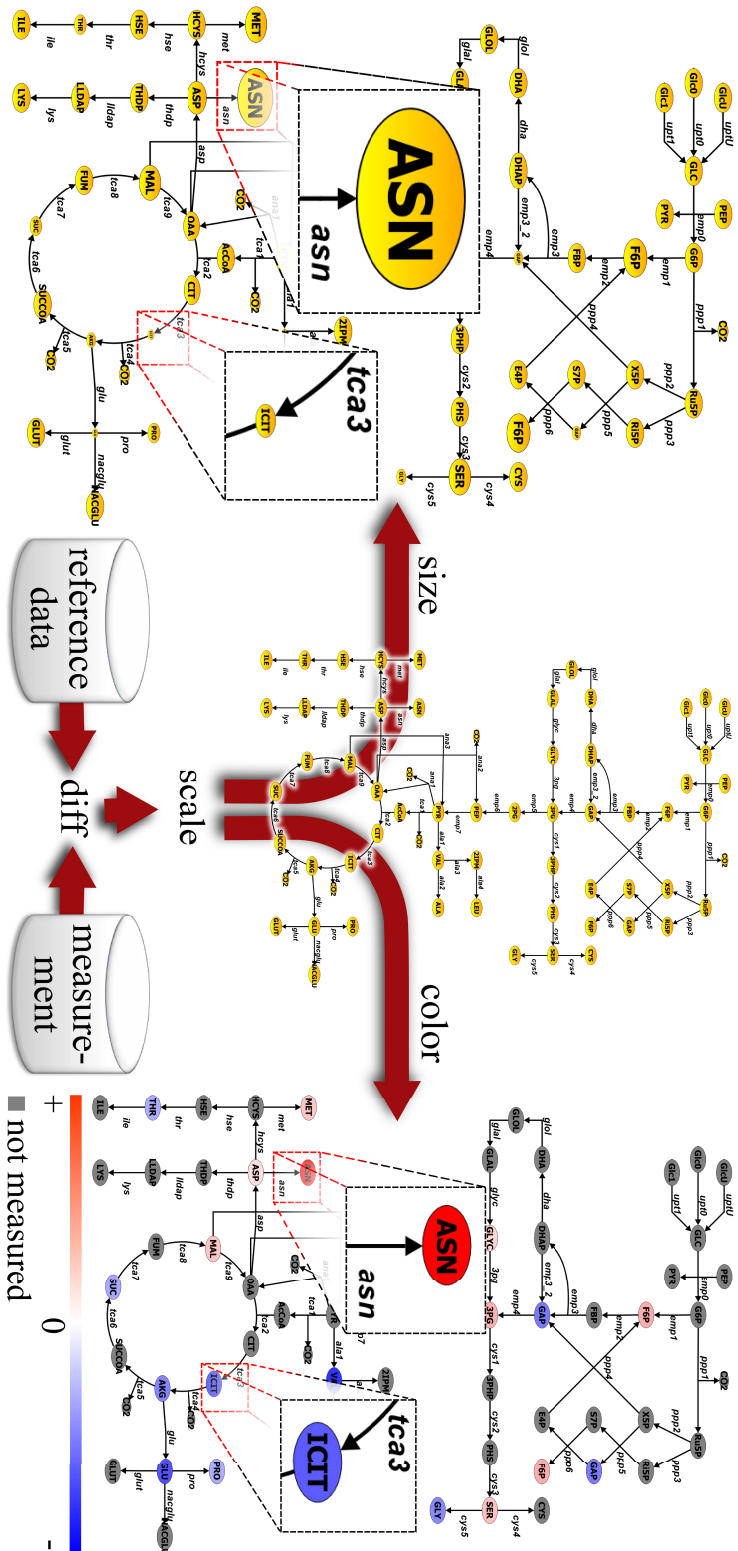


Figure 10.1.: Steps to a comparative visualization of metabolome data. The measurement data is compared with a reference dataset (center). The comparative data is scaled in an appropriate range. Thereafter, the data can be visualized by size (left) and color (right).



## 10.2. Visualization in the Context of Metabolic Flux Analysis

---

As indicated in Fig. 10.1 (center), the measured concentration data is pre-processed before visualized in the network context. In a first step, the data is normalized against the reference dataset. Here, the comparison to a reference dataset leads to widespread proportions with several orders of magnitude. It is a challenge to visualize these proportions without hiding small changes (less than factor of 2). Thus, the step of data scaling has to be done with care, in order to avoid a biased or distorted impression. There are different methods (linear, exponential etc.) to scale data into a defined range feasible for visualization. The data pre-processing has been realized in *Matlab* and *Excel*.

After scaling the comparative data in range  $[-1, 1]$  they are visualized in the network diagram. Therefore, two methods have been implemented in OVL. The first method shown in Fig. 10.1 (left) is mapping the data onto the metabolite symbol size. Big symbols indicate higher concentrations in the measurement than in the reference dataset and, vice versa, smaller symbols show lower concentrations. Here, a challenge is the limited space for scaling the metabolite symbols. Thus, the scale factor 2 has been chosen to represent the maximum discrepancy.

Another way, to visualize the comparative data is by coloring the metabolite symbols (cf. Fig. 10.1 right). Here, a color interpolation from red to blue has been chosen. White color represents equality between a measured and the corresponding reference concentration value. Higher concentration is colored red, lower blue. By this, the intensity of the color indicates the deviation from the reference value. The color-coded data visualization, furthermore, makes it possible to show unmeasured metabolites in the diagram. This information is given by gray colored metabolite symbols.

The graphical illustration of comparative metabolome data in biochemical network diagrams allows a fast qualitative impression of the dataset in relation to the network. In this way, significant deviations can directly be detected and influences of cultivation conditions or genetic modifications can be classified in biochemical pathways. The visualization example consists of 90 lines OVL code. The script has been developed by a biotechnology engineer with metabolomics and fluxomics objective.

## 10.2. Visualization in the Context of Metabolic Flux Analysis

One of the important aims of systems biology is to determine the velocity of all reactions (the fluxes) in a metabolic network. The field of study dealing with this aim is metabolic flux analysis (MFA). The most important tool for this purpose is the isotope labeling experiment (ILE) (Wiechert 2001). Here shortly spoken, the measurement of isotope labeling distributions in metabolic networks leads to an insight about the intracellular reaction rates. The quantification of flux rates is a close interplay between *in vivo* experimental studies and *in silico* simulation. Here, the intracellular fluxes are estimated by applying an iterative high-performance computation approach (Weitzel et al. 2007, Antoniewicz et al. 2007). The resulting fluxome data can be visualized in metabolic network diagrams.

## Chapter 10. Application Examples

Whereas in Section 10.1 metabolome data was mapped to the metabolite network nodes of the network with omitted reaction nodes, the situation here is reversed: fluxome data correspond to the reaction rates and, therefore, can be visualized by mapping onto the visual appearance of reaction nodes. Another, more intuitive solution for visualizing fluxomes is to change the line strength of the flux edges because their arrow symbolic intuitively indicates some kind of flux velocity of throughput.

The present example is the visualization of fluxome data of the organism *Gluconobacter oxydans*. Fig. 10.2 a) shows a basic reaction model representing the central metabolism of the microorganism. The diagram contains metabolites indicated by rectangles and reactions represented by diamonds. An additional attribute of reactions are their directions. Flux directions may be either irreversible, i.e. the net flux of the reaction mainly proceeds in the direction indicated by the arrowhead, or reversible meaning that flux edges are equipped with both, start and end arrowheads.

The diagram furthermore contains several metabolic pathways shown by highlighting the corresponding sets of reactions and flux edges. Additionally, compartments are shown

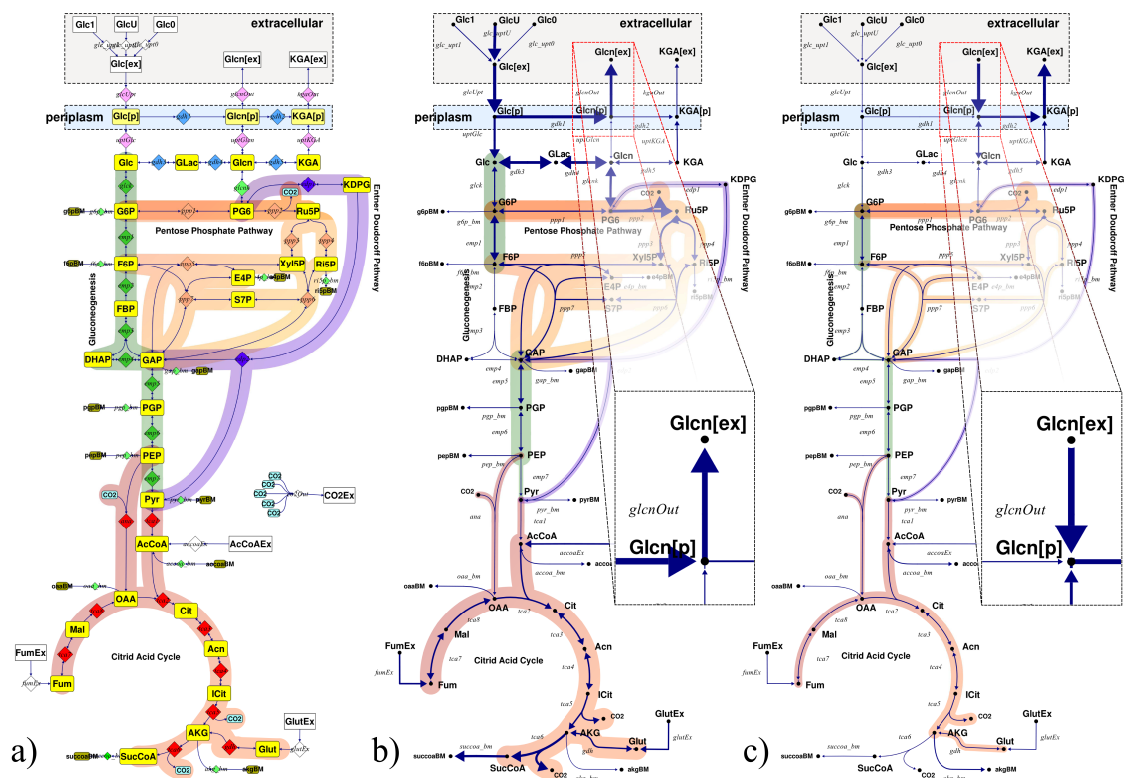


Figure 10.2.: Visualization of flux distributions in a metabolic network – a) shows the diagram without visualized data, b) and c) show flux data from two different phases. The enlarged subsections shows the inversion of a flux direction.

### 10.3. Visualization in the Context of Synthetic Biology

---

in the diagram represented by rectangular areas. The network model uses a compartment in order to mark metabolite pools as to be extracellular, periplasmic or cytosolic (cf. Fig. 10.2 a) top). The *periplasm* is a special compartment located in the cell wall of the *Gluconobacter* bacteria species, separated by the inner and outer cell membrane.

In order to sensibly reduce the plenty of information in the diagram, reaction and metabolite nodes are hidden for the fluxome visualization in Fig. 10.2 b) and c). Metabolites are represented by labeled dots, reactions are represented by joined flux edges.

In Fig. 10.2 b) and c) the diagram is augmented with data from the metabolic flux analysis. Here, the flux strength is mapped to the width of the flux edges: strong lines indicate high reaction conversion rates. This information is also reflected by the local width of the pathway stroke. Hence, a global view of the flux distribution over the whole network is facilitated.

A further visual property used in the example is an inversion flag owned by all reactions (cf. Chapter 3.2.1). If a reaction is set to be inverted, the arrow direction of all its flux edges reverses (cf. Fig. 10.2 b) and c) enlarged subsections). This makes it easy to display a change in the nominal net flux direction without redrawing the edges in the diagram.

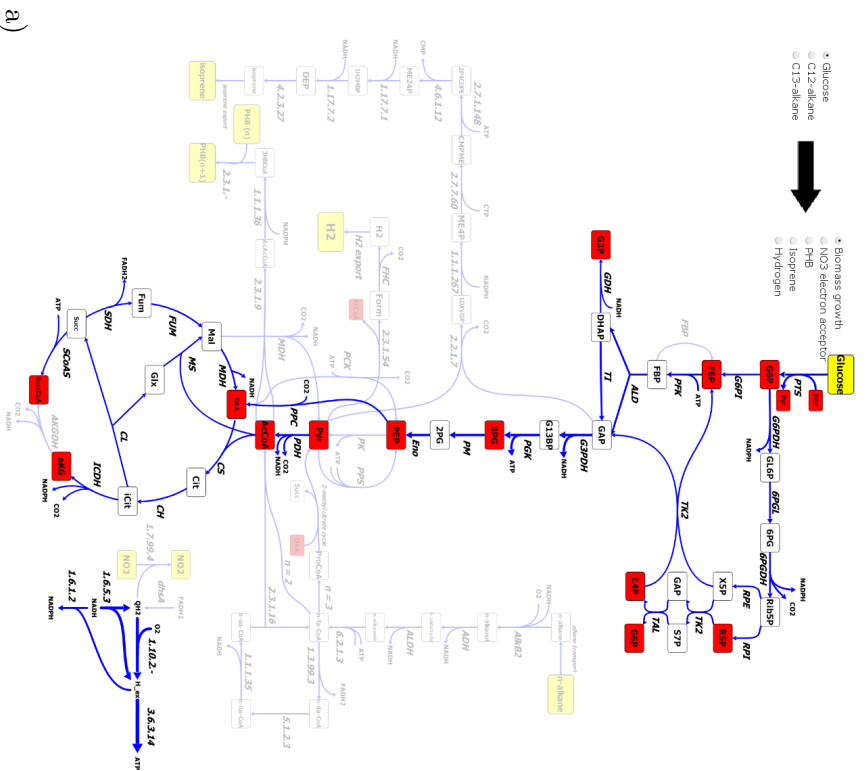
Fig. 10.2 b) shows the flux distribution of the organism at growth on glucose. In this phase, more than 80% of the input substrate is metabolized in the periplasmic space to gluconic acid. When glucose in the medium is (nearly) depleted, the metabolism of *Gluconobacter oxydans* adapts by a metabolic shift towards gluconic acid utilization. Fig. 10.2 c) shows the fluxes in a second phase. In this phase, the organism re-metabolizes the previously produced gluconic acid in order to provide its central metabolism with input substrate.

As it was demonstrated, enrichment of metabolic pathways with visualizations of metabolite concentrations, flux rates or a combination of both can be very useful in understanding the concerted changes of metabolite pools within the cell. The visualization solution consists of 122 lines OVL code developed by a mathematician primarily dealing with MFA.

### 10.3. Visualization in the Context of Synthetic Biology

The Team TU Delft recently published the results of an *in silico* MFA on a synthetic organism designed during the International Genetically Engineered Machines (iGEM) competition 2010. The MFA has been performed by using *CellNetAnalyzer* (Klamt et al. 2007). Aim of the study was the optimization of the metabolic network toward the maximal rates of certain products for different input substrates as computational pre-processing step preceding the experimental design. The results have been visualized by using the network visualization tool *Omix*. The website of the iGEM Team 2010 TU Delft (2010) shows the reference network. The user can choose between different input substrates and product optimizations. By this, the image of the corresponding data visualization is laid over the semi-transparent reference image. Fig. 10.3 on the following page shows two snapshots of this interactive visualization.

Visualization of pathways



Visualization of pathways

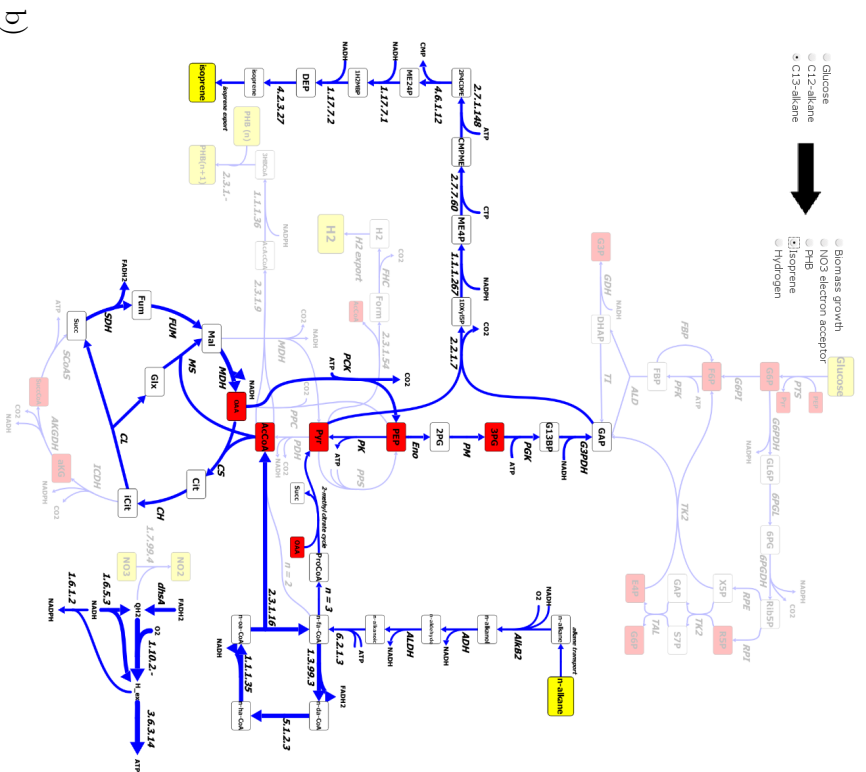


Figure 10.3.: Results of a metabolic flux analysis visualized by using the tool *Omix*. The single network images show the optimal flux distribution for different input substrates and different target products. These different conditions can be interactively explored on the project homepage of the iGEM Team 2010 TU Delft (cf. a and b). The pictures were taken from (iGEM Team 2010 TU Delft 2010).

## 10.4. Visualization of Transcriptome Data

With transcriptome analysis cellular processes are screened under different external conditions. A micro-array based experiment allows for the simultaneous measurement of mRNA levels (the transcriptome) of thousands of genes (Shoemaker and Linsley 2002). Repeated for different conditions, a massive amount of data is generated.

The main objective of the visualization example presented in Fig. 10.4 is to establish a relation between the transcriptome data corresponding to the enzymes in the biochemical system and the metabolic reaction catalyzed by these enzymes. The data used for this example has been taken from Ishii et al. (2007). Here, omics data of 43 experiments is made publicly available. The dataset is related to a reference state and logarithmically scaled. For further information about data processing see Ishii et al. (2007).

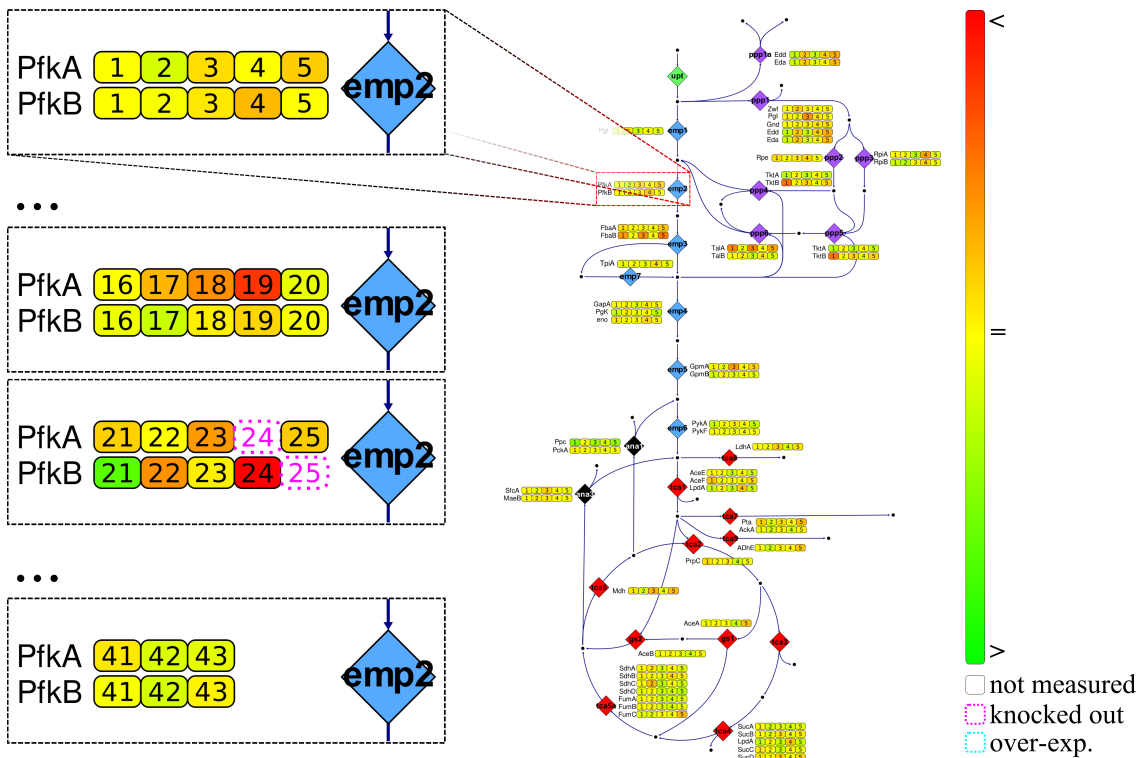


Figure 10.4.: Visualization of a metabolic network of *Escherichia coli*. The change in transcript level is indicated by the expression index for each protein participating in a reaction. The subsections of the network diagram on the left side indicate that the user can switch between 43 different datasets visualized in the network.

The experimental data related to the enzymes is displayed in the network diagram beside the corresponding reactions. All catalyzing enzymes of a reaction are listed in

single rows. In every row, an array of rounded rectangles (SubNode accessory cf. Section 9.1.6) represents a set of experimental data corresponding to the reaction. The index of the displayed experiment is shown in each SubNode item. The user can switch between several datasets as indicated in Fig. 10.4 (left). The number of experiments that are simultaneously visualized is restricted to five in order to avoid an information overload of the diagram.

The transcript level is displayed in a color code indicating a lower (red), equal (yellow) or higher (green) expression in comparison with a specified reference state. This color range is typical for transcriptome visualization (cf. Mlecnik et al. 2005, Neuweger et al. 2009) as a side effect of microarray imaging technologies (Saraiya et al. 2005). If an enzyme has not been measured in an experiment, the corresponding spot is displayed white. If an enzyme is knocked out in an experiment the spot representing the experimental data is marked with a magenta-colored, dotted stroke. Likewise, an over-expression of an enzyme coding gene is marked with a cyan-colored, dotted stroke (not present in Fig. 10.4). In Fig. 10.4, an enlarged subsection of the diagram shows the reaction of *phosphofructokinase* (*Pfk*) using the two enzymes *Pfk-A* and *Pfk-B*. Both of them contribute to the overall activity in the cell (Daldal 1984, Vinopal et al. 1975). In order to quickly identify the minima and maxima in the dataset, the user can select to display the indexes of the experiments with the lowest and highest transcript level, respectively. This min/max visualization is shown in Fig. 10.5 on the next page.

The described way of displaying transcriptome data is typically useful for gene knock-out and over-expression studies, because the experimenter can see direct effects on the transcript level in the network context. The visualization solution has been created by a graduate biotechnologist and consists of 235 lines OVL source code.

### 10.5. Animation of a Pulse Experiment

Previous sections have shown the visualization of static data concerning one omics level. But in systems biology, investigations, whether in experiment or in simulation, result in a plenty of data. This data usually spreads over various omics domains and is particularly time-dependent. The current example shows the power of OVL in animating time-series of multi-omics data in a metabolic network diagram.

To holistically understand the metabolism of cells, dynamic modeling of cellular processes has become an important task in systems biology. Dynamic models describe the state of each reacting metabolite and the enzymatic reaction rates in a time-dependent manner. To set up such models, appropriate (mechanistic or approximate) rate laws have to be assigned to each reaction within the network. Hence, dynamic models usually contain a large number of parameters like the reaction rates, Michaelis constants, limiting rates or constants describing the influence of inhibitors or activators (Chassagnole et al. 2002). Once a model is set up it provides an instance for making predictions how the cell responds to environmental changes or perturbations. Visual screening of the systems'

## 10.5. Animation of a Pulse Experiment

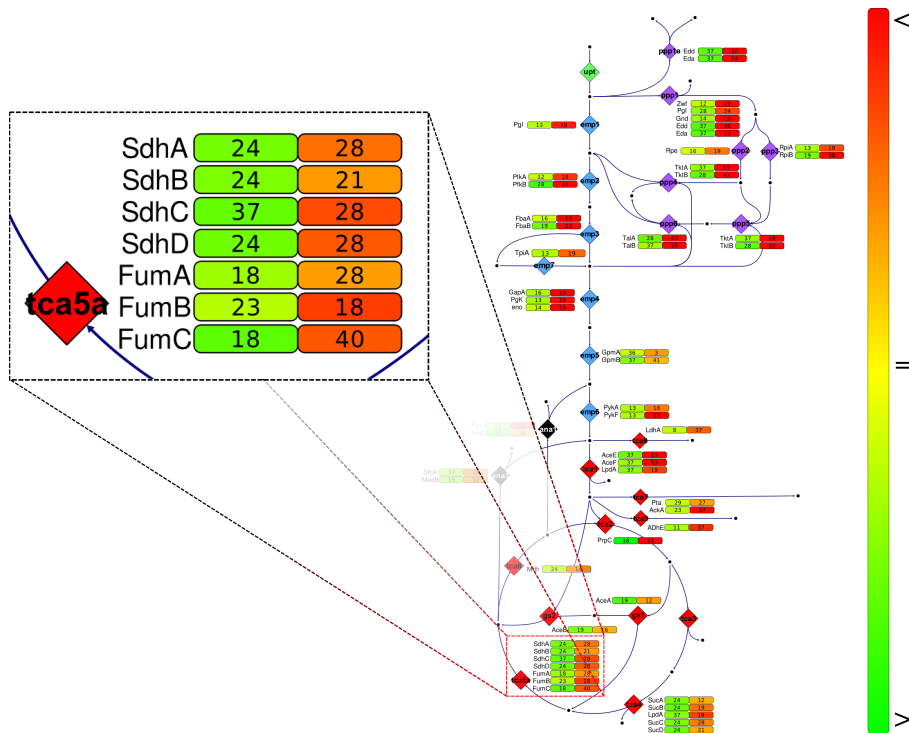


Figure 10.5.: Visualization of minimal and maximal transcript levels.

characteristics by variation of parameter values or the choice of alternative rate equations offer insights to complex metabolic control mechanisms of metabolic fluxes.

The diagram in Fig. 10.6 on the following page shows the glycolysis and the pentose phosphate pathway of *Escherichia coli*. For details about the underlying rate laws and kinetic parameters see Chassagnole et al. (2002). The model is validated with measured metabolite concentrations at transient conditions by performing a glucose stimulus experiment. Here, the concentration of an extracellular metabolite, typically glucose, is rapidly increased in the culture and the response in the cells in terms of intracellular metabolite concentrations is measured.

The diagram shown in Fig. 10.6 visualizes metabolite concentrations, reaction rates and regulatory strength in a metabolic network (Noack et al. 2007). Metabolite concentrations are mapped to the color fill level of the metabolite nodes. Strength of the fluxes is shown by the fill level of the reaction nodes which changes pie-chart-like. Furthermore, the flux strength is indicated by the line width of the flux edges (cf. Section 10.2). In analogy to the reaction rates, the regulatory strength is mapped to the line width of the effector edges.

The simulated model response to the step increase of the extracellular glucose concentration at time point 0.8 s is shown as animation. Temporal changes of all state variables in the network can be followed over time in an automated or step-by-step manner (cf.

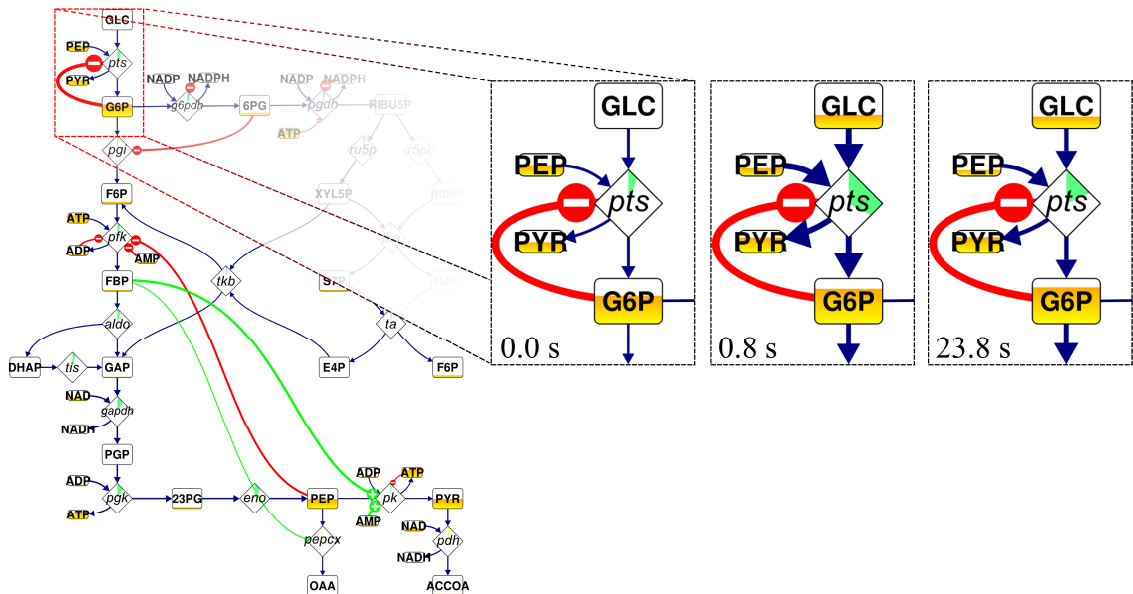


Figure 10.6.: Visualization of dynamic changes in the network by animation. The network starts in a steady state (cf. image at 0.0 s) until it is triggered by a substrate pulse. The pulse causes a dynamic response of the concentrations, fluxes and regulatory influences all over the network (cf. images at time points 0.8 s and 23.8 s).

Fig. 10.6). As a pronounced feature the strong non-linear feedback-feedforward link between the *phosphotransferase system* (PTS) and the glycolytic metabolite concentrations of phosphoenolpyruvic acid (PEP) and pyruvic acid (PYR) can be regarded.

Since the visualization tool allows hiding information in the network diagram, the visualization can be displayed in a higher abstraction level. As an example in Fig. 10.7 on the next page everything in the network is hidden except for the pathways. The pathway shape can also be used for information visualization because reactions can have influence on the pathway's local appearance. The contour around a reaction can be thinned (cf. Fig. 10.7). By this, the visualization of information connected with metabolic reactions can be performed in a network-wide, abstract view. This feature is an outstanding and novel visualization method provided by the network symbolism and the levels of detail combined with the OVL scripting language introduced in this work. In order to see the global changes in the network in combination with the regulatory influences, the effector edges are additionally shown in Fig. 10.7. The OVL script performing the here presented visualization consists of 126 lines and is introduced as expanded, commented example program in the *OVL Technical Manual* (Droste 2008b).



## 10.6. Visualization of Isotopic Mass Distributions

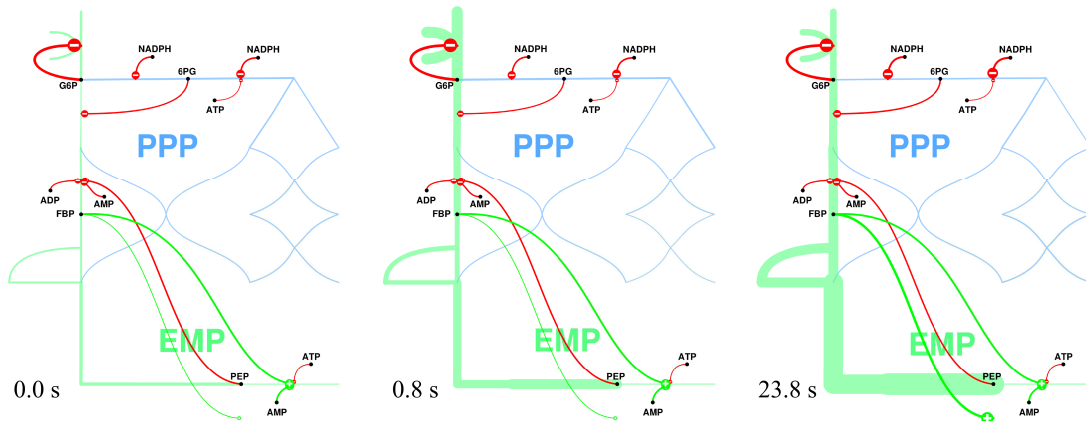


Figure 10.7.: View of the data from Fig. 10.6 at a higher abstraction level.

## 10.6. Visualization of Isotopic Mass Distributions

In carbon labeling experiments, the natural  $^{12}\text{C}$  carbon atoms in the molecules of the input substrate are specifically substituted with the heavy  $^{13}\text{C}$  isotope and the distribution of the isotopes over the whole metabolism is observed. By this method, knowledge about the intracellular fluxes is gained (Klapa et al. 1999). The isotope enrichment leads to distinguishable higher masses of the single metabolites measurable by mass spectrometry. A metabolite with  $n$  carbon atoms can appear with  $n + 1$  different peaks in a mass spectrum. These peaks represent different labeling states of the metabolite. Consequently, the concentration of a metabolite can be subdivided into the concentrations of these labeling states.

The present visualization example extends the visualization principle described in Section 10.5 with ratios of isotope labeling states. Therefore, each metabolite is equipped with an array of `SubNode` accessories. The single graphical items appear as circles beside their corresponding metabolite symbol. Each `SubNode` in the array represents one of the  $n + 1$  different masses of the metabolite. The ratio of each labeling state in the overall-concentration of the metabolite is visualized by a pie-chart-like change of the nodes' fill level.

A small network is depicted in Fig. 10.8 on the following page containing several metabolites and reactions composed to a small example network. The network starts with an input at the top and ends with two outputs bottom left. Additionally this network contains three inhibiting effector edges. In analogy to the example of Section 10.5, metabolite concentrations are mapped to the color fill level of the metabolite nodes, flux rates are indicated by the width of the flux edges and the strength of regulatory influence is displayed by the width of the effector edge.

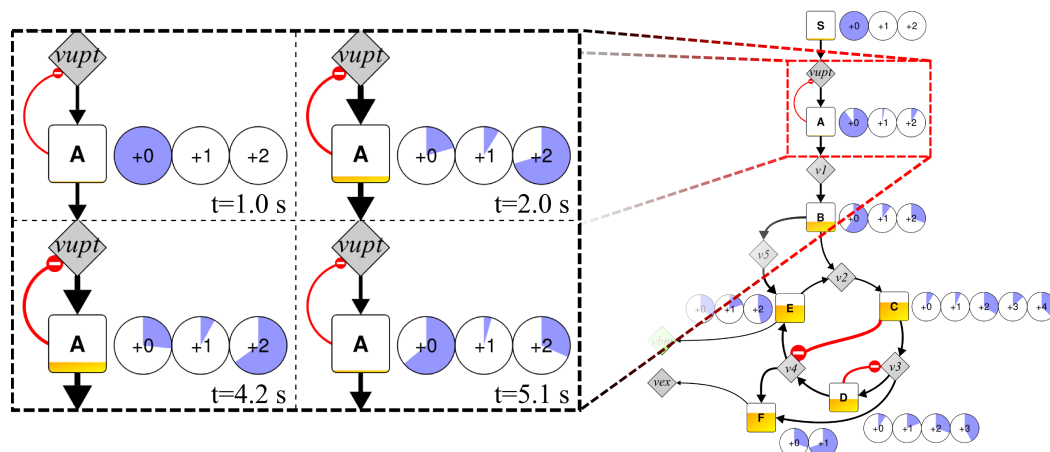


Figure 10.8.: Visualization of isotopic distributions in a network

Fig. 10.8 is a snapshot of an animation that visualizes dynamic changes of those four different kinds of omics data regarding the network. Starting the animation, the dynamics of the system stimulated with a labeling pulse from the top of the network can be tracked. This pulse causes a network wide change of substance concentrations, flux rates, regulatory influences and concentration rates of labeling states. The enlarged subsections on the right show the local changes at one metabolite at four different time steps. The example visualizes simulated data. The underlying model is a simplified example network. The visualization illustrates a high aim of the metabolic flux analysis: to be able to predict the systemic behavior under metabolically and isotopically non-stationary conditions. The OVL script performing the visualization has been developed by a bioprocess engineer based upon the given example in the *OVL Technical Manual* (Droste 2008b) consisting of 151 lines.

## 10.7. Visualization of Enzyme Kinetics

The last example presented here visualizes the scientific insights into the kinetic behavior of the alcohol dehydrogenase (ADH) from *Lactobacillus brevis* published by Schroer et al. (2009). The enzyme catalyzes the reduction of methyl acetoacetate (MAA) in a substrate-coupled cofactor regeneration approach by oxidation of isopropanol as shown in Fig. 10.9 on the next page. Schroer et al. (2009) introduce a comprehensive mathematical model of the enzyme kinetics including all quantified model parameters.

The visualization example presents the results from (Schroer et al. 2009) by using different visualization techniques that have been introduced in previous chapters:

- The network diagram shown in Fig. 10.9 is displayed with hidden reaction and metabolite nodes (cf. Chapter 3.4). Instead, only the labels of the network components are visible.

## 10.7. Visualization of Enzyme Kinetics

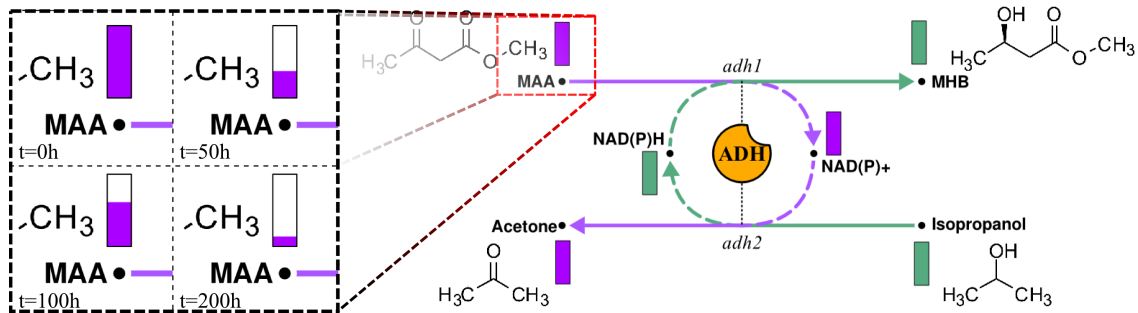


Figure 10.9.: Visualization of enzyme kinetics by animation.

- The orange symbol in the center of the diagram is a custom node representing the enzyme connected to the catalyzed reactions by custom edges (cf. Chapter 3.2.4).
- The main metabolites of the ADH reactions are annotated with their respective chemical structure. The trained eye can immediately see that a hydrogen atom is transported from the isopropanol via NAD(P)H to the methyl 3-hydroxybutyrate (MHB) molecule.
- The enzyme kinetics is visualized by animating the concentration changes of the participating metabolites and cofactors (cf. Fig. 10.9 enlarged image section).
- The constants and kinetic parameters in relation to the single components of the diagram can be interactively inspected by Visualization on Demand (VoD) (cf. Chapter 9.3). The metabolites reveal their kinetic parameters ( $k_{pi}$ ,  $k_{si}$ ,  $k_m$  cf. Schroer et al. 2009) as well as a function plot showing the concentration change in time as shown in Fig. 10.10. When the enzyme symbol is touched by mouse cursor the weight of the enzyme molecule is displayed. The reactions show their maximal velocity on demand. All values are presented with standard deviation.

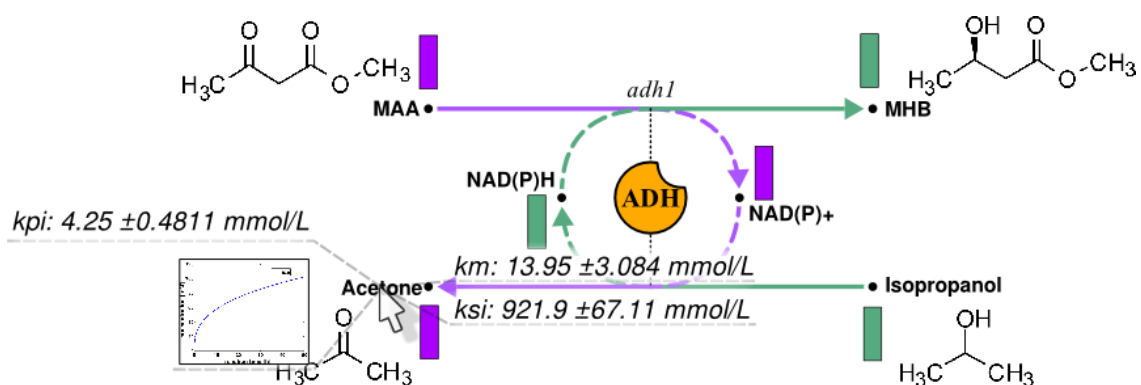


Figure 10.10.: Interactive visualization of enzyme parameters on demand.

## Chapter 10. Application Examples

---

By default, the visualization tool *Omix* is not able to display chemical structures in the network diagram. In order to be adaptable to this specific and many other requirements the software allows to be extended by plug-ins as introduced in Part IV.

# Omix Feature Sheet

Data Visualization Features	Page
• Script-based visualization with the <i>Omix Visualization Language</i> (OVL)**	71
• Syntactically derived from Java	72
• Easy-written user manual imparting skills in OVL development*	72
• Full documented in a technical manual*	72
• Extending existing type definitions by the keyword <b>extend</b> **	72
• Programmable access to all components of the network drawing including their relations	73
• Programmable access to the visual properties of network components**	73
• Equipping network components with custom properties	73
• Simplified access by assignments	73
• The same type system as Java	74
• Simplified programming of interactive components**	74
• Novel syntax for accessories that is short, exact and avoids programming errors**	75
• Simplified programming of additional information carriers**	78
• Arrays of accessories with flexible size**	79
• Meta-information about programming constructs*	79
• Exchanging OVL programs between different documents	80
• Complete development environment for OVL programs*	80
• OVL debugger*	80
• Availability of the complete Java runtime library in OVL	80
• Extensive, full documented data visualization library for OVL*	80
• Interoperability between Java and OVL code**	81
• <i>Visualization on Demand</i> for the interactive exploration of hierarchically organized data in networks**	82
• Defining own data types for the custom organization of data	84
• Editor dialog windows for the dynamic instantiation of arbitrary data types*	85
• OVL code is easy to understand and to adapt	85
• Many existing OVL solutions	89
• OVL allows data visualization by mapping on visual properties	89
• OVL allows data visualization by annotating network components	95
• OVL allows visualization of time-dependent data*	96
• Information visualization on the pathway shape**	98

\*particular feature of *Omix*, i.e. not novel in general but hardly realized by other tools for visualization in the context of biochemical networks.

\*\*novel approach introduced in this thesis.

# Omix Feature Sheet

Plug-In Features	Page
• Omix is extensible by plug-ins through the <i>Omix API</i> . . . . .	107
• Manual about plug-in development* . . . . .	108
• Full documented API* . . . . .	108
• Plug-in manager module for handling the interplay between plug-ins and the main application . . . . .	109
• Java interfaces for the mutual abstraction of application and plug-in functionality . . . . .	109
• Plug-in manifest describing plug-in functionality for the registration at the plug-in manager . . . . .	110
• Many possibilities to extend the application . . . . .	111
• Import and export of SBML . . . . .	113
• Export of network stoichiometries to CSV and <i>Matlab</i> * . . . . .	113
• Export of kinetic models to Modelica code* . . . . .	113
• Loading metabolic pathways from KEGG . . . . .	114
• Visualization of chemical structures . . . . .	114
• <i>13 C FLUX 2</i> modeling with <i>Omix</i> ** . . . . .	116
• Import and export of FML files** . . . . .	117
• Graphical front end for the <i>13 C FLUX 2</i> simulator framework** . . . . .	117
• Import of <i>13 C FLUX 2</i> simulation results in OVL** . . . . .	117
• Interactive exploring and visualizing elementary flux modes* . . . . .	117
• Performing and visualizing flux balance analyses* . . . . .	117
• Interactively exploring and visualizing free fluxes* . . . . .	117
• 3D visualization of isotope labeling networks** . . . . .	119
• 3D visualization of all atom transitions of a metabolic network** . . . . .	119
• Visual decomposition of isotope labeling networks** . . . . .	121
• Visual path tracing of labeling states** . . . . .	121
• OVL driven 3D visualization of network thermodynamics** . . . . .	122
• Accessing SSH and SMB shares . . . . .	124
• Data import from <i>Excel</i> and <i>OpenOffice.org</i> spreadsheet formats . . . . .	124
• Export of animations to <i>Flash</i> files* . . . . .	124
• Export of AVI and MPEG videos* . . . . .	124

\*particular feature of *Omix*, i.e. not novel in general but hardly realized by other tools for visualization in the context of biochemical networks.

\*\*novel approach introduced in this thesis.

**Part IV.**

**Extensibility**





# Chapter 11.

## Introduction to Extensibility

*“Plugins are an important way for advanced users to customize and extend an application”* (Suderman and Hallett 2007)

### 11.1. Motivation

The visualization tool *Omix* faces the challenge to be adaptable to future requirements in systems biology not only by providing a script-driven visualization approach. The range of features of the software can be extended by plug-ins as illustrated in Fig. 11.1. Therefore, the tool is equipped with an extension interface, also called application programming interface (API). Today, extensibility is state-of-the-art in software development. Plug-ins are known from web browsers, media player software, graphics and office programs. Even in the area of visualization software for biological networks extensibility by plug-ins is a widespread concept for keeping the applications flexible. Examples are *CytoScape* (Shannon et al. 2003), *CellDesigner* (Funahashi et al. 2003), *VisANT* (Hu et al. 2005), *ProViz* (Iragne et al. 2005) and *Vanted* (Junker et al. 2006). Extensibility is also realized by the approach introduced in this work.

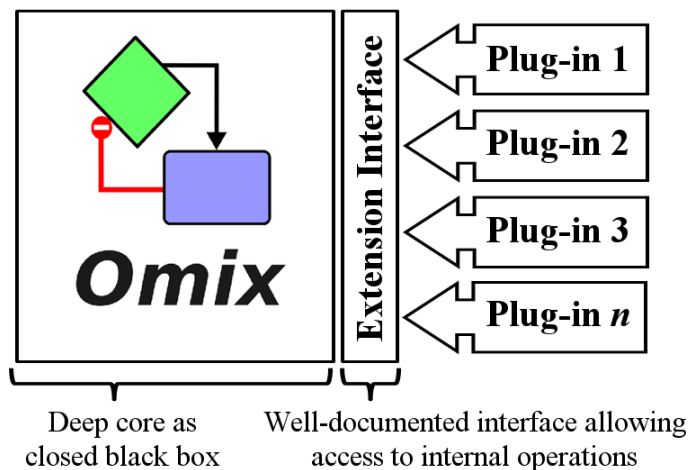


Figure 11.1.: Relation between the visualization tool *Omix* and plug-ins.

Fig. 11.1 on the preceding page shows a principle scheme of the relation between the software *Omix* and its plug-ins. *Omix* is a closed source project. The internal architecture is not published and can neither be accessed nor changed by third person. By using the extension interface, further called *Omix API*, specialized features useful in different areas of life sciences can be added to the functional range of the tool. The *Omix API* is a well documented interface allowing the communication between *Omix* and the plug-in module. In this way, a plug-in can access internal operations and can offer the implemented features to *Omix*. The *Omix Plug-in Development Manual* (Droste 2011) introduces how to develop *Omix* plug-ins in detail. Furthermore, a reference documentation of all definitions of the *Omix API* is publicly available (Droste 2010).

### 11.2. Differentiation to OVL

This section aims at clarifying the difference between OVL and a plug-in. OVL is a scripting language whereas plug-ins are to be developed in Java. An OVL script runs in a network diagram. It is textual and can be edited as part of the document. Unlike OVL, a plug-in extends the entire visualization tool. Usually, a plug-in is distributed in binary form. The user can neither inspect nor change the plug-in program. The plug-in code is loaded at startup time and becomes an unchangeable part of the software.

The OVL scripting language is suitable for solving simple tasks of visualization inside of the diagram whereas plug-ins can realize complex features. OVL scripts can be created from the scratch by the user. Plug-ins can only be installed by the user. They are created by programming experts and software developers. To a certain extend, OVL scripts can be compared with so called “macros” in the popular word processing tools.

# Chapter 12.

## The Omix API

### 12.1. Interface Concept

The core of the extensibility realized in *Omix* is a plug-in manager. This software component loads all plug-ins at startup and integrates their functionality into the main application. The *Omix* API is a set of Java interface definitions. Such an interface is an abstraction of the internal realization of a feature. The communication between the main application and the plug-ins is exclusively realized via Java interfaces. A plug-in realizes functionality that must be registered inside of the application in order to be available. This is done by defining the feature by using the interfaces provided by the *Omix* API. On the other side, the application must allow limited access to internal data and operations for the plug-ins. This is provided as service for the plug-ins via interface definitions that allow access on an abstract level.

Fig. 12.1 on the next page illustrates the principles behind the extensibility of *Omix* in a schematic diagram. The interaction between the application and a database plug-in is depicted in a simplified manner. The fictive plug-in sketched here provides a further menu entry on the menubar. When the user selects the menu entry, a database can interactively be searched and a network can be selected by the user. Thereafter, *Omix* loads the network from the database as new document.

- (1) At startup, the plug-in is instantiated by the plug-in manager.
- (2) The plug-in has permissions to insert a further menu entry in the menu bar via the corresponding application interface. This menu entry is connected with one of the internal functions of the plug-in.
- (3) At user interaction, the menu entry calls the feature that realizes the interactive database search.
- (4) After finishing the database search a network is available to be loaded as new *Omix* diagram. Therefore, the plug-in prompts the application to start a document loading procedure.
- (5) The document loading service again calls the plug-in to assemble the network stoichiometry.

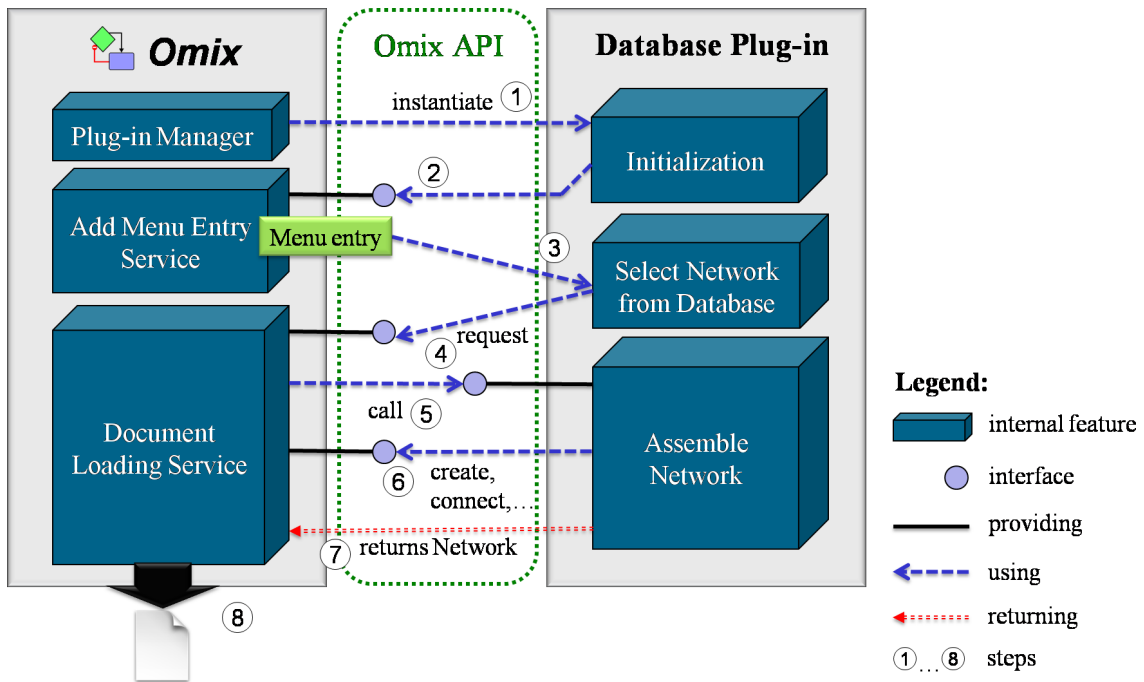


Figure 12.1.: Communication between *Omix* and a plug-in.

- (6) Only in this phase of the process, the plug-in gets access to the internal operations of *Omix* that instantiate new nodes and edges and assemble them to a network.
- (7) After assembling the network upon the user selected database content, the plug-in returns the results to the application.
- (8) *Omix* starts a new document with the selected network as initial content.

In principle, the extensibility of *Omix* is realized in this way. The *Omix* API contains over 80 different interface definitions for distinct types of plug-ins as introduced in the next section. A plug-in is an archive file containing a manifest and program code. The manifest describes the plug-in in detail. This includes the name of the plug-in and, particularly, the implemented interfaces. Based on the information from the manifest, the plug-in functionality can be registered at the plug-in manager.

## 12.2. Features of the Omix API

The *Omix* API allows to develop plug-ins for different purposes. Each plug-in type has limited access to a specific set of core operations and a specific life time during the runtime of the application. A plug-in can basically implement multiple or even all of the available types simultaneously. The plug-ins types can be classified as follows:

### Application Plug-in

An application plug-in is active for the complete runtime of *Omix*. It can, for instance, add items to the menubar and insert toolbars on the main window. In this way, *Omix* can be equipped with interactive components that provide plug-in functionalities. An application plug-in can request access to the currently edited document which includes the complete inherent network structure as well as the document embedded OVL script. Furthermore, a new network can be assembled and submitted to be loaded as new document. This case has been exemplified in Section 12.1.

### Document Plug-in

A document plug-in has the same privileges as the application plug-in but is document bounded, i.e. it has to be explicitly activated on an open network document.

### OVL Scripts

Plug-ins can be equipped with OVL code (cf. Chapter 9). When the plug-in is activated on a network document, the OVL code runs parallel to the user defined script in the network.

### OVL Libraries

Plug-ins can extend *Omix* with any libraries to be used in custom OVL scripts.

### Network Import/Export File Filter

A file filter registers a certain file format suffix at *Omix*. In an export operation a file filter is called to traverse the internal network structure and generate the corresponding constructs in the target file. During an import operation the file filter plug-in gets the privilege to compose a network and commit it as new document.

### Image export Filter

A plug-in can define a paint engine for certain image or video formats that can be selected for image and animation export.

### Data Type Management

*Omix* plug-ins can introduce new data types, for instance, concerning simulator specific network parameters. The editing and visualization of plug-in specific data is handled by a data type manager.

### Plug-in Configuration

Plug-ins can register a configuration window at *Omix* appearing as subcomponent in the configuration manager window of the application. By this, a plug-in can allow to be configured by the user. A plug-in configuration manager has the permission to store its specific settings in the *Omix* configuration registry.

### **Communication Protocols**

*Omix* can be extended by communication plug-ins that enable the software to handle file management via particular Internet protocols. This includes security handling and data streaming.

### **Internationalization**

Plug-ins can provide multiple languages for the graphical user interface realized inside of the plug-ins. Furthermore, additional languages can be provided for the entire application.

# Chapter 13.

## Existing Plug-In Solutions

Multiple plug-ins for *Omix* have been realized dealing with current issues in systems biology research. This chapter gives an overview of the available plug-ins in order to demonstrate the use of the *Omix* API.

### 13.1. Compatibility and External Resources

As figured out by Suderman and Hallett (2007) and Pavlopoulos et al. (2008) compatibility with other software in the context of biochemical networks as well as the utilization of available data sources is an important prerequisite for a visualization tool like *Omix*. This Section introduces a set of plug-ins that meet this requirement.

#### 13.1.1. Model Describing Formats

A widespread data exchange format for biochemical networks is the Systems Biology Markup Language (SBML) (Hucka et al. 2003). Consequently, a file format filter plug-in has been developed that allows the export and import of network stoichiometries from SBML files. Another plug-in allows exporting a stoichiometry matrix representing the network in *Omix* to *Matlab* (Attaway 2009, Sharma and Martin 2009) and CSV spreadsheets.

Furthermore, a plug-in has been created for generating Modelica code from a given network model (Tiller 2001). The plug-in requires an existing Modelica library implementing reaction kinetics (Nilsson and Fritzson 2005). The user can select the kinetic laws for each reaction in *Omix* and specify model parameters. The generated code instantiates, parametrizes and connects programming constructs from the underlying kinetics library. After export, the generated code can be simulated in respective tools like *Dymola* (Elmqvist 1978) or *OpenModelica* (Fritzson et al. 2006).

#### 13.1.2. Database Connectivity

A large number of databases exist storing information about biochemical networks aplenty. (Cary et al. 2011) list over 300 publicly available resources. Examples are *BRENDA* (Schomburg et al. 2002), *BioCyc* (Karp et al. 2005), *Gene Ontology* (Ashburner et al. 2000) or *WikiPathways* (Pico et al. 2008).

## Chapter 13. Existing Plug-In Solutions

In order to make such resources available for *Omix* a plug-in has been developed connecting the network editor with the *Kyoto Encyclopedia of Genes and Genomes (KEGG)* database system (Kanehisa and Goto 2000). Hereby, the huge amount of information about the metabolic pathways of hundredth of organisms provided by *KEGG* is available in *Omix*. The big advantages of this Japanese database is, it is connected to many other related databases and it provides a programming interface, the *KEGG API*. Based upon the *KEGG API*, software can directly access the database content. The *KEGG* plug-in for *Omix* enables the user to search the database for certain information and download single reactions, pathways or even entire organisms directly into *Omix*. Fig. 13.1 shows the *KEGG* import dialog window.

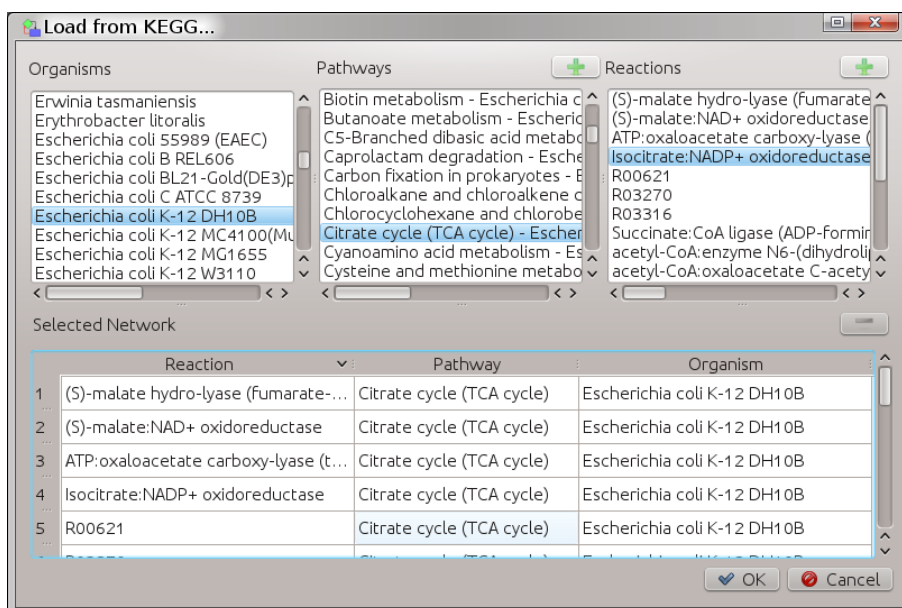


Figure 13.1.: Dialog window for loading networks from the *KEGG* databases.

### 13.2. Visualization of Chemical Structures

Biologists need a biologically relevant representation of biochemical networks. In various cases the symbolism of *Omix* networks consisting of primitive shapes like rectangles, circles and diamonds might not meet the users' requirements. A rather detailed depiction of metabolites is desired. Metabolites represent the biochemical compounds involved in the metabolic reactions and a compound consists of a distinct molecular structure.

In order to allow molecular structures to be displayed in *Omix* network diagrams the Mol plug-in has been created; "Mol" because it bases on the Mol data format for describing chemical structures (Dalby et al. 1992). The plug-in equips each metabolite in a diagram with a property holding the chemical structure. In combination with the *KEGG* plug-



## 13.3. Metabolic Flux Analysis Work Flow

in (cf. Section 13.1.2) chemical structures are automatically downloaded when the user generates a network from the database. Otherwise, the structural formulas can be selected from available compounds in the *KEGG* databases or even drawn manually (see Fig. 13.2 a). Subsequently, the chemical structures can be inspected in the VoD mode (cf. Chapter 9.3) or can be visualized directly in the network diagram as shown in Fig. 13.2 b).

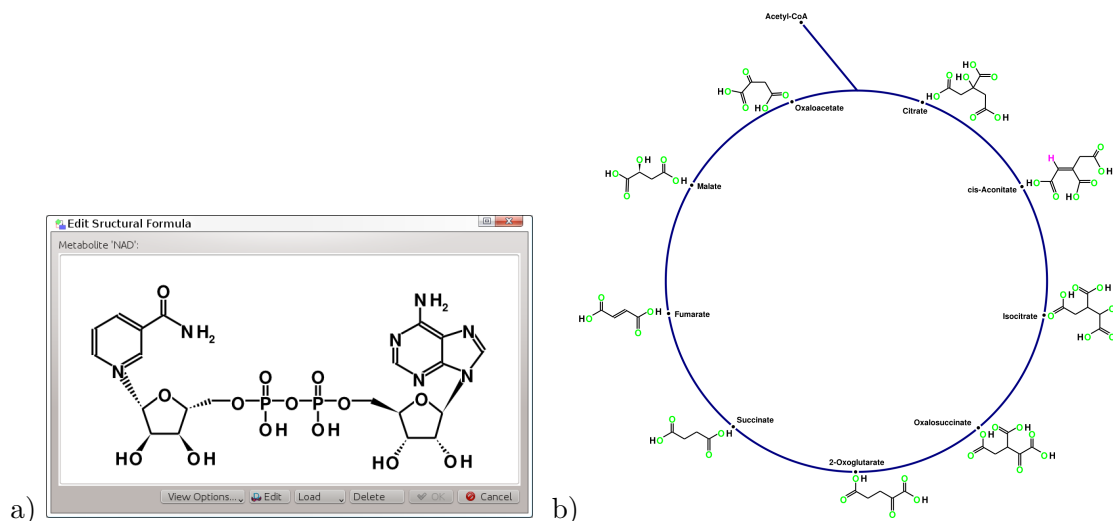


Figure 13.2.: Editor for chemical structures utilizing *KegDraw* (Kanehisa et al. 2005) (a). Visualization of chemical structures in a metabolic network (b).

## 13.3. Metabolic Flux Analysis Work Flow

### 13.3.1. Background

Metabolic flux analysis (MFA) aims at identification and quantification of intracellular fluxes (cf. Chapter 10.2). The flux rates in a cell cannot directly be measured. By labeling the input substrates with  $^{13}\text{C}$  isotopes and by measuring the labeling enrichment of the metabolites all over the network with highly sensitive mass spectrometry devices (Szyperski 1998, Christensen and Nielsen 1999) the intracellular flux distribution can be estimated (Wiechert 2001). Therefore, computer simulation is necessary. *13C FLUX 2* is such a simulator framework for the quantification of metabolic fluxes from experimental measurement data with high-performance computational methods (Weitzel 2009). The framework consists of about twenty different command-line applications for simulation, exploration, parameter fitting, statistical analysis and other purposes (Dalman et al. 2010a).

Simulation with *13C FLUX 2* bases on models that do not only include the stoichiometric information about a metabolic network. The number of carbon atoms for each involved metabolite as well as the atom transitions of all reactions are also required. A

reaction's atom transition describes the way the single carbon atoms are exchanged between the reaction's substrates to the products. *13C FLUX 2* has an own XML-based file format called *FML* carrying the network stoichiometry, atom numbers and transitions and specific information necessary for a simulation run like start conditions, configurations of the network, measurement data, simulation preferences and others. Results of most *13C FLUX 2* simulation tools are stored in an own XML-based file format called *FWDSim* (Dalman et al. 2010b).

Basically, FML files can be completely developed by the modeler. However, since the visualization of simulation and experimental results requires a diagram of the network model it makes sense to model in a graphical manner. Another pragmatic reason for graphical modeling is: graphical modeling is much more intuitive and fail-safe than editing a metabolic network model in a textual manner. A graphically modeled network structure is easy to grasp because of the immediate visual representation. Hence, errors in the model can be discovered very fast. Several *Omix* plug-ins have been developed dealing with the pre- and post-processing of simulations with *13C FLUX 2*.

### 13.3.2. 13C FLUX 2 Modeling Plug-In

The *13C FLUX 2* modeling plug-in extends *Omix* with the ability to edit FML models for *13C FLUX 2* including all related information. The plug-in equips all metabolites in the network with a "number of atoms" property and the reactions consequently with an "atom transition" property. The network is equipped with properties for the constraints, the simulation configuration, the network input and other relevant information. The plug-in offers several dialog windows for graphical editing of these simulation parameters. Fig. 13.3 shows the atom transition dialog window. The molecules are abstracted by a "pea pod"-like item. The atoms are represented by different colored, numbered circles. By drag and drop, every single substrate atom can be connected to a product atom. The concrete chemical structures are not shown in the atom transition editor because the simulator uses the same abstract representation of molecules and atom transition.

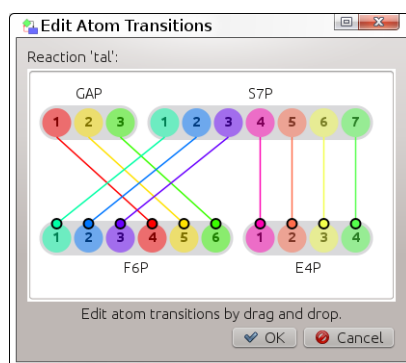


Figure 13.3.: Editor dialog for atom transitions.

For now, the abstract array of atoms cannot automatically be translated to the atoms in a molecule because of a lack of standardized atom numbering in chemical structures. Closing this gap is a possible future task.

After editing all atom transitions of the model, a validation utility provided by the *13C FLUX 2* modeling plug-in warns against inconsistent parameter settings and an invalid network topology. After editing the model, a plug-in inherent file filter allows to export the model from *Omix* to an FML file as simulator input. Likewise, existing FML files can be imported into *Omix* by the plug-in.

### 13.3.3. *13C FLUX 2* Launcher Plug-In

As illustrated in (Dalman et al. 2010a), the Unix-based applications of the *13C FLUX 2* simulation framework have been wrapped by Java web services and are hereby available for each programs on each platform via local network of the research institute. This situation allows to run the simulator tools directly from inside the *Omix* application. Therefore, the *13C FLUX 2* launcher plug-in extends the *Omix* menubar with entries for starting the single simulator tools remotely. When a simulator is started, the network structure of the *Omix* document including all specified simulation parameters is converted to a FML network description and sent to the web service that executes the simulation. After finishing, the results are returned as data stream in FWDSim format. For now, the import of measurement data into the FML specification is not realized. This is an important future project in order to complete the MFA work flow performed with *Omix*.

### 13.3.4. FWDSim Import Plug-In

In order to make the simulation results available for visualization in the network diagrams the FWDSim import plug-in realizes the loading of *13C FLUX 2* results into *Omix*. This is realized by providing a set of classes that represent the contents of FWDSim files for OVL. By this, the *Omix* user can develop own solutions in OVL for the visualization of flux rates and other simulation results of *13C FLUX 2*.

### 13.3.5. Network Analysis Plug-Ins

Further plug-ins have been developed dealing with metabolic network analysis, for instance, searching and visualizing elementary flux modes (Terzer and Stelling 2008), performing flux balance analysis (Orth et al. 2010) and determining free fluxes (Weitzel et al. 2007). The mathematical background of these plug-ins cannot be discussed here for the sake of brevity. However, a short description of the Free Fluxes plug-in demonstrates the value of visualization-aided network analysis.

In metabolic flux analysis all fluxes of a network can be computed when a certain number of fluxes are known. These known fluxes are called *free fluxes* while all others are called *dependent* (Weitzel et al. 2007). The Free Fluxes plug-in equips every reaction with a button and a small, yellow colored spot as depicted in Fig. 13.4 on the following page. By pressing the button, the reaction is chosen to be a free flux. This is immediately indicated

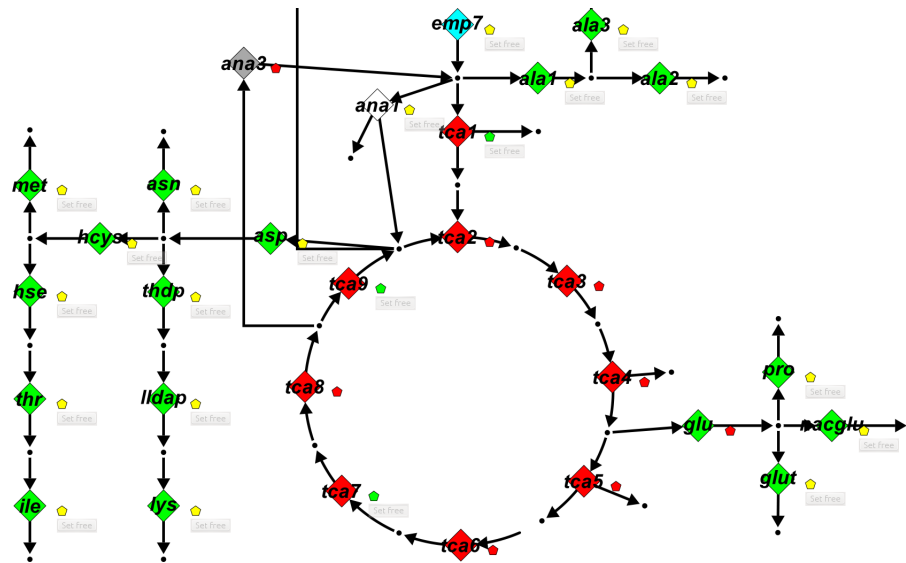


Figure 13.4.: Interactive determination and visualization of free fluxes.

by the small spot icon that changes to green. By selecting a free flux, consequently certain other fluxes become dependent. This is internally computed and the corresponding reactions are labeled with a red spot while their button disappears. In this way, the dependencies in a network can be visually explored. The network analysis plug-ins are well-suited to teach the underlying mathematical analysis methods with an explorative element.

## 13.4. 3D Visualization of Isotope Labeling Networks

### 13.4.1. Foundations

In Section 13.3 the carbon labeling approach has been shortly introduced as an important tool in the MFA.\* Isotope labeling experiments (ILEs) are an important tool to determine intracellular fluxes in a living organism. In this context not the metabolite network has to be visualized but a related network that shows the flow of  $^{13}\text{C}$  labeled carbon atoms or groups of such carbon atoms. Isotope labeling networks (ILNs) associated to metabolic networks have an extremely high dimension. Details on these networks can be found in (Nöh et al. 2008, Weitzel 2009). Essentially, they describe the fate of isotope labeled substances metabolized by the cell. To account for the fate of each differently labeled molecule each metabolite pool has to be divided into so-called *cumomers* (different labeled/unlabeled states).

\*Excerpts of this section originate from (Droste et al. 2008b;a).

## 13.4. 3D Visualization of Isotope Labeling Networks

If a metabolite has  $n$  atoms which are accessible by isotope labeling then there are  $2^n$  different cumomers associated to this compound. These cumomers are connected by cumomer reactions which are analogous to the underlying reactions in the metabolic network. Interestingly, it could be proven in (Weitzel 2009) that cumomer networks can always be decomposed into smaller subnetworks. The exploitation of this decomposition has a strong impact on the performance of simulation algorithms, measurement evaluation or experimental design (Wiechert et al. 1999). The cumomer network decomposes into different levels. The levels are defined by the number of labeled atoms of the cumomers that belong to the level. The lowest level (level 0) corresponds to the metabolic network. The level 1 is equivalent to the atom transition network. Any higher level describes the way certain molecule fragments are transported through the network. Each level is furthermore decomposed into connected components (CCs) being isolated subnetworks (Weitzel et al. 2007). Each CC can further be decomposed to strongly connected components (SCCs) which represent cyclic paths in the network.

### 13.4.2. The CumoVis Plug-In

In order to communicate the mathematical background of MFA with  $^{13}\text{C}$  carbon labeling experiments to practitioners from biology and engineering and to allow the visual exploration of ILNs, especially of the atom transition network, the plug-in *CumoVis* has been developed. *CumoVis* allows an interactive exploration of any kind of ILN in a 3D

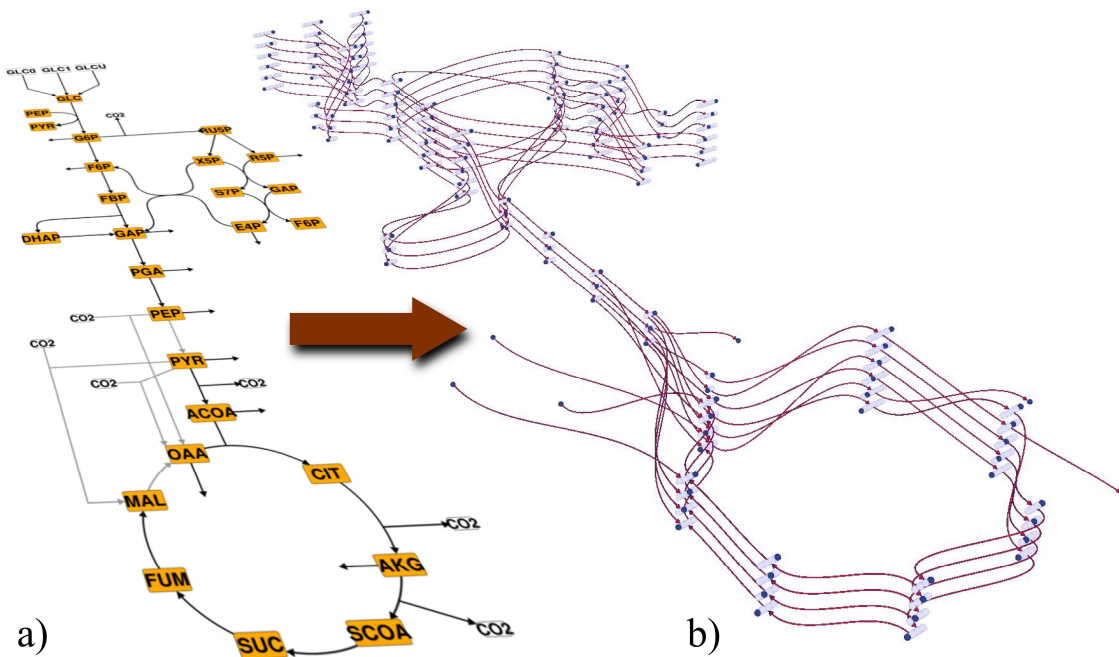


Figure 13.5.: Cumomer visualization (b) in relation to the metabolic network (a).

visualization. Thus, the tool helps to bridge the communication gap between modeler and experimenters. A comprehensive introduction into the scientific and mathematical background of the cumomer visualization as well as the implementation details of the *CumoVis* tool is given in (Droste et al. 2008b).

The cumomer visualization plug-in uses the planar metabolic network from *Omix*, stacks the cumomers on top of the corresponding metabolites and interconnects them with edges according to the atom transitions of the metabolic reactions. The edge shapes are automatically computed with focus on a minimized number of edge intersections.

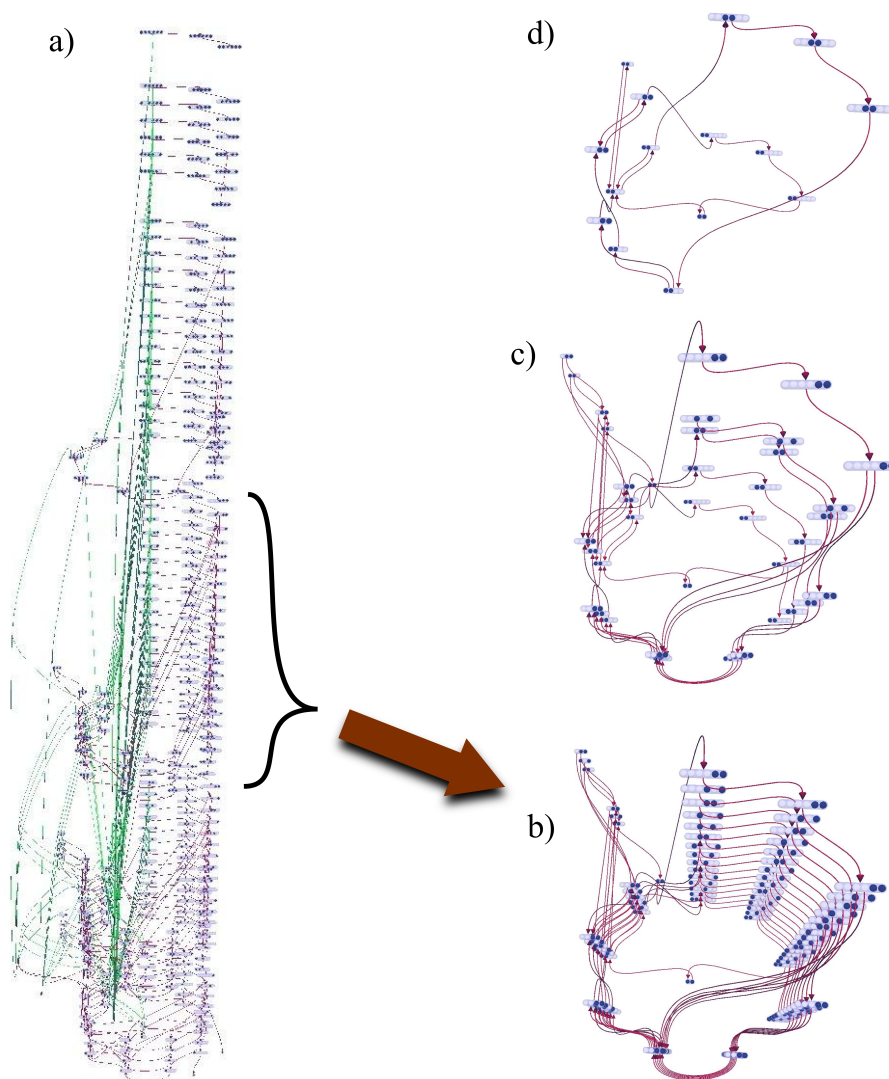


Figure 13.6.: Decomposition of the cumomer network: (a) complete network, (b) single level, (c) CC, (d) SCC.

## 13.4. 3D Visualization of Isotope Labeling Networks

Fig. 13.5 on page 119 shows the metabolic network diagram from *Omix* (a) and the 3D visualization of level 1 of the cumomer network (b). *CumoVis* uses a “pea pod”-like symbol to represent cumomers (cf. Section 13.3.2).

The 3D scene can be interactively explored by moving, zooming and rotating. The user can choose between viewing the entire cumomer network as shown in Fig. 13.6 a) and viewing the different levels of the network (see Fig. 13.6 b). Additionally, the user can hierarchically break up the visual representation of the network by choosing the CCs (Fig. 13.6 c) and SCCs (Fig. 13.6 d) of the single levels which are automatically computed by graph analysis algorithms. To understand the labeling dynamics in an experiment the search for cyclic paths is very important because, essentially, these cycles make the quantitative determination of isotope enrichment in a network a non-trivial task.

The fate of a labeled particle can be studied by path tracing in forward and backward direction. The user can choose a certain cumomer as start node for path tracing. Consequently, all successors or predecessors are shown up to a certain path length. In the same way, the user can choose a source and a target cumomer and the tool displays a path by which these two particles are connected. By choosing the same cumomer as source and target it is possible to display closed isotopomer fragment cycles. The respective reaction paths are graphically highlighted as depicted by Fig. 13.7.

*CumoVis* provides an overview of the atom transitions of a complete metabolic network. By this, errors in network models for the *<sup>13</sup>C FLUX 2* simulator can easily be detected. Beside that, the path tracing is a very valuable feature because it is one of the most time consuming operations that are often manually performed by experimenters. Since *CumoVis* visualizes the complete ILNs and allows to interactively break up the network the tool serves as an important educational tool in MFA that can be used to explain the rather complicated structure of ILNs to newcomers.

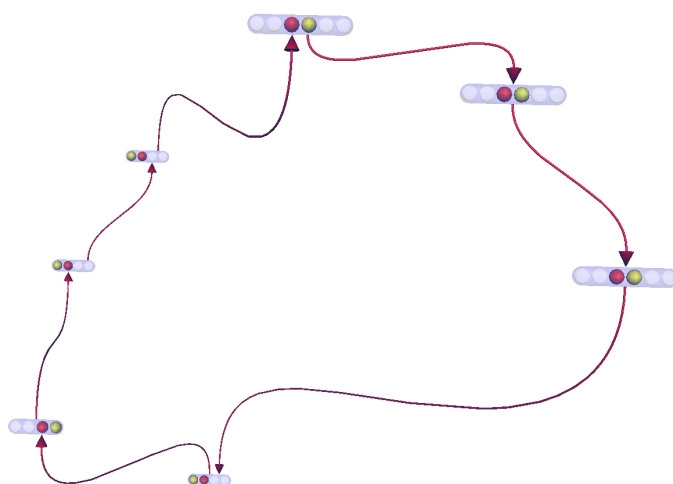


Figure 13.7.: Path tracing of a molecule fragment in the cumomer network.

## 13.5. 3D Visualization of Network Thermodynamics

### 13.5.1. Foundations

Like<sup>†</sup> any other process in nature biochemical reactions are governed by energetic principles. A reaction can only proceed in its nominal direction if the sum of the energetic reaction potentials of the educt metabolites is larger than the sum of these of the product metabolites. Here it should be noticed that thermodynamics does only make a statement on the feasibility of assumed reaction directions but not on the reactions' velocity. Even if a reaction has a large energy gradient there might be some regulatory effects on the actual flux rates.

The free Gibbs energy potential of a chemical substance with concentration (or, more precisely, activity)  $x$  is given by Eq. (13.1).

$$\Delta G = \Delta G^0 + RT \ln(x) \quad (13.1)$$

Here the term  $\Delta G^0$  is the free Gibbs energy under standard conditions.  $R$  is the Boltzmann constant and  $T$  is temperature (Beard et al. 2004, Kummel et al. 2006, Maskow and von Stockar 2005, Qian and Beard 2005). If a chemical reaction step has sources A, B, ... and targets U, V, ... then Eq. (13.2) must hold for the reaction to proceed in forward direction.

$$\Delta G_A + \Delta G_B + \dots > \Delta G_U + \Delta G_V + \dots \quad (13.2)$$

The set of thermodynamic constraints obtained in this way by collecting the respective inequalities for every reaction step imposes a multidimensional relation between possible substance concentrations, free Gibbs energies under standard conditions and flux directions. Because Eq. (13.2) holds both for the steady state and for dynamic transients, the display of thermodynamic potentials is an interesting complement to other simulation data like concentrations or fluxes.

### 13.5.2. The ThermoVis Plug-in

Using the third dimension is an appealing method to represent energy levels in a metabolic network. Here the analogy between reaction flow and hydrodynamic flow helps to understand the thermodynamic concept more intuitively. For this purpose the plug-in *ThermoVis* has been developed, a three dimensional thermodynamics visualization utility that visualizes the energies of metabolites and reactions on a two dimensional network using the third dimension. Such layered representations using the third dimension for information visualization have also been called “2½D” in the literature (Brandes et al. 2004).

Fig. 13.8 on the next page shows a three-dimensional representation of some parts of central metabolism with certain measured substance concentrations. Here, the Gibbs energy potentials of each metabolite is indicated by the height of the metabolite symbol.

---

<sup>†</sup>This Section has been published in (Droste et al. 2008a).



### 13.5. 3D Visualization of Network Thermodynamics

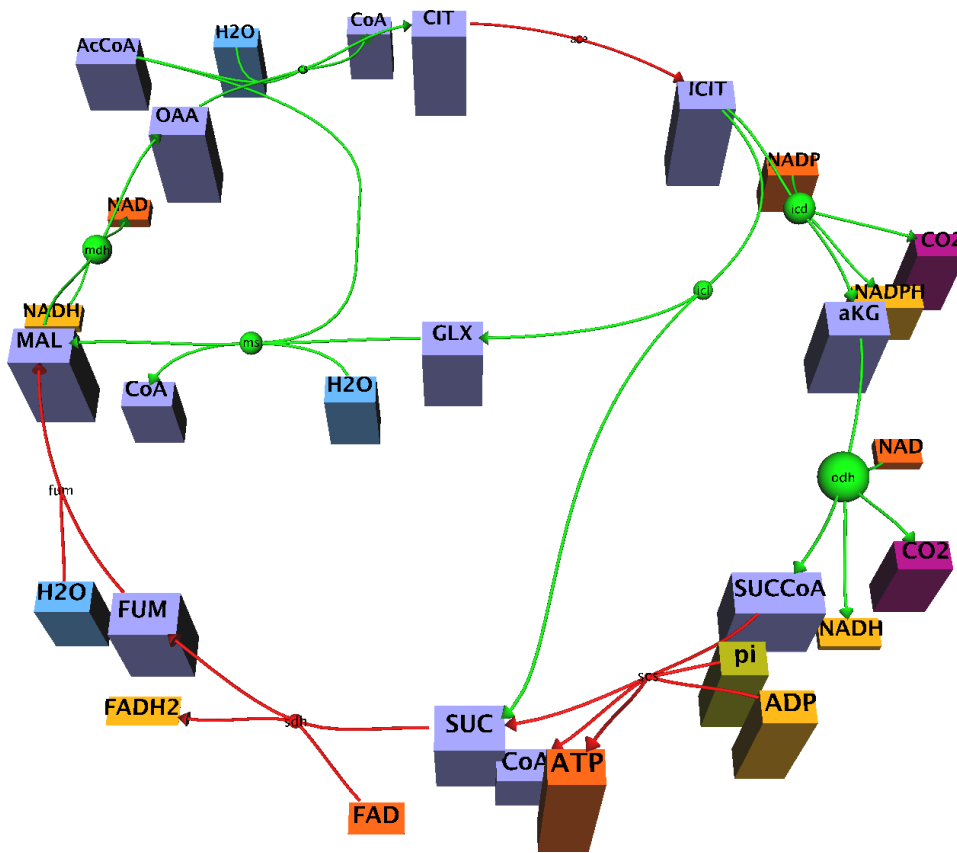


Figure 13.8.: Snapshot of a thermodynamic visualization. Here the citric acid cycle is shown.

In *ThermoVis*, reactions are shown as spheres. For fast and intuitive conceiving of the feasibility of reaction directions *ThermoVis* uses color. If an assumed reaction direction is feasible, the reaction symbol as well as the incoming and outgoing edges are colored green. The opposite case is indicated with red colorization. The tool supports interactive navigation in the network moving, rotating and zooming in the scene and also by focusing single reactions or whole pathways.

Many reacting compounds have almost the same Gibbs energy level and, of course, are not really distinguishable. Because reactions between such metabolites cannot be classified in their direction *ThermoVis* indicates size of the reaction's energy gradient by the diameter of the reaction symbol (see Fig. 13.8 and Fig 13.9 on the next page).

Fig. 13.9 a) shows a reaction that can be easily directed by the help of the reaction's energy level. As exemplified in Fig. 13.9 b) some reactions proceed only in the proposed direction by using energetic cofactors (like ATP and ADP or NADPH and NADP). These supporting metabolites balance the missing energy potential and realize the permutation from low-energetic metabolites to high-energetic ones.

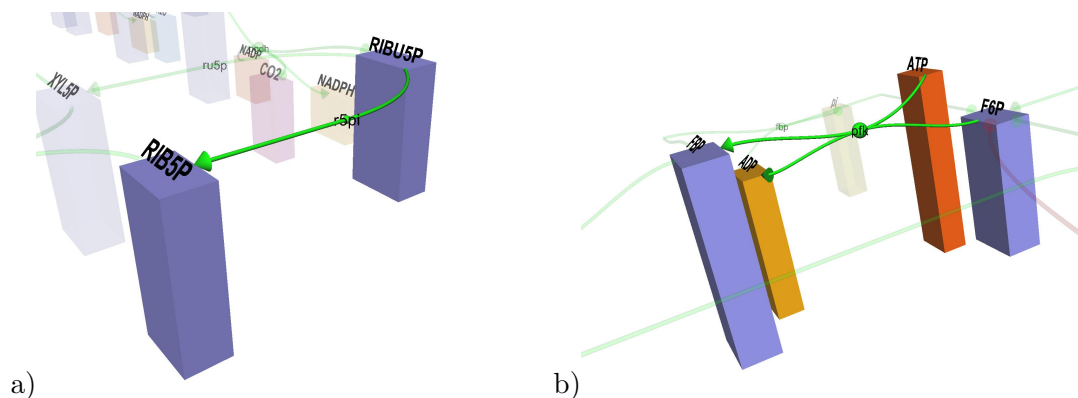


Figure 13.9.: *ThermoVis* snapshots of single reactions.

*ThermoVis* is a plug-in for *Omix*, i.e. it creates the three-dimensional network upon the two-dimensional drawing. The layout of nodes and edges as well as the visual appearance is completely taken from the network document. The biggest strength of the plug-in is: *ThermoVis* is highly customizable because the visualization is controlled by the OVL scripting engine. Hence, the user can equip the network documents with own OVL scripts that map thermodynamic data of any format on the height of the metabolites and radius of the reaction symbols, respectively.

## 13.6. Further Features

Further available plug-ins for *Omix* realize...

- ...network communication protocols. It is possible to load and store data on remote devices via SMB and SSH protocols independent from the executing operation system.
- ...data input from the popular spreadsheet programs *Excel* and *OpenOffice.org*. The plug-ins provide classes for OVL that allow users to load corresponding files directly within the visualization scripts.
- ...graphics paint engines. Two plug-ins have been available that allow to save images and animations as *Flash* file and to produce AVI and MPEG videos from visualizations of time-dependent data.

# Closing



# Chapter 14.

## Summary and Discussion

### 14.1. Aims

The aim of this dissertation was the development of a new approach for information visualization in metabolic networks. The realizing visualization tool must meet following requirements:

- The approach must combine drawing and modeling features with highly customizable abilities of data visualization.
- The resulting visualization tool must be oriented at the end users and their requirements.
- The drawing capabilities of the new tool must allow to create qualitative diagrams with a biologically motivated symbolism and customizable style.
- Techniques must be provided that accelerate the manual drawing process.
- It must be possible to view the diagrams in different levels of detail.
- The new approach must allow the visualization of data by annotation as well as by mapping on the visual properties of the network components.
- The tool must face the challenge to be adaptable to a rapidly changing application field.
- The visualization tool should be compatible with other modeling and simulation tools. Therefore, the software should be extensible by plug-ins.

The overall vision for this work was to create a tool that is the visual front end in a scientific work flow as depicted in Fig. 1.6 on page 16. Here, the tool serves as visual modeling software. The created model is exported to a simulator. Simulations are performed that can be started and controlled in the visualization tool. Subsequently, the simulation results as well as experimental data are visualized in the network diagram. The new visualization tool has to be designed such that the simulation tools and data formats in the scientific work flow are arbitrarily exchangeable.

### 14.2. Results

#### 14.2.1. Novel Solutions

The present work successively introduced the software tool *Omix*, an experimental platform for a large number of novel methods for the customizable visualization in the context of metabolic networks. The feature sheets (pages 41, 42, 103 and 104) list 67 particular features of *Omix* that distinguish the visualization tool from other related approaches. 36 of these features are completely novel developments that have not existed until now. Amongst them are important features like:

- The automatic duplication of multiply connected metabolites (cf. Chapter 3.2.1)
- The manner metabolic pathways are visually represented and the ability to use these graphical elements for data visualization purpose (cf. Chapters 3.2.2 and 10.5)
- The layout pattern, a skeleton that helps to arrange pathways in the diagram (Chapter 5).
- Motif stamps accelerate the manual drawing process by wrapping multiply occurring drawing steps (Chapter 6).
- Making visualization of data programmable in the novel *Omix Visualization Language* (OVL) (Chapter 9.1).
- A complete OVL development environment including a debugger for OVL code (Chapter 9.2).
- Providing extensive visualization features like annotating the network components, mapping data on the visual appearance of graphical elements and animating time-dependent data (Chapter 10).
- The *Visualization on Demand* mode for the interactive exploration of hierarchically organized data in networks (Chapter 9.3.1).
- A graphical modeling and visualization framework for *13C FLUX 2* (Chapter 13.3).
- 3D visualization of atom transition and isotope labeling networks (Chapter 13.4.2).

The broad usage of the software was proven in several application examples with respect to the drawing features (cf. Chapter 3.7), the visualization capabilities (see Chapter 10) and the possibilities in extending the tool with new features (cf. Chapter 13).

By providing extensive flexibility, *Omix* is very well suited as pre- and post-processing framework for experimentation and simulation in a multi-disciplinary research field. Biologists, for instance, can use *Omix* for creating diagrams, for presenting results and for interpreting data from the visualizations. Engineers can use *Omix* for network modeling and visual data analysis. Mathematicians contribute their expertise in model validation

and data pre- and post-processing and, finally, computer scientists can realize comprehensive visualization methods in OVL, have expertise to develop plug-ins and can embed the software in scientific work flows.

### 14.2.2. Extraordinary Achievement

An extraordinary achievement of this doctoral work is the creation of a visualization tool that is usable, stable and robust combined with a large amount of novel approaches developed in a one-person project. Killcoyne and Boyle (2009) stated that many tools in the life sciences were developed only for publication purpose. “The expectation that software should be as novel as a scientific discovery has meant that even software that a developer would consider a failure (due to lack of use) can be considered a success by scientists” (ibid.). This attitude must be seen critically because a novel idea about visualization published in a journal paper is not necessarily a significant scientific contribution without being realized in any kind of employable tool.

The maturity state of the here presented visualization tool *Omix* is far beyond that of a prototype. The software project consists of ca. 300,000 lines source code originally developed for *Omix*. The stability and usability of *Omix* has been proven, for it is employed in multiple scientific groups since 2009. *Omix* has become an established standard

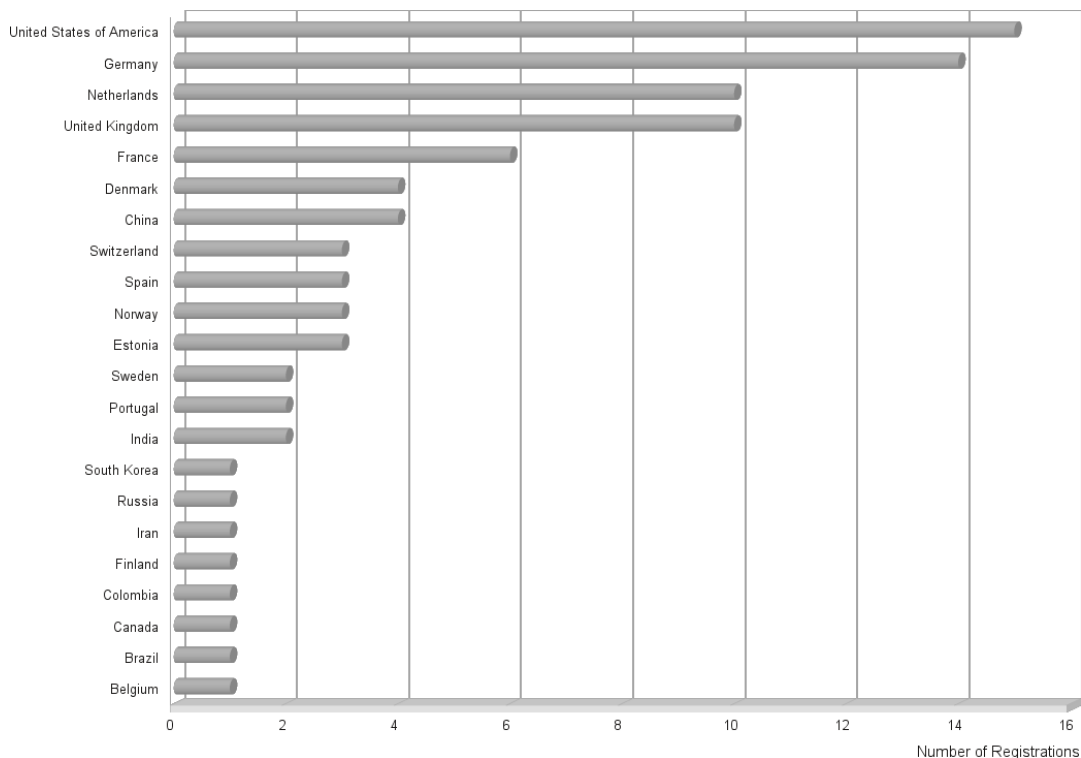


Figure 14.1.: Number of *Omix* users per national origin.

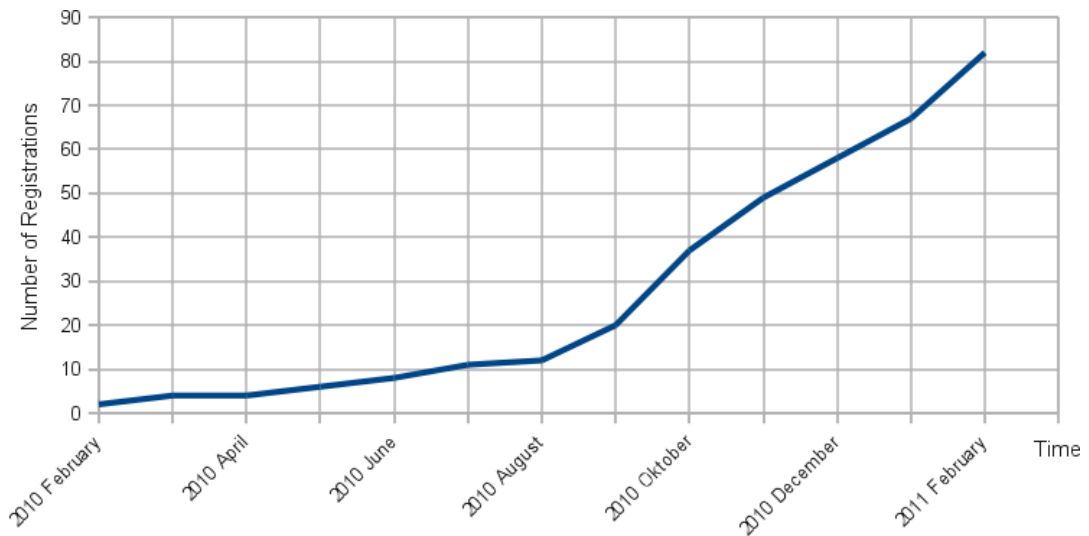


Figure 14.2.: Number of registrations over time.

for network drawing and visualization in the Institute of Bio- and Geosciences, IBG-1 of the research center Jülich. The novel concepts realized in *Omix* have been published in nine journal papers and conference contributions.

As has been said before, the program is available for non-commercial, academic use. Since February 2010, *Omix* can be downloaded after registration on the website [www.13cflux.net/omix](http://www.13cflux.net/omix). In March 2011, 89 persons from 67 different institutes in 22 countries worldwide have registered at the download server as shown in Fig. 14.1. The number of registrations continually increases (cf. Fig. 14.2). All these new customers can hardly be directly associated with, for instance, international conference talks about the software. This indicates a broad popularity of the tool.

### 14.2.3. Realized Vision

In *Omix* the overall vision is realized as shown in Fig. 14.3 on the next page. The software allows to create network diagrams. Plug-ins can be activated on the document that equip the network with certain parameters necessary in computer simulation. Likewise, a plug-in can realize the user interaction for editing these parameters as has been demonstrated in Chapters 13.1.1 and 13.3.2. Available plug-ins allow kinetic modeling in combination with *Modelica* and the modeling of carbon labeling experiments for *13C FLUX 2*. After creating the diagram and specifying the model parameters the network can be exported to the simulator specific data format. Here, plug-ins are available for the export to several simulator specific formats like FML used by *13C FLUX 2* (Dalman et al. 2010a) or SBML (Hucka et al. 2003).

The plug-in interface allows to equip the application with interactive components and to develop comprehensive dialogs. By this, a plug-in can realize the control of simulator



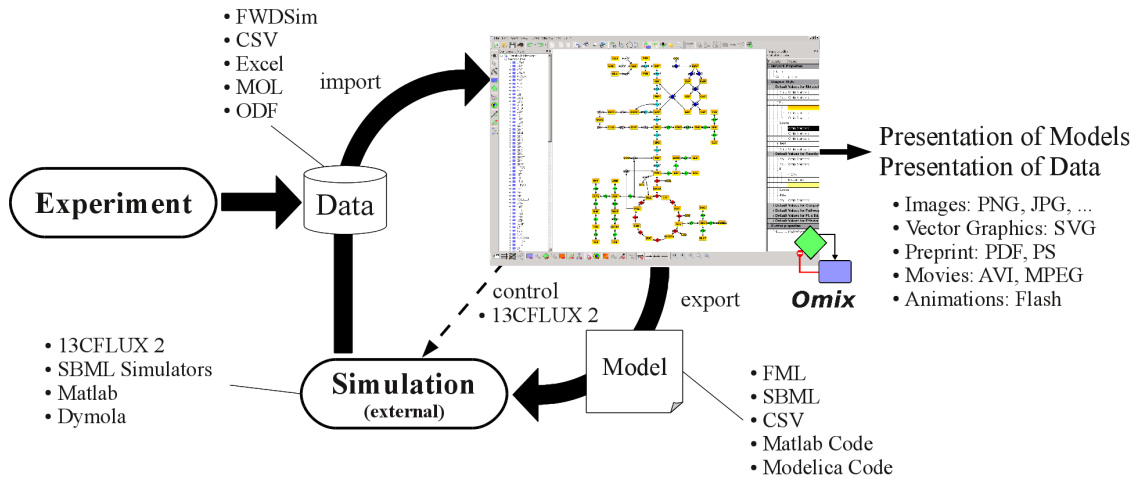


Figure 14.3.: *Omix*, a visual front end in a scientific work flow consisting of network modeling, simulations and experiments, data visualization and presenting of results.

programs. Many simulators are command line applications. By providing a window-based dialog system for handling the simulations the user friendliness of those simulators is enormously enhanced. This is demonstrated by the launcher plug-in for *13C FLUX 2* introduced in Chapter 13.3.3.

After simulation the results can be visualized in the network diagrams by creating a specific visualization solution adapted to the given situation. By default *Omix* can handle the CSV file format but the software can be extended by further file filters for the import of data from experiment and simulation. The FWDSim data format of *13C FLUX 2* (Dalman et al. 2010a) is fully supported by an available plug-in (cf. Chapter 13.3.4). The visualization can be used for presentation of results, scientific discussions and publication. The software provides a set of export filters for generating images. Even this set can be extended by plug-ins as demonstrated in Chapter 13.6.

The full cycle of modeling, simulation and visualization depicted in Fig. 14.3 has been realized in the context of this work for the *13C FLUX 2* simulation framework. It has become clear, that *Omix* can be employed as visual front end for many other simulators dealing with metabolic networks only by implementing the necessary interfaces between the simulators and *Omix* in plug-ins.

### 14.3. Future Perspectives

Although the here presented visualization approach brings a large number of new features that meet the urgent requirements of the research context, many wishes remain unfulfilled. This section provides ideas for possible future projects in combination with the here presented customizable visualization approach.

#### 14.3.1. Other Network Layers

As introduced in Chapter 1.8 the current approach focuses on metabolic networks. A possible future work can concentrate on the extension of the approach by the ability to visualize genetic networks and signal transduction networks. Here, essential work has been done in other approaches dealing with the layout of large scale networks (Batagelj and Mrvar 2002, Shannon et al. 2003, Freeman et al. 2007). An interesting issue would be to combine these approaches with the script-based data visualization presented in this work.

#### 14.3.2. Network Layout

Another interesting task is the combination of the layout pattern approach (cf. Chapter 5) with automatic layout algorithms. The layout pattern is well-suited to arrange familiar pathways from a given layout-less network. However, in case of large networks this takes only a rather short phase of a very time-consuming drawing process especially when most network parts are unknown. Here, the manual arrangement of network components could be prepared by applying for instance force-directed placement whereas all nodes and edges on the layout pattern are fixed. It would be interesting to evaluate whether the combination of semi- and full-automatic layout improves the time factors while preserving the traditional layout conventions in a network drawing process.

#### 14.3.3. Metabolic Flux Analysis

##### Atom Transitions

In the modeling process of carbon labeling experiments the atom transitions of the individual reactions are edited with an abstract representation of the involved metabolites. However, the representation of the atoms of a molecule by a set of colored circles is not intuitive (cf. Chapter 13.3.2). It would be appealing if the single metabolites are represented by the actual chemical structure of the compound. This would enormously improve the visual modeling process because structural errors in the transition models can immediately be identified.

The crucial gap is the lack of standardized atom numbering. The FML format of the *13C FLUX 2* simulator only represents atom transitions in abstract manner. The actual position of the atoms in the molecules is not specified in FML. The Mol plug-in introduced in Chapter 13.2 computes the visual image of chemical structures from the Mol data

format (Dalby et al. 1992). Unfortunately, this format does not include a standardized sequence of atoms. Two Mol files can represent the same molecule with different atom order. Here, a promising solution may be given in the *InChI code*, a textual, unified representation of chemical molecules (Stein et al. 2003). The InChI code realizes a unique atom numbering, however, atom coordinates are not represented. Therefore, converters between InChI and Mol exist.

An important future task is the utilization of the InChI code for the visualization of chemical structures in combination with the modeling of atom transitions. A related task is the extension of the *CumoVis* plug-in by the ability to visualize the spacial structures of the single molecules.

### Measurement Data

Another required improvement of the *13C FLUX 2* Modeling Plug-in is the ability to load measurement data into the model. Currently, this can only be done by manually copying contents into the FML files in a usual text editor. However, this procedure is very error-prone. An interesting solution would be to realize inter-process communication between the data post-processing software tools and the *Omix* plug-in for a file-system-independent exchange of measurement data. Here, another crucial gap is the lack of unified identifiers of reactions and metabolites. Sooner or later this can only be solved by embedding the modeling processes in an integrative scientific work flow system that utilizes various databases for the unification of identifiers.

### 14.3.4. 3D Visualization

As demonstrated in Chapter 13.5 the third dimension is suitable for the visualization of information on a planar network diagram. An interesting task would be to find other application fields for this approach, for instance, the visualization of network sensitivities.

In the context of 3D visualization, the representation on plain screens hampers the ability to grasp the spacial dimension of the visualization. Interesting advances have been published about biochemical network visualization in combination with virtual reality (Dickerson et al. 2003, Yang et al. 2005; 2006). It should be discussed to equip meeting rooms of scientific groups with stereoscopy technologies as it nowadays becomes state-of-the-art in cinemas.



# Appendix



# Appendix A.

## Drawing Tool Survey

In this appendix the user guidance and drawing capabilities of selected software tools for drawing biochemical networks are compared with common graphics and office tools.

### A.1. Office Software

*Microsoft Office* by *Microsoft Corporation* is a proprietary, widespread collection of various office tools including a text processing application and a slide presentation program, amongst others. A drawing engine is embedded in these tools as shown in Fig. A.1). The user can draw items on the page or slide from a set of standard shapes like rectangle, ellipse, triangle, pentagon etc. The shapes can carry additional text that is positioned in the center of the item by default. Furthermore, text can be inserted as stand-alone component of the diagram. Beside these components, edges can be inserted. Here, the drawing engine provides three possible edge shapes: line, piecewise linear curve with orthogonal segments and S-bend. Edges can be connected to the graphical items at a number of docking positions. By this, flux diagrams can be created. In addition to the standard sets for graphical items and edges, the user can draw arbitrary item shapes or edge curves. Here, the shapes can be edited as sequence of cubic Bézier curves (Bézier spline; refer Salomon 2006).

The single utilities for drawing components on the drawing area are available on a toolbar (cf. Fig. A.1). Every drawing step consist of activating the corresponding feature on the toolbar and dragging over the drawing area in order to define the shape or bounds of the new component. The single drawing utilities are only active until the corresponding component is inserted in the diagram. The activity of the drawing utility is indicated by the mouse cursor.

Graphical items and edges can be changed in their appearance in many ways. After selecting a component a further toolbar appears giving access to the item's visual properties. The user can change the fill, line thickness, line shape, arrows etc. Furthermore, the items can be equipped with a drop shadow or three-dimensional perspective. Drawings in a text document or a slide show can be exported from *Microsoft Office* to established image formats like PNG, TIFF and JPG. An open source software collection with nearly the same drawing features and a comparable user guidance is *OpenOffice.org*.

## Appendix A. Drawing Tool Survey

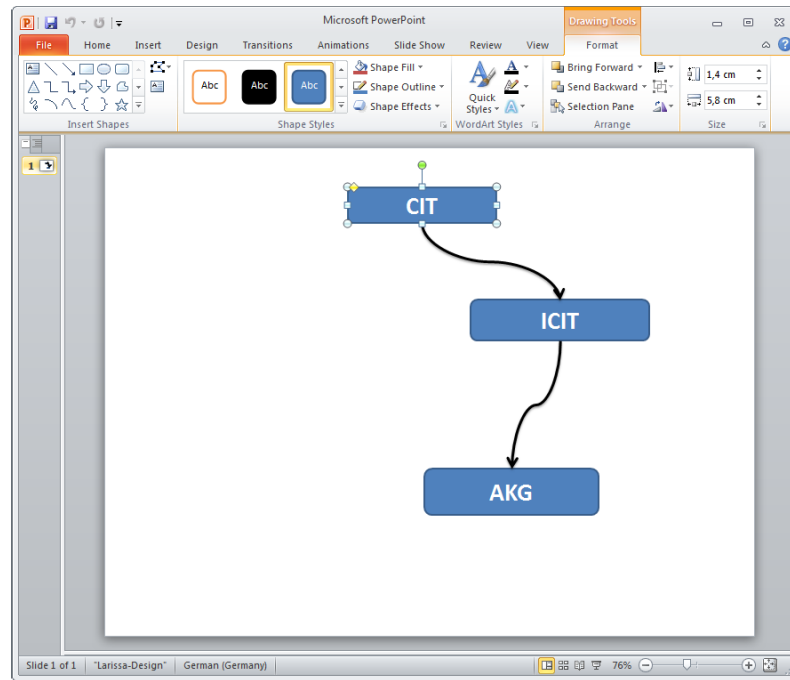


Figure A.1.: Screen shot of the office software *Microsoft PowerPoint*.

### A.2. Graphics Software

A popular graphics software is *Corel Draw*, a proprietary drawing tool by *Corel Corporation* (see Fig. A.2). The software is vector-oriented and aims at providing high quality graphics. *Corel Draw* allows to edit multiple graphics simultaneously. The single document is page-based. Multiple pages can be created each carrying an independent drawing. The drawing area always shows one page. Axis rulers, a background grid and the option to insert guides help to locate items on the drawing area.

In *Corel Draw*, arbitrary graphical components can be drawn. The software provides several standard shapes like rectangles, ellipses and polygons available as utility button on a toolbar. Moreover, the user can create arbitrary figures with a freehand bend utility. Curves are represented as Bézier splines that can be handled very intuitively. Curve editing is done with an extra utility. The utility allows to split and join curves, define the curve smoothness and even change the edge direction. Furthermore, different operations can be performed on multiple polygons like combination, union and intersection. The software also allows the integration of text fields in diagrams.

In *Corel Draw*, the activity of a drawing utility is always indicated by the appearance of the mouse cursor. Here, the cursor shows individual symbols for the different active features. A drawing utility is always enabled until the user selects another tool. The single features of the drawing tool are available as individual buttons on the left, vertical toolbar shown in Fig. A.2.



### A.3. Drawing Software for Biochemical Networks

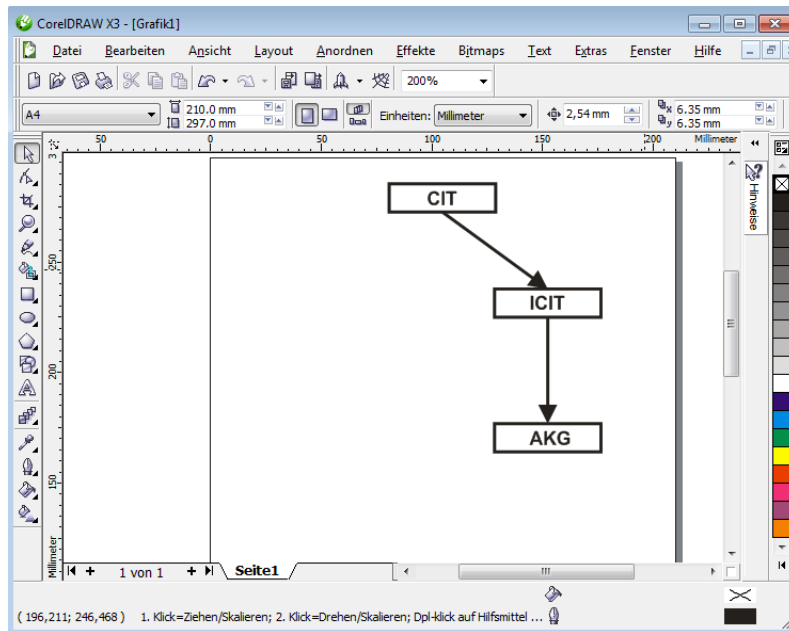


Figure A.2.: Screen shot of the graphics design software *CorelDraw*.

The visual appearance of the graphical components can be adapted in many ways. Here, *Corel Draw* has some outstanding features that allow filling shapes with color gradients, fill patterns and generated textures that can be freely parametrized. When a graphical component is selected in the drawing area, a further toolbar appears giving access to the particular properties of the component like position, width and height. Furthermore, the style preferences of selected components can be simply changed by choosing corresponding utilities on the toolbar, for instance, the fill color utility used to change a rectangle's color.

Analog to office software, flux diagrams can be created in *Corel Draw* by connecting graphical items with edges and arrows. *Corel Draw* allows the export of drawings into a large set of image and vector file formats. Other widespread vector graphics programs are *Adobe Illustrator*, *Microsoft Visio* and *Inkscape* which provide similar functionalities and user guidance.

### A.3. Drawing Software for Biochemical Networks

In the following, a review is given about three selected software tools that are developed for the application field of biochemical network. All tools deal with modeling and/or visualization of biochemical network as pre- and post-processing of experiment and simulation.

### A.3.1. CytoScape

*CytoScape* is an open source visualization software for biochemical network originally developed at the Institute of Systems Biology in Seattle, USA and published by Shannon et al. (2003). Fig. A.3 shows a screen shot of the main window. The software allows to map data to the visual properties of network nodes and edges in the network diagram. By this, the software realizes the visualization approach advocated in this thesis. Hence, it is worth taking a look on the network drawing capabilities of the application as a comparable related work.

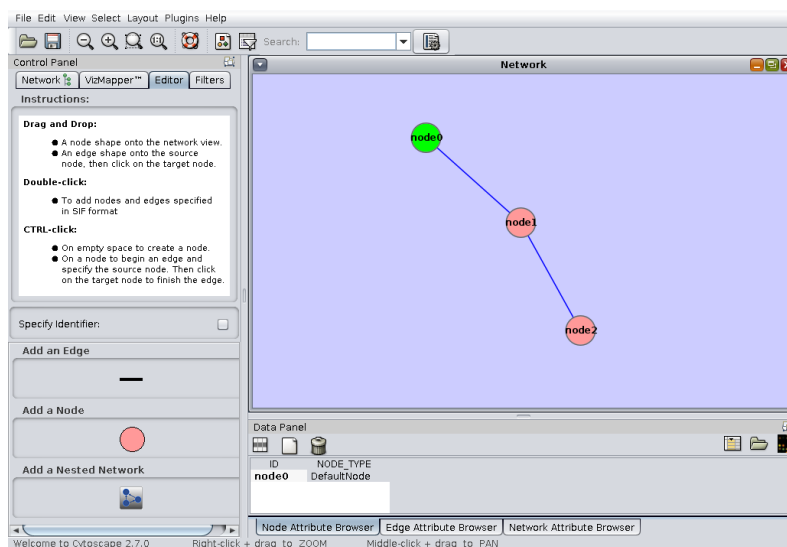


Figure A.3.: Screen shot of the visualization tool *CytoScape*.

In general, *CytoScape* can be used to draw metabolic network diagrams. However, the user guidance of *CytoScape* is very different from the above discussed office and graphics programs and, in most cases, takes a lot of getting used to:

1. The software is not document-based but session-based. A session can contain multiple networks, each drawn and displayed in an own subwindow. The entire session is stored in a file. Only one file can be opened simultaneously.
2. The network editing utilities are not directly present on the main window. They do neither appear on the toolbar nor in the menubar. On the left side the main window contains a sidebar called “Control Panel”. This window component contains a tabs representing further subcomponents. Here, the third tab called “Editor” reveals the utilities for inserting edges and nodes as shown in Fig. A.3.
3. The edge and node adding feature is not available as tool button. A node and edge symbol is displayed on a label widget. By dragging the node symbol and dropping

### A.3. Drawing Software for Biochemical Networks

---

it above the drawing area, a new node is inserted. New nodes are initially labeled with a consecutively numbered default name.

An edge is inserted between two nodes by dragging the edge symbol from the control panel dropping it above the start node and, subsequently, clicking on an end node. By this, it is not yet defined, what kind of biological entities the nodes represent and what kind of relation the edge displays.

4. The type of a node and likewise of an edge is property of the network item that must be specified in the “Data Panel” which is a sidebar at the bottom of the main window. After selecting components in the diagram area their properties appear as rows in a table. Initially, only an identifier attribute of the nodes and edges is displayed as column. The next step is to activate further displayed attributes from a menu. The attributes “NODE\_TYPE” or “interaction” can be edited to define the type of the selected node and edge, respectively. As a further limitation, the attribute editor does not offer a set of valid options. Without expert knowledge it is simply impossible to make the correct input.
5. Changing a node’s or edge’s visual appearance is done globally in the style sheet of the diagram. The network style includes global appearance properties for all nodes and for all edges. Thus, in case of a metabolic network where two node symbols are required, i.e. for metabolites and reactions, all reactions in the diagram have to be changed individually in order to have a different appearance. Access to the individual visual properties of a node and edge, respectively, is given in the context menu. Here, the user can override the global settings for the respective network component.
6. The drawing area does not offer scrollbars. Moving the drawing area in the bounds of the displaying window component is done in a “bird’s-eye-view”. being a small overview image of the diagram displayed on the left side of the main window. The bird’s-eye-view shows the entire diagram and a rectangle representing the the currently displayed subsection. By moving the rectangle in the overview diagram, the drawing area is scrolled.

This completely exceptional user guidance makes the access to the software very hard for someone who is accustomed to office or graphics software. Actually, *CytoScape* cannot be recommended for drawing metabolic networks from scratch. The user guidance let the drawing process be even more time consuming than in a common drawing software by guess. This is not an unexpected result because the software’s main purpose is the visualization of molecular interaction networks with gene expression profiles (Pavlopoulos et al. 2008). In this application field, networks are not drawn manually but origin from databases or are computed from experimental interactome data.

### A.3.2. Vanted

*Vanted* is a visualization software developed at the Leibniz Institute of Plant Genetics and Crop Plant Research (IPK), Gatersleben, Germany and originally published by Junker et al. (2006). Gehlenborg et al. (2010) list *Vanted* as one of a small set of tools that are recommended for building metabolic network diagrams from scratch or editing existing ones. Fig. A.4 shows a screen shot of the main window.

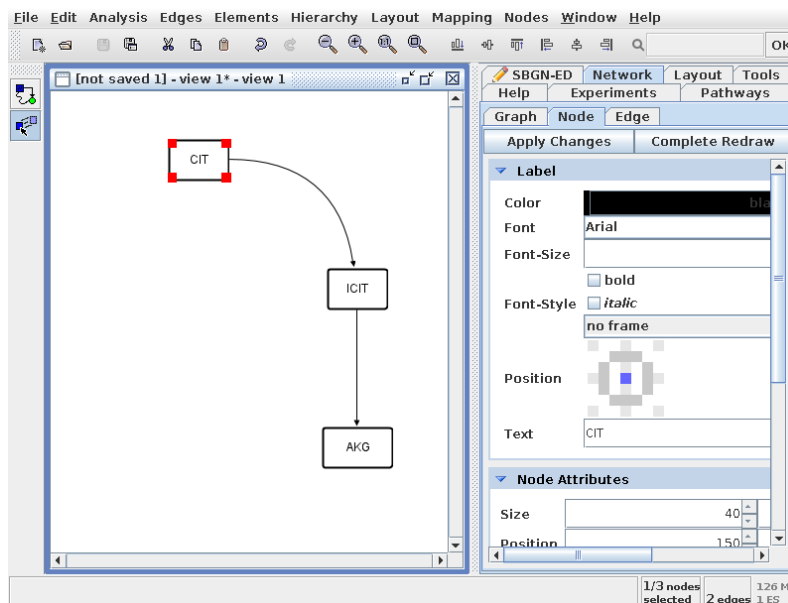


Figure A.4.: Screen shot of the visualization tool *Vanted*.

*Vanted* is a document based program. Multiple network files can be edited simultaneously. One file can only contain one network. For drawing a network the tool provides only two utilities. The first utility allows to insert nodes as well as edges and the second one is used to select items in the diagram, to move and to reshape them. Both utilities are available as tool buttons on the left, vertical toolbar in the main window. When the utility for creating nodes is active, the mouse cursor shows a hand shape indicating that components can be inserted. By clicking on an empty area in the diagram, nodes are inserted. They appear as initially unlabeled and uncolored rounded squares. By clicking on a node, edge drawing is started. By subsequently clicking on the drawing area control points are inserted that define the shape of the Bézier curve. After clicking on a second node the edge is finally inserted.

Thereafter, the moving/selecting utility allows to edit the single components individually and define their visual appearance. Double-click on a node or edge opens a property dialog window. Here, a label can be entered. Furthermore, the visual properties of a selected node and edge can be edited in the right side window. Here, a few visual properties like font, text color, fill color, frame color, line thickness, arrow head and shape can be

### A.3. Drawing Software for Biochemical Networks

---

changed. For edge editing, *Vanted* implements Bézier curve handling usability. Against the common standard, edges are not realized as series of cubic Bézier curves but as single Bézier curve of arbitrary degree. Every time the curve is dragged in a certain direction, a further control point is inserted. By this, the degree of the Bézier curve continually increases. The only way to diminish the number of control points is to remove all of them by converting the curve into a line. Another confusing behavior of the software in combination with edge editing is that the single control points disappear while they are moved.

By default, nodes and edges in *Vanted* do not represent a biological entity. They are just network components. There is no way to assign a biological semantics to the graphical items. Thus, it is not possible to export the diagram inherent network topology to a simulator. Consequently, *Vanted* documents do not contain models, they are just diagrams for visualization purpose. The lack of semantics of the diagram components in *Vanted* is especially problematic in metabolic networks. As introduced in Section 1.5, metabolic networks can be represented either by hypergraphs or by bipartite graphs (cf. Fig. 1.3). Unfortunately, *Vanted* does neither support hyperedges nor two node types by default. There is an add-on available integrating the semantics of the Systems Biology Graphical Notation (SBGN) (Le Novere et al. 2009) into *Vanted* (Czauderna et al. 2010).

#### A.3.3. CellDesigner

*CellDesigner* is an editor for genetic and metabolic networks. The tool serves as modeling tool for various simulators by supporting the Systems Biology Markup Language (SBML) (Hucka et al. 2003) in combination with the *Systems Biology Workbench* (Sauro et al. 2003). The software first published by Funahashi et al. (2003) has been developed in multiple Japanese institutions and is freely available. Like *Vanted*, Gehlenborg et al. (2010) recommend *CellDesigner* to be used for pathway drawing purpose. Metabolic networks in *CellDesigner* are drawn as SBGN process diagram.

*CellDesigner* is a document-based editor. Multiple documents can be edited simultaneously. The main window of the software is shown in Fig. A.5. After creating a new document, the user is asked to define the maximal bounds of the diagram. The dimensions of the new diagram must be well-approximated because they can never be changed anymore. Thereafter, network components can be added to the drawing area. For this, the software provides over 30 different node and edge symbols which requires background knowledge about their semantic meanings. The utilities for adding nodes and edges are available on several toolbars in the main window. Nodes are added to the drawing area by choosing one of the node adding utilities and simply clicking on an intended position in the diagram. The mouse cursor does not indicate the activity of an adding utility. Before a component is inserted, the user must enter the dedicated name of the new biological entity. After adding a component, the editor returns to the selection mode.

Nodes are equipped with docking positions for edges. For adding edges to the drawing area, a docking point of a source node and of a target node has to be selected. As a

## Appendix A. Drawing Tool Survey

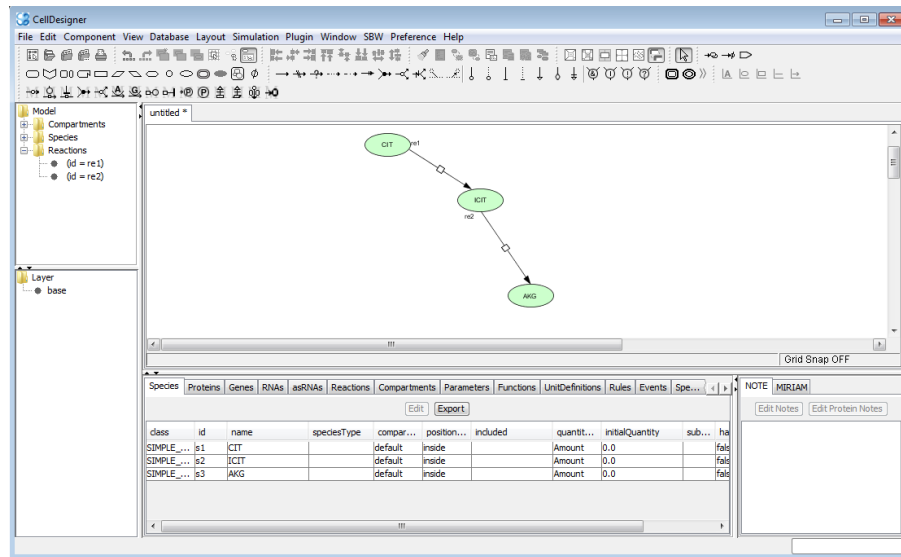


Figure A.5.: Screen shot of the network drawing software *CellDesigner*.

particular feature, *CellDesigner* allows to add further source and target nodes to edges and, by this, realizes a proper hypergraph representation of the network.

In the selection mode, nodes and edges can be selected, moved and changed. Changing the visual appearance of nodes is restricted to the node bounds, fill color, stroke thickness and font size. The node shape cannot be changed unless the node type is changed. Edges can only be adapted in their color and line thickness. Concerning edge shapes the user only has the choice between a straight line and a polyline with optionally orthogonal segments. Basically, *CellDesigner* allows the global definition of node and edge appearances in the application's preferences.

## Appendix B.

# Evaluation of the Semi-Automatic Layout Approach

### B.1. Case Study Design

To investigate the effectiveness of the semi-automatic layout techniques developed in Part II a case study has been performed consisting of two phases. In the first phase, a layout-less network had to be drawn with *Omix* either with or without the assistance of the semi-automatic techniques. In a second phase, the resulting diagrams were rated by a jury. The case study does not represent an exhaustive evaluation but it demonstrated the use of the layout pattern and motif stamps for network drawing.

### B.2. Elaboration

Sixteen researchers with bio(techno)logy background from five scientific groups participated in phase one. One half of the participants was encouraged to use the semi-automatic layout techniques (group 1) whereas the other half was banned from using them (group 2). The network to be drawn was medium-sized (44 metabolites and reactions composed in five metabolic pathways, cf. Fig. B.1). The network topology did not contain duplicated metabolites. A sample solution was not made available. In order to introduce the drawing capabilities of the tool and using the layout pattern as well as the motif stamps we provided video lessons. After a short training phase the network was to be drawn and the time effort for the drawing exercise was measured. It was up to the participants to chose a personal trade-off between drawing fast and drawing a “nice” diagram. After finishing, group 1 was asked to assess the ease-of-use of both semi-automatic layout techniques and their benefit for solving the exercise.

In the second phase, the drawing results were examined by ten experienced life scientists. In addition to the manual drawings, a set of diagrams has been created by applying automatic graph layout algorithms on the given network. Every jury member received a random collection of automatically, full-manually and semi-automatically drawn diagrams. Each drawing was rated with respect to its recognition value and aesthetics.

## Appendix B. Evaluation of the Semi-Automatic Layout Approach

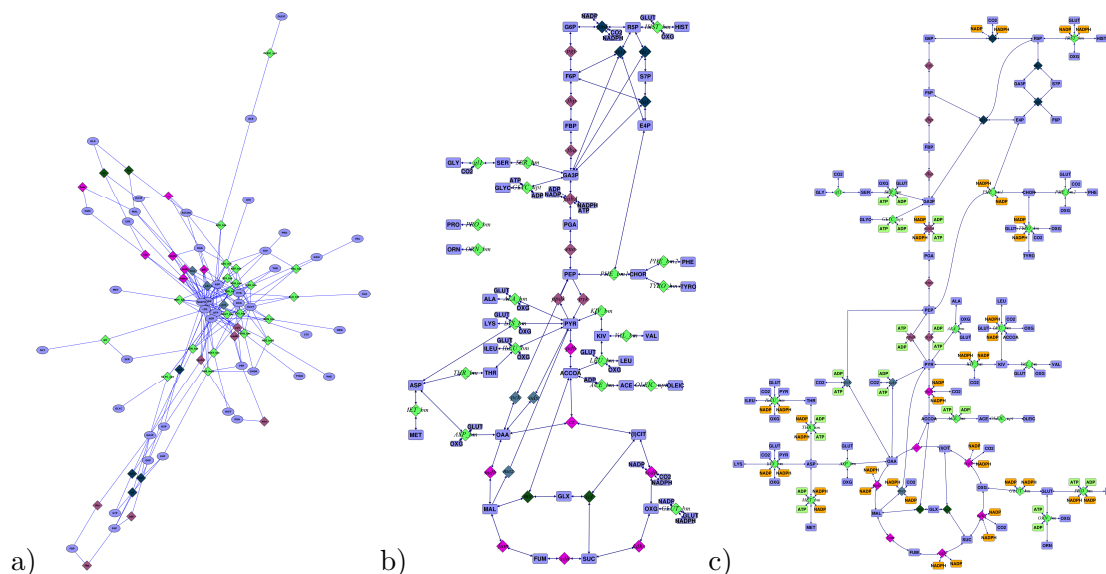


Figure B.1.: Selected results of the network drawing case study: (a) automatically (spring embedded layout), (b) manually and (c) semi-automatically drawn network.

### B.3. Results

The results of the surveys are shown in Tables B.1 and B.2. Fig. B.1 shows one example diagram from the automatically generated (Fig. B.1 a), the manually drawn (Fig. B.1 b) and the software-assisted drawn networks (Fig. B.1 c) in each case. In average, the semi-automatic layout methods accelerated the drawing process for about half an hour that roughly is 20% of the average duration (see Table B.1). The expert group valued the recognition value as well as the aesthetics of the automatic computed diagrams very low (see Table B.2). The recognition value of both, full and partly manually drawn networks were comparably rated in the upper third. That is not surprising because the central pathways of all diagrams were arranged according traditional conventions (cf. Fig. 4.1 on page 46). The aesthetics of the semi-automatic drawings is slightly higher valued than the manual drawings. Here, it was noticeable that the nodes and edges of the TCA pathway in nearly all semi-automatic drawings were shaped to a perfect circle in contrast to the manual arrangements (cf. Fig. B.1 b and c). This noticeably improves the aesthetic impression as well as the recognition value of a diagram.

The benefit of the layout pattern for solving the exercise was assessed 6.9 and of the motif stamps 6.4 in average (1=not useful, 10=very useful) (cf. Table B.2). The ease-of-use of the layout pattern was valued 6.1 and of the motif stamps 7.0 in average.



## B.4. Conclusion

---

	Time Effort	Recognition Value	Aesthetics
Automatic Drawing	< 1 min	1.6	1.7
Manual Drawing	2:07 hours	7.6	5.8
Semi-Automatic Drawing	1:40 hours	7.7	6.7

Table B.1.: Time effort and ratings of the diagrams drawn with different approaches. The ratings were given from a scale between 1 and 10 (1=very low, 10=very high).

	Benefit	Ease of Use
Layout Pattern	6.1	6.9
Motif Stamps	7.0	6.4

Table B.2.: Benefit of the two semi-automatic layout methods (1=not useful, 10=very useful) and ease-of-use (1=very difficult, 10=very easy).

## B.4. Conclusion

The case study demonstrates, that the semi-automatic layout techniques reduce the time effort to manually layout a diagram according to established layout standards while they simultaneously improve personal aesthetics. From further, informal discussions with the participants it was found out that the layout techniques were considered valuable for the arrangement of familiar pathways but that the most time-consuming issue was to understand unfamiliar parts of the network before being able to arrange them. In a scenario, where the drawing process intimately lies in the research objective of the user, the layout techniques may be more beneficial. This scenario, however, was not intended by this case study and remains as an interesting future evaluation task.



# Original Publications

## Documentations

- Droste, P. 2008a. *Omix – User Manual*. Institute of Bio- and Geosciences, IBG-1 Biotechnology, Forschungszentrum Jülich, 52425 Jülich, Germany.
- Droste, P. 2008b. *Omix Visualization Language - Technical Manual*. Institute of Bio- and Geosciences, IBG-1 Biotechnology, Forschungszentrum Jülich, 52425 Jülich, Germany.
- Droste, P. 2010. Omix Library Reference Documentation. [www.13cflux.net/omix/doc](http://www.13cflux.net/omix/doc). Visited March 2011.
- Droste, P. 2011. *Omix Plug-in Development Manual*. Institute of Bio- and Geosciences, IBG-1 Biotechnology, Forschungszentrum Jülich, 52425 Jülich, Germany.

## Related Publications

- Dalman, T., Droste, P., Weitzel, M., Wiechert, W., and Nöh, K. 2010. Workflows for Metabolic Flux Analysis: Data Integration and Human Interaction. In T. Margaria, and B. Steffen (Eds.) *Leveraging Applications of Formal Methods, Verification, and Validation*, vol. 6415 of *Lecture Notes in Computer Science*, (pp. 261–275). Springer Berlin / Heidelberg.
- Droste, P., Hadlich, F., Wiechert, W., and Qeli, E. 2008a. 3D Visualization of Simulation Data in a Network Context: A Case study from Systems Biology. In S. L. C. Y. Louca, Z. Oplatkova, and K. Al-Begain (Eds.) *in Proceedings of the ECMS 2008, Nicosia, Cyprus*, (pp. 227–233).
- Droste, P., Miebach, S., Niedenführ, S., Wiechert, W., and Nöh, K. 2011a. Visualizing multi-omics data in metabolic networks with the software Omix–A case study. *Biosystems*, 105(2), 154–161. Proceedings of the workshop Edinburgh, U. K., 15 October 2010 - IOMPA 2010.
- Droste, P., Noack, S., Nöh, K., and Wiechert, W. 2009a. Customizable Visualization of Multi-omics Data in the Context of Biochemical Networks. In *Proceedings of the Second International Conference in Visualization*, (pp. 21–25). Los Alamitos, CA, USA: IEEE Computer Society.

## Original Publications

---

- Droste, P., Nöh, K., and Wiechert, W. 2009b. Omix - Editor für biochemische Netzwerke und Visualisierungswerkzeug. In *ProcessNet-Jahrestagung und 27. Jahrestagung der Biotechnologen*, vol. 81, (pp. 1244–1245). WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim. (in German).
- Droste, P., Nöh, K., and Wiechert, W. 2010a. Omix – Ein höchst anpassbarer Editor für biochemische Netzwerke und Visualisierungswerkzeug für Multi-Omics-Datensätze. In *ProcessNet-Jahrestagung 2010 und 28. Jahrestagung der Biotechnologen*, vol. 82, (pp. 1554–1555). WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim. (in German).
- Droste, P., von Lieres, E., Wiechert, W., and Nöh, K. 2010b. Customizable Visualization on Demand for Hierarchically Organized Information in Biochemical Networks. In Reneta P. Barneva and Valentin E. Brimkov and Herbert A. Hauptman and Renato M. Natal Jorge and João Manuel R. S. Tavares (Ed.) *Computational Modeling of Objects Represented in Images*, vol. 6026 of *Lecture Notes in Computer Science*, (pp. 163–174). Springer Verlag.
- Droste, P., Weitzel, M., and Wiechert, W. 2008b. Visual Exploration of Isotope Labeling Networks in 3D. *Bioprocess and Biosystems Engineering*, 31(3), 227–239.
- Droste, P., Wiechert, W., and Nöh, K. 2011b. Semi-automatic drawing of metabolic networks. *Information Visualization*, in press.

## Further Publications

- Droste, P. 2006. *Entwicklung eines objektorientierten Frameworks zur Simulation und Visualisierung von SAR-Szenarien*. Diploma thesis (in German), Department: Dept. Simulation, FB 11, University of Siegen.
- Droste, P., Zimmermann, M., Küter, G., Hofmann, C., Klein, U., and Sedler, M. 2005. Entwicklung eines Visualisierungssystems für die Simulation von Hochenergie-Kugelmöhlen. Project thesis (in German), Department: Dept. Simulation, FB 11, University of Siegen.
- Kalkuhl, M., Droste, P., Wiechert, W., Knedlik, S., Nies, H., and Loffeld, O. 2006a. Simulation of Multistatic Synthetic Aperture Radar Scenarios: A Multisensoric Simulation Framework. In M. Becker, and H. Szczerbicka (Eds.) *ASIM - Simulationstechnik: 19th Symposium, Hannover*.
- Kalkuhl, M., Droste, P., Wiechert, W., Nies, H., and Loffeld, O. 2006b. Modular SAR Simulator for Bi- and Multistatic Constellations. In *IGARSS '06, Denver (Colorado), USA*, (pp. 1212–1215).
- Reichardt, R., Grollius, S., Droste, P., and Wiechert, W. 2007. When Simulation is as Complex as Reality: Visualization as a Data Evaluation Tool for Many-Particle Models. In B. Zupancic, R. Karba, and S. Blazic (Eds.) *Eurosim 2007, Ljubljana, Slovenia*.

## Bibliography

- Antoniewicz, M., Kelleher, J., and Stephanopoulos, G. 2007. Elementary metabolite units (EMU): A novel framework for modeling isotopic distributions. *Metabolic Engineering*, 9, 68–86.
- Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. 2000. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nature Genetics*, 25(1), 25–29.
- Attaway, S. 2009. *Matlab: A Practical Introduction to Programming and Problem Solving*. Butterworth-Heinemann.
- Batagelj, V., and Mrvar, A. 2002. Pajek – Analysis and Visualization of Large Networks. In *Graph Drawing*, vol. 2265 of *Lecture Notes in Computer Science*, (pp. 8–11). Springer Berlin / Heidelberg.
- Beard, D. A., Babson, E., Curtis, E., and Qian, H. 2004. Thermodynamic constraints for biochemical networks. *Journal of Theoretical Biology*, 228(3), 327 – 333.
- Becker, S., Price, N., and Palsson, B. 2006. Metabolite coupling in genome-scale metabolic networks. *BMC Bioinformatics*, 7(1), 111.
- Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. 2000. The Protein Data Bank. *Nucleic Acids Research*, 28(1), 235–242.
- Bourqui, R., Auber, D., Lacroix, V., and Jourdan, F. 2006. Metabolic network visualization using constraint planar graph drawing algorithm. In *Proceedings of the 10<sup>th</sup> International Conference on Information Visualisation (IV'06)*, (pp. 489–496). London, UK: IEEE Computer Society.
- Brandes, U., Dwyer, T., and Schreiber, F. 2004. Visualizing Related Metabolic Pathways in Two and a Half Dimensions. In G. Liotta (Ed.) *Graph Drawing*, vol. 2912 of *Lecture Notes in Computer Science*, (pp. 111–122). Springer Berlin / Heidelberg.
- Breitkreutz, B.-J., Stark, C., and Tyers, M. 2003. Osprey: a network visualization system. *Genome Biology*, 4(3), R22.

## Bibliography

---

- Byrnes, R., Cotter, D., Maer, A., Li, J., Nadeau, D., and Subramaniam, S. 2009. An editor for pathway drawing and data visualization in the biopathways workbench. *BMC Systems Biology*, 3(1), 99.
- Card, S. 2007. Information visualization. In J. Jacko, and A. Sears (Eds.) *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications*. Lawrence Erlbaum Associates Inc.
- Card, S. K., Mackinlay, J. D., and Shneiderman, B. 1999. *Readings in information visualization: using vision to think*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
- Cary, M., Pavlovic, V., Demir, E., Bader, G., and Donaldson, S. 2011. PathwayGuide: the pathway resource list. <http://www.pathguide.org>. Visited March 2011.
- Chassagnole, C., Noisommit-Rizzi, N., Schmid, J. W., Mauch, K., and Reuss, M. 2002. Dynamic modeling of the central carbon metabolism of *Escherichia coli*. *Biotechnology and Bioengineering*, 79(1), 53–73.
- Chen, C. 2005. Top 10 unsolved information visualization problems. *Computer Graphics and Applications, IEEE*, 25(4), 12 – 16.
- Christensen, B., and Nielsen, J. 1999. Isotopomer Analysis Using GC-MS. *Metabolic Engineering*, 1(4), 282 – 290.
- Chung, H.-J., Kim, M., Park, C. H., Kim, J., and Kim, J. H. 2004. ArrayXPath: mapping and visualizing microarray gene-expression data with integrated biological pathway resources using Scalable Vector Graphics. *Nucleic Acids Research*, 32(suppl 2), W460–W464.
- Cook, D., Farley, J., and Tapscott, S. 2001. A basis for a visual language for describing, archiving and analyzing functional models of complex biological systems. *Genome Biology*, 2(4), RESEARCH0012.
- Cortassa, S., Aon, M. A., Iglesias, A. A., and Lloyd, D. 2002. *An Introduction to Metabolic and Cellular Engineering*. World Scientific Publishing Company.
- Czauderna, T., Klukas, C., and Schreiber, F. 2010. Editing, validating and translating of SBGN maps. *Bioinformatics*, 26(18), 2340–2341.
- Dalby, A., Nourse, J. G., Hounshell, W. D., Gushurst, A. K. I., Grier, D. L., Leland, B. A., and Laufer, J. 1992. Description of several chemical structure file formats used by computer programs developed at molecular design limited. *Journal of Chemical Information and Computer Sciences*, 32(3), 244–255.
- Daldal, F. 1984. Nucleotide sequence of gene pfkB encoding the minor phosphofructokinase of *Escherichia coli* K-12. *Gene*, 28(3), 337 – 342.

- Dalman, T., Droste, P., Weitzel, M., Wiechert, W., and Nöh, K. 2010a. Workflows for Metabolic Flux Analysis: Data Integration and Human Interaction. In T. Margaria, and B. Steffen (Eds.) *Leveraging Applications of Formal Methods, Verification, and Validation*, vol. 6415 of *Lecture Notes in Computer Science*, (pp. 261–275). Springer Berlin / Heidelberg.
- Dalman, T., Juhnke, E., Dörnemann, T., Weitzel, M., Nöh, K., Wiechert, W., and Freisleben, B. 2010b. Service Workflows and Distributed Computing Methods for  $^{13}\text{C}$  Metabolic Flux Analysis. In *Proceedings of 7th EUROSIM Congress on Modelling and Simulation*.
- Demir, E., Babur, O., Dogrusoz, U., Gursoy, A., Nisanci, G., Cetin-Atalay, R., and Ozturk, M. 2002. Patika: an integrated visual environment for collaborative construction and analysis of cellular pathways. *Bioinformatics*, 18(7), 996–1003.
- Dickerson, J., Cox, Z., Wurtele, E., and Fulmer, A. 2001. Creating metabolic and regulatory network models using fuzzy cognitive maps. In *IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th*, vol. 4, (pp. 2171–2176).
- Dickerson, J., Yang, Y., Blom, K., Reinot, A., Lie, J., Cruz-Neira, C., and Wurtele, E. S. 2003. Using Virtual Reality to Understand Complex Metabolic Networks. In *Atlantic Symposium on Computational Biology and Genome Informatics, 7th Joint Conference on Information Science (JCIS)*, (pp. 950–953). Durham, North Carolina: JCIS/Association for Intelligent Machinery, Inc.
- Droste, P. 2008a. *Omix – User Manual*. Institute of Bio- and Geosciences, IBG-1 Biotechnology, Forschungszentrum Jülich, 52425 Jülich, Germany.
- Droste, P. 2008b. *Omix Visualization Language - Technical Manual*. Institute of Bio- and Geosciences, IBG-1 Biotechnology, Forschungszentrum Jülich, 52425 Jülich, Germany.
- Droste, P. 2010. Omix Library Reference Documentation. [www.13cflux.net/omix/doc](http://www.13cflux.net/omix/doc). Visited March 2011.
- Droste, P. 2011. *Omix Plug-in Development Manual*. Institute of Bio- and Geosciences, IBG-1 Biotechnology, Forschungszentrum Jülich, 52425 Jülich, Germany.
- Droste, P., Hadlich, F., Wiechert, W., and Qeli, E. 2008a. 3D Visualization of Simulation Data in a Network Context: A Case study from Systems Biology. In S. L. C. Y. Louca, Z. Oplatkova, and K. Al-Begain (Eds.) *Proceedings of the ECMS 2008, Nicosia, Cyprus*, (pp. 227–233).
- Droste, P., Miebach, S., Niedenführ, S., Wiechert, W., and Nöh, K. 2011a. Visualizing multi-omics data in metabolic networks with the software Omix—A case study. *Biosystems*, 105(2), 154–161.  
URL <http://www.sciencedirect.com/science/article/pii/S0303264711000761>

## Bibliography

---

- Droste, P., Noack, S., Nöh, K., and Wiechert, W. 2009a. Customizable Visualization of Multi-omics Data in the Context of Biochemical Networks. In *Proceedings of the Second International Conference in Visualization*, (pp. 21–25). Los Alamitos, CA, USA: IEEE Computer Society.
- Droste, P., Nöh, K., and Wiechert, W. 2009b. Omix - Editor für biochemische Netzwerke und Visualisierungswerkzeug. In *ProcessNet-Jahrestagung und 27. Jahrestagung der Biotechnologen*, vol. 81, (pp. 1244–1245). WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim. (in German).
- Droste, P., Nöh, K., and Wiechert, W. 2010a. Omix – Ein höchst anpassbarer Editor für biochemische Netzwerke und Visualisierungswerkzeug für Multi-Omics-Datensätze. In *ProcessNet-Jahrestagung 2010 und 28. Jahrestagung der Biotechnologen*, vol. 82, (pp. 1554–1555). WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim. (in German).
- Droste, P., von Lieres, E., Wiechert, W., and Nöh, K. 2010b. Customizable Visualization on Demand for Hierarchically Organized Information in Biochemical Networks. In Reneta P. Barneva and Valentin E. Brimkov and Herbert A. Hauptman and Renato M. Natal Jorge and João Manuel R. S. Tavares (Ed.) *Computational Modeling of Objects Represented in Images*, vol. 6026 of *Lecture Notes in Computer Science*, (pp. 163–174). Springer Verlag.
- Droste, P., Weitzel, M., and Wiechert, W. 2008b. Visual Exploration of Isotope Labeling Networks in 3D. *Bioprocess and Biosystems Engineering*, 31(3), 227–239.
- Droste, P., Wiechert, W., and Nöh, K. 2011b. Semi-automatic drawing of metabolic networks. *Information Visualization*, in press.
- Dumas, J. 2007. The great leap forward: The birth of the usability profession (1988-1993). *Journal of Usability Studies*, 2, 54–60.
- Eades, P. A. 1984. A heuristic for graph drawing. *Congressus Numerantium*, 42, 149–160.
- Elmqvist, H. 1978. *A Structured Model Language for Large Continuous Systems*. Ph.D. thesis, Dept. of Automatic Control, Lund Institute of Technology, Lund Sweden.
- Flórez, L. A., Lammers, C. R., Michna, R., and Stülke, J. 2010. CellPublisher: a web platform for the intuitive visualization and sharing of metabolic, signalling and regulatory pathways. *Bioinformatics*, 26(23), 2997–2999.
- Forman, I. R., Forman, N., and Vlissides, J. 2004. *Java Reflection in Action*. Manning Publications, Greenwich.
- Freeman, T. C., Goldovsky, L., Brosch, M., van Dongen, S., Mazière, P., Grocock, R. J., Freilich, S., Thornton, J., and Enright, A. J. 2007. Construction, Visualisation, and Clustering of Transcription Networks from Microarray Expression Data. *PLoS Comput Biol*, 3(10), e206.



- Frick, O., and Wittmann, C. 2005. Characterization of the metabolic shift between oxidative and fermentative growth in *Saccharomyces cerevisiae* by comparative  $^{13}\text{C}$  flux analysis. *Microbial Cell Factories*, 4, 1–16. 10.1186/1475-2859-4-30.
- Fritzson, P., Aronsson, P., Lundvall, H., Nyström, K., Pop, A., Saldamli, L., and Broman, D. 2006. OpenModelica - A free open-source environment for system modeling, simulation, and teaching. In *Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE*, (pp. 1588–1595).
- Fruchterman, T. M. J., and Reingold, E. M. 1991. Graph drawing by force-directed placement. *Software - Practice and Experience*, 21(11), 1129–1164.
- Funahashi, A., Morohashi, M., Kitano, H., and Tanimura, N. 2003. CellDesigner: a process diagram editor for gene-regulatory and biochemical networks. *BIOSILICO*, 1(5), 159–162.
- García-Alcalde, F., García-Lopez, F., Dopazo, J., and Conesa, A. 2010. Paintomics: a web based tool for the joint visualization of transcriptomics and metabolomics data. *Bioinformatics*.
- Gehlenborg, N., O’Donoghue, S. I., Baliga, N. S., Goesmann, A., Hibbs, M. A., Kitano, H., Kohlbacher, O., Neuweger, H., Schneider, R., Tenenbaum, D., and Gavin, A.-C. 2010. Visualization of omics data for systems biology. *Nature Methods*, 7, S56–S68.
- Google Inc. 2011. Google Maps Javascript API. <http://code.google.com/apis/maps/documentation/javascript/>. Visited March 2011.
- Gosling, J., Joy, B., Steele, G., and Bracha, G. 2005. *Java™ Language Specification, Third Edition*. Addison Wesley.
- Gould, J. D. 1988. *How to design usable systems*, (pp. 757–789). Handbook of Human-Computer Interaction. Morgan Kaufmann Publishers Inc.
- Government of the Federal Republic of Germany 2011. Förderkatalog. <http://foerderportal.bund.de/foekat/>. Visited March 2011.
- Holford, M., Li, N., Nadkarni, P., and Zhao, H. 2005. VitaPad: visualization tools for the analysis of pathway data. *Bioinformatics*, 21(8), 1596–1602.
- Hu, Z., Mellor, J., Wu, J., Yamada, T., Holloway, D., and DeLisi, C. 2005. Visant: data-integrating visual framework for biological networks and modules. *Nucleic Acids Research*, 33, W352–357.
- Hu, Z., Ng, D. M., Yamada, T., Chen, C., Kawashima, S., Mellor, J., Linghu, B., Kanehisa, M., Stuart, J. M., and DeLisi, C. 2007. VisANT 3.0: new modules for pathway visualization, editing, prediction and construction. *Nucleic Acids Research*, 35(suppl 2), W625–W632.

## Bibliography

---

- Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., et al. 2003. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4), 524–531.
- iGEM Team 2010 TU Delft 2010. Pathway Visualization. [http://2010.igem.org/Team:TU\\_Delft#page=Modeling/MFA](http://2010.igem.org/Team:TU_Delft#page=Modeling/MFA). Visited March 2011.
- Iragne, F., Nikolski, M., Mathieu, B., Auber, D., and Sherman, D. 2005. ProViz: protein interaction visualization and exploration. *Bioinformatics*, 21(2), 272–274.
- Ishii, N., Nakahigashi, K., Baba, T., Robert, M., Soga, T., Kanai, A., Hirasawa, T., Naba, M., Hirai, K., Hoque, A., Ho, P. Y., Kakazu, Y., Sugawara, K., Igarashi, S., Harada, S., Masuda, T., Sugiyama, N., Togashi, T., Hasegawa, M., Takai, Y., Yugi, K., Arakawa, K., Iwata, N., Toya, Y., Nakayama, Y., Nishioka, T., Shimizu, K., Mori, H., and Tomita, M. 2007. Multiple High-Throughput Analyses Monitor the Response of *E. coli* to Perturbations. *Science*, 316(5824), 593–597.
- ISO 9241-11 1998. Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability.
- ISO 9241-110 2006. Ergonomics of human-system interaction – Part 110: Dialogue principles.
- Junker, B., Klukas, C., and Schreiber, F. 2006. Vanted: A system for advanced data analysis and visualization in the context of biological networks. *BMC Bioinformatics*, 7(1), 109.
- Kanani, H., Chrysanthopoulos, P. K., and Klapa, M. I. 2008. Standardizing GC-MS metabolomics. *Journal of Chromatography B-Analytical Technologies in the Biomedical and Life Sciences*, 871(2), 191–201.
- Kanehisa, M., and Goto, S. 2000. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, 28(1), 27–30.
- Kanehisa, M., Goto, S., Hattori, M., Aoki-Kinoshita, K. F., Itoh, M., Kawashima, S., Katayama, T., Araki, M., and Hirakawa, M. 2005. From genomics to chemical genomics: new developments in KEGG. *Nucleic Acids Research*, 34(suppl 1), D354–D357.
- Karp, P. D., Ouzounis, C. A., Moore-Kochlacs, C., Goldovsky, L., Kaipa, P., Ahrén, D., Tsoka, S., Darzentas, N., Kunin, V., and López-Bigas, N. 2005. Expansion of the BioCyc collection of pathway/genome databases to 160 genomes. *Nucleic Acids Research*, 33(19), 6083–6089.
- Khan, M. T. H., and Ather, A. 2006. Metabolomics – systematic studies of the metabolic profiling. *Advances in Phytomedicine*, 2, 411–419.

- Kiefer, P., Heinzle, E., Zelder, O., and Wittmann, C. 2004. Comparative metabolic flux analysis of lysine-producing *Corynebacterium glutamicum* cultured on glucose or fructose. *Applied and Environmental Microbiology*, 70(1), 229–239.
- Killcoyne, S., and Boyle, J. 2009. Managing Chaos: Lessons Learned Developing Software in the Life Sciences. *Computing in Science and Engineering*, 11(6), 20–29.
- Kitano, H. 2002. Systems Biology: A Brief Overview. *Science*, 295(5560), 1662–1664.
- Kitano, H., Funahashi, A., Matsuoka, Y., and Oda, K. 2005. Using process diagrams for the graphical representation of biological networks. *Nature Biotechnology*, 23(8), 961–966.
- Kitano, H., Oda, K., Kimura, T., Matsuoka, Y., Csete, M., Doyle, J., and Muramatsu, M. 2004. Metabolic Syndrome and Robustness Tradeoffs. *Diabetes*, 53(suppl 3), S6–S15.
- Klamt, S., Saez-Rodriguez, J., and Gilles, E. 2007. Structural and functional analysis of cellular networks with cellnetanalyzer. *BMC Systems Biology*, 1(1), 2.
- Klapa, M. I., Park, S. M., Sinskey, A. J., and Stephanopoulos, G. 1999. Metabolite and isotopomer balancing in the analysis of metabolic cycles: I. Theory. *Biotechnology and Bioengineering*, 62(4), 375–391.
- Köhler, J., Baumbach, J., Taubert, J., Specht, M., Skusa, A., Rüegg, A., Rawlings, C., Verrier, P., and Philippi, S. 2006. Graph-based analysis and visualization of experimental results with ONDEX. *Bioinformatics*, 22(11), 1383–1390.
- Kohn, K. W., Aladjem, M. I., Weinstein, J. N., and Pommier, Y. 2006. Molecular interaction maps of bioregulatory networks: A general rubric for systems biology. *Molecular biology of the cell*, 17(1), 1–13.
- Kolpakov, F. A., Ananko, E. A., Kolesov, G. B., and Kolchanov, N. A. 1998. GeneNet: a gene network database and its automated visualization. *Bioinformatics*, 14(6), 529–537.
- Koshland, D. E. 2002. The Seven Pillars of Life. *Science*, 295(5563), 2215–2216.
- Kummel, A., Panke, S., and Heinemann, M. 2006. Putative regulatory sites unraveled by network-embedded thermodynamic analysis of metabolome data. *Molecular Systems Biology*, 2.
- Le Novere, N., et al. 2009. The Systems Biology Graphical Notation. *Nature Biotechnology*, 27(8), 735–741.
- Lederberg, J., and McCray, A. T. 2001. 'Ome Sweet 'Omics— A Genealogical Treasury of Words. *The Scientist*, 17(7).

## Bibliography

---

- Li, W., and Kurata, H. 2005. A grid layout algorithm for automatic drawing of biochemical networks. *Bioinformatics*, 21(9), 2036–2042.
- Luo, B., Groenke, K., Takors, R., Wandrey, C., and Oldiges, M. 2007. Simultaneous determination of multiple intracellular metabolites in glycolysis, pentose phosphate pathway and tricarboxylic acid cycle by liquid chromatography-mass spectrometry. *Journal of Chromatography A*, 1147(2), 153–164.
- Madigan, M. T., Martinko, J. M., and Parker, J. 2003. *Brock biology of microorganisms*, vol. 10. Pearson Education.
- Maskow, T., and von Stockar, U. 2005. How reliable are thermodynamic feasibility statements of biochemical pathways? *Biotechnology and Bioengineering*, 92(2), 223–230.
- Michal, G. 1968-2005. Biochemical Pathways. (wall poster) <http://www.expasy.ch/tools/pathways/>. Visited March 2011.
- Michal, G. 1998. On representation of metabolic pathways. *Biosystems*, 47(1-2), 1 – 7.
- Michal, G. 1999. *Biochemical Pathways - An Atlas of Biochemistry and Molecular Biology*. Wiley.
- Mlecnik, B., Scheideler, M., Hackl, H., Hartler, J., Sanchez-Cabo, F., and Trajanoski, Z. 2005. PathwayExplorer: web service for visualizing high-throughput expression data on biological pathways. *Nucleic Acids Research*, 33(suppl 2), W633–W637.
- Müller, D., Aguilera-Vázquez, L., Reuss, M., and Mauch, K. 2005. Integration of metabolic and signaling networks. In L. Alberghina, and H. Westerhoff (Eds.) *Systems Biology*, vol. 13 of *Topics in Current Genetics*, (pp. 227–241). Springer Berlin / Heidelberg. 10.1007/b136529.
- Nagasaki, M., Saito, A., Jeong, E., Li, C., Kojima, K., Ikeda, E., and Miyano, S. 2010. Cell Illustrator 4.0: A Computational Platform for Systems Biology. *In Silico Biology*, 10, 5–26.
- Neuweger, H., Persicke, M., Albaum, S., Bekel, T., Dondrup, M., Huser, A., Winnebald, J., Schneider, J., Kalinowski, J., and Goesmann, A. 2009. Visualizing post genomics data-sets on customized pathway maps by ProMeTra - aeration-dependent gene expression and metabolism of *Corynebacterium glutamicum* as an example. *BMC Systems Biology*, 3(1), 82.
- Nilsson, E. L., and Fritzson, P. 2005. A Metabolic Specialization of a General Purpose Modelica Library for Biological and Biochemical Systems. In G. Schmitz (Ed.) *Proceedings of the 4th International Modelica Conference*, vol. 1, (pp. 85–93). Hamburg, Germany.
- Noack, S., Wahl, A., Qeli, E., and Wiechert, W. 2007. Visualizing regulatory interactions in metabolic networks. *BMC Biology*, 5(1), 46.

- Nöh, K., Weitzel, M., and Wiechert, W. 2008. From Isotope Labeling Patterns to Metabolic Flux Rates. In U. H. E. Hansmann, J. H. Meinke, S. Mohanty, W. Nadler, and O. Zimmermann (Eds.) *From Computational Biophysics to Systems Biology (CBSB08)*, vol. 40 of *NIC Series*, (pp. 345–348).
- Nokia Corporation 2011. Qt. <http://qt.nokia.com>. Visited March 2011.
- O'Donoghue, S., Gavin, A.-C., Gehlenborg, N., Goodsell, D. S., Hériché, J.-K., Nielsen, C. B., North, C., Olson, A. J., Procter, J. B., Shattuck, D. W., Walter, T., and Wong, B. 2010. Visualizing biological data – now and in the future. *Nature Methods*, 7, S2–S4.
- Okuda, S., Yamada, T., Hamajima, M., Itoh, M., Katayama, T., Bork, P., Goto, S., and Kanehisa, M. 2008. KEGG Atlas mapping for global analysis of metabolic pathways. *Nucleic Acids Research*, 36(suppl 2), W423–W426.
- Oldiges, M., Lütz, S., Pflug, S., Schroer, K., Stein, N., and Wiendahl, C. 2007. Metabolomics: current state and evolving methodologies and tools. *Applied Microbiology and Biotechnology*, 76, 495–511. 10.1007/s00253-007-1029-2.
- Oracle Corporation 2011. Java SE 7 Documentation. <http://download.oracle.com/javase/7/docs/>. Visited March 2011.
- Orlev, N., Shamir, R., and Shiloh, Y. 2004. PIVOT: Protein Interactions Visualization Tool. *Bioinformatics*, 20(3), 424–425.
- Orth, J. D., Thiele, I., and Palsson, B. O. 2010. What is flux balance analysis? *Nature Biotechnology*, 28(3), 245–248.
- Paek, E., Park, J., and Lee, K.-J. 2004. Multi-layered Representation for Cell Signaling Pathways. *Molecular & Cellular Proteomics*, 3(10), 1009–1022.
- Papin, J. A., Price, N. D., Wiback, S. J., Fell, D. A., and Palsson, B. O. 2003. Metabolic pathways in the post-genome era. *Trends in Biochemical Sciences*, 28(5), 250 – 258.
- Parr, T. J., and Quong, R. W. 1995. ANTLR: A predicated-LL(k) parser generator. *Software – Practice and Experience*, 25(7), 789–810.
- Pavlopoulos, G., Wegener, A.-L., and Schneider, R. 2008. A survey of visualization tools for biological network analysis. *BioData Mining*, 1(1), 12.
- Pico, A. R., Kelder, T., van Iersel, M. P., Hanspers, K., Conklin, B. R., and Evelo, C. 2008. WikiPathways: Pathway Editing for the People. *PLoS Biology*, 6(7), e184.
- Pirson, I., Fortemaison, N., Jacobs, C., Dremier, S., Dumont, J. E., and Maenhaut, C. 2000. The visual display of regulatory information and networks. *Trends in Cell Biology*, 10(10), 404 – 408.

## Bibliography

---

- Purchase, H. C. 1997. Which aesthetic has the greatest effect on human understanding? In G. D. Battista (Ed.) *Proceedings of the 5th International Symposium on Graph Drawing*, Lecture Notes In Computer Science, (pp. 248 – 261). London, UK: Springer.
- Qeli, E., Wahl, A., Degenring, D., Wiechert, W., and Freisleben, B. 2003. MetVis: A tool for designing and animating metabolic networks. In *Proceedings of the 2003 European Simulation and Modelling Conference*. Naples, Italy: Eurosis Press.
- Qeli, E., Wiechert, W., and Freisleben, B. 2005. Visual exploration of time-varying matrices. In *Information Visualisation, 2005. Proceedings. Ninth International Conference on*, (pp. 889 – 895).
- Qian, H., and Beard, D. A. 2005. Thermodynamics of stoichiometric biochemical networks in living systems far from equilibrium. *Biophysical Chemistry*, 114(2-3), 213 – 220.
- Qt Jambi Team 2011. Qt Jambi. <http://qt-jambi.org>. Visited March 2011.
- Reiß, T. 2002. *Systeme des Lebens – Systembiologie*. 53170 Bonn, Germany: Bundesministerium für Bildung und Forschung (BMBF). (in German).
- Rohrschneider, M., Heine, C., Reichenbach, A., Kerren, A., and Scheuermann, G. 2009. A Novel Grid-based Visualization Approach for Metabolic Networks with Advanced Focus&Context View. In *Proceedings of the 17th International Symposium on Graph Drawing (GD '09)*, LNCS. Springer.
- Salomon, D. 2006. *Curves and Surfaces for Computer Graphics*. Springer-Verlag.
- Saraiya, P., North, C., and Duca, K. 2005. Visualizing biological pathways: requirements analysis, systems evaluation and research agenda. *Information Visualization*, 4(3), 191–205.
- Sauer, U. 2004. High-throughput phenomics: experimental methods for mapping fluxomes. *Current Opinion in Biotechnology*, 15, 58–63.
- Sauro, H. M., Hucka, M., Finney, A., Wellock, C., Bolouri, H., Doyle, J., and Kitano, H. 2003. Next Generation Simulation Tools: The Systems Biology Workbench and BioSPICE Integration. *OMICS: A Journal of Integrative Biology*, 7(4), 355–372.
- Schomburg, I., Chang, A., Hofmann, O., Ebeling, C., Ehrentreich, F., and Schomburg, D. 2002. BRENDA: a resource for enzyme data and metabolic information. *Trends in Biochemical Sciences*, 27(1), 54–56.
- Schroer, K., Zelic, B., Oldiges, M., and Lütz, S. 2009. Metabolomics for biotransformations: Intracellular redox cofactor analysis and enzyme kinetics offer insight into whole cell processes. *Biotechnology and Bioengineering*, 104(2), 251–260.
- Schwikowski, B., Uetz, P., and Fields, S. 2000. A network of protein-protein interactions in yeast. *Nature Biotechnology*, 18(12), 1257–1261.

- Scott, M. L. 2009. *Programming language pragmatics*, vol. 3. Morgan Kaufmann.
- Shannon, P., Markiel, A., Ozier, O., Baliga, N. S., Wang, J. T., Ramage, D., Amin, N., Schwikowski, B., and Ideker, T. 2003. Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. *Genome Research*, 13(11), 2498–2504.
- Sharma, G., and Martin, J. 2009. MATLAB®: A Language for Parallel Computing. *International Journal of Parallel Programming*, 37, 3–36. 10.1007/s10766-008-0082-5.
- Shoemaker, D. D., and Linsley, P. S. 2002. Recent developments in DNA microarrays. *Current Opinion in Microbiology*, 5(3), 334 – 337.
- Sletta, G. 2006. Meet Qt Jambi. *Qt Quarterly*, 20(Q4 2006).
- Sorokin, A., Paliy, K., Selkov, A., Demin, O. V., Dronov, S., Ghazal, P., and Goryanin, I. 2006. The Pathway Editor: A tool for managing complex biological networks. *IBM Journal of Research and Development*, 50(6), 561 – 573.
- Stein, S. E., Heller, S. R., and Tchekhovskoi, D. 2003. An Open Standard for Chemical Structure Representation: The IUPAC Chemical Identifier. In *Proceedings of the International Chemical Information Conference (Nimes) Infonortics*, (pp. 131–143).
- Stephanopoulos, G. 1994. Metabolic engineering. *Current Opinion in Biotechnology*, 5(2), 196 – 200.
- Stephanopoulos, G. N., Aristidou, A. A., and Nielsen, J. 1998. *Metabolic Engineering: Principles and Methodologies*. Academic Press.
- Streit, M., Lex, A., Kalkusch, M., Zatloukal, K., and Schmalstieg, D. 2009. Caleydo: connecting pathways and gene expression. *Bioinformatics*, 25(20), 2760–2761.
- Stryer, L. 1995. *Biochemistry*. W.H. Freeman Company.
- Suderman, M., and Hallett, M. 2007. Tools for Visually Exploring Biological Networks. *Bioinformatics*, (20), btm401.
- Sugiyama, K., Tagawa, S., and Toda, M. 1981. Methods for visual understanding of hierarchical systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(2), 109–125.
- Szyperski, T. 1998. <sup>13</sup>C-NMR, MS and metabolic flux balancing in biotechnology research. *Quarterly Reviews of Biophysics*, 31(01), 41–106.
- Tao, Y., Friedman, C., and Lussier, Y. A. 2004. Visualizing information across multidimensional post-genomic structured and textual databases. *Bioinformatics*, 21(8), 1659–1667.

## Bibliography

---

- Terzer, M., and Stelling, J. 2008. Large-scale computation of elementary flux modes with bit pattern trees. *Bioinformatics*, 24(19), 2229–2235.
- Tiller, M. 2001. *Introduction to Physical Modeling with Modelica*. The Springer International Series in Engineering and Computer Science. Springer.
- Trost, E., Hackl, H., Maurer, M., and Trajanoski, Z. 2003. Java editor for biological pathways. *Bioinformatics*, 19(6), 786–787.
- Vemuri, G. N., and Aristidou, A. A. 2005. Metabolic engineering in the -omics era: elucidating and modulating regulatory networks. *Microbiology and Molecular Biology Reviews*, 69(2), 197–216.
- Venter, J. C., Adams, M. D., Myers, E. W., et al. 2001. The sequence of the human genome. *Science*, 291(5507), 1304–1351.
- Villas-Bôas, S. G., Moxley, J. F., Akesson, M., Stephanopoulos, G., and Nielsen, J. 2005. High-throughput metabolic state analysis: the missing link in integrated functional genomics of yeasts. *Biochemical Journal*, 388(2), 669–677.
- Villéger, A. C., Pettifer, S. R., and Kell, D. B. 2010. Arcadia: a visualization tool for metabolic pathways. *Bioinformatics*, 26(11), 1470–1471.
- Vinopal, R. T., Clifton, D., and Fraenkel, D. G. 1975. PfkA locus of Escherichia coli. *Journal of Bacteriology*, 122(3), 1162 – 1171.
- Wang, W., Hollmann, R., Furch, T., Nimtz, M., Malten, M., Jahn, D., and Deckwer, W.-D. 2005. Proteome analysis of a recombinant bacillus megaterium strain during heterologous production of a glucosyltransferase. *Proteome Science*, 3(1), 4.
- Weitzel, M. 2009. *High Performance Algorithms for Metabolic Flux Analysis*. Ph.D. thesis, University of Siegen, Germany.
- Weitzel, M., Wiechert, W., and Nöh, K. 2007. The topology of metabolic isotope labeling networks. *BMC Bioinformatics*, 8(1), 315.
- White, H. B. 2001. CHEM-643 INTERMEDIARY METABOLISM Metabolic Pathway Resource Page. <http://www.udel.edu/chem/white/C643/MetabPathways.html>. Visited March 2011.
- Wiechert, W. 2001. <sup>13</sup>C Metabolite Flux Analysis. *Metabolic Engineering*, 3, 195–206.
- Wiechert, W., Mollney, M., Isermann, N., Wurzel, M., and de Graaf, A. 1999. Bidirectional reaction steps in metabolic networks: III. Explicit solution and analysis of isotopomer labeling systems. *Biotechnology and Bioengineering*, 66(2), 69–85.



- Wiklund, S., Johansson, E., Sjöström, L., Mellerowicz, E. J., Edlund, U., Shockcor, J. P., Gottfries, J., Moritz, T., and Trygg, J. 2008. Visualization of GC/TOF-MS-based metabolomics data for identification of biochemically interesting compounds using OPLS class models. *Analytical Chemistry*, 80(1), 115–122.
- Wolkenhauer, O. 2001. Systems biology: The reincarnation of systems theory applied in biology? *Briefings in Bioinformatics*, 2(3), 258–270.
- Yang, Y., Engin, L., Wurtele, E. S., Cruz-Neira, C., and Dickerson, J. A. 2005. Integration of metabolic networks and gene expression in virtual reality. *Bioinformatics*, 21(18), 3645–3650.
- Yang, Y., Wurtele, E. S., Cruz-Neira, C., and Dickerson, J. A. 2006. Hierarchical visualization of metabolic networks using virtual reality. In *Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, VRCIA '06, (pp. 377–381). New York, NY, USA: ACM.
- Yao, D., Wang, J., Lu, Y., Noble, N., Sun, H., Zhu, X., Lin, N., Payan, D. G., Li, M., and Qu, K. 2004. PathwayFinder: paving the way towards automatic pathway extraction. In *Proceedings of the second conference on Asia-Pacific bioinformatics*, vol. 29, (pp. 53–62). Australian Computer Society, Inc.
- Zaparty, M. e. a. 2010. “Hot standards” for the thermoacidophilic archaeon *Sulfolobus solfataricus*. *Extremophiles*, 14, 119–142.
- Zupan, B., Demsar, J., Bratko, I., Juvan, P., Halter, J. A., Kuspa, A., and Shaulsky, G. 2003. GenePath: a system for automated construction of genetic networks from mutant data. *Bioinformatics*, 19(3), 383–389.