Analyse und Visualisierung von Experimentdaten im Kontext biologischer Netzwerke

Dissertation

zur Erlangung des akademischen Grades
doctor rerum naturalium (Dr. rer. nat.)

vorgelegt der
Naturwissenschaftlichen Fakultät III der
Martin-Luther-Universität Halle-Wittenberg

von Herrn
Diplom-Wirtschaftsinformatiker Christian Klukas,
geb. am 25.08.1977 in Magdeburg

Gutachter:

Prof. Dr. habil. Falk Schreiber

Prof. Dr. habil. Ralf Hofestädt

Verteidigung am:

8. Oktober 2009

Magdeburg, im Mai 2009

Inhaltsverzeichnis

1 Einführung

1.1 Problembeschreibung und Ziele der Arbeit

Die in der biologischen Forschung vielfältig angewandten, stetig verbesserten Analyseverfahren sind die Basis für einen umfassenden Blick auf die Biochemie der untersuchten Organismen. Der Fokus der Forschung liegt zunehmend nicht nur in der Betrachtung interessanter Einzelphänomene, vielmehr stehen vielfältige Ursache–Wirkungsbeziehungen im Mittelpunkt der Untersuchungen. Die Aufklärung solch komplexer Zusammenhänge setzt die übergreifende Betrachtung verschiedener Domänen, wie der genetischen, biochemischen und phänotypischen Ebene, voraus. Diese Vorgehensweise ist für das Forschungsgebiet Systembiologie charakteristisch. Die Systembiologie stützt sich dabei in hohem Maße auf Methoden und Algorithmen, die durch die Bioinformatikforschung entwickelt werden. Nur durch die Nutzung spezialisierter Algorithmen können die immer umfangreicheren biologischen Messdaten verwaltet und ausgewertet werden.

Viele Bioinformatikprojekte stehen in engem Zusammenhang mit konkreten biologischen Fragestellungen. Der Fokus liegt dann beispielsweise in der Etablierung einer spezialisierten Datenbank und unterstützenden Analyse- und Visualisierungsmethoden für bestimmte Organismen oder ausgewählte Datendomänen. Solche Projekte sind somit in ihrer Anwendung limitiert und unflexibel. Da bisher nur wenige Forschungsprojekte zur Entwicklung von Systemen geführt haben, welche flexibel und domänenübergreifend Messdaten netzwerkintegriert visualisieren können, bietet dieser Bereich vielfältige Anknüpfungspunkte für die weitere Forschung. Das Ziel der vorliegenden Arbeit ist die Entwicklung von flexibel einsetzbaren Methoden zur netzwerkintegrierten Visualisierung, Exploration und Analyse von komplex strukturierten Experimentdaten. Eine benutzerfreundliche, leicht zugängliche Implementation der entwickelten Methodik soll Biologen in die Lage versetzen, umfangreiche Messdaten auf einfache Art und Weise mit biologischen Netzwerken oder Klassifikationshierarchien zu verknüpfen und integriert zu betrachten. Grundlegende, direkt einsetzbare statistische Funktionen sollen die Datenanalyse erleichtern und beschleunigen. Die Ergebnisse der interaktiven Datenexploration, anschauliche Visualisierungen für Vorträge und Veröffentlichung sollen idealerweise direkt am eigenen Arbeitsplatz-PC erstellt und aus dem Softwaresystem exportiert werden können.

1.2 Struktur der Arbeit

Kapitel 2 gliedert sich in drei Unterabschnitte. Ziel des Kapitels 2.1 ist, einen kurzen Überblick über die Grundlagen des Lebens aus Sicht der Molekularbiologie zu geben.

Es werden die drei biologischen „Domänen“ Genomics, Proteomics und Metabolomics und dazu jeweils relevante biologische Netzwerke und Klassifikationssysteme sowie experimentelle Messmethoden vorgestellt. In Kapitel 2.2 folgt die Beschreibung statistischer Grundlagen zur Auswertung von Experimenten. Der dritte Unterabschnitt 2.3 stellt grundlegende Begriffe, Datenstrukturen und Algorithmen für die Bereiche Graphentheorie und Informationsvisualisierung vor. Diese drei thematischen Komplexe bilden die Grundlage für das Verständnis von Kapitel 3. In Kapitel 3 wird eine Methodik vorgestellt, welche domänenübergreifend komplex strukturierte Experimentdaten mit den Elementen biologischer Netzwerke oder Klassifikationshierarchien in Beziehung setzt. Dies bildet die Basis für die Entwicklung und Vorstellung flexibler Visualisierungs-, Interaktions- und Analysemethoden. Die Umsetzung dieser Methodik in Form eines interaktiven Anwendungssystems ermöglicht die automatisierte Erzeugung von Darstellungen, welche einerseits leicht verständlich, anderseits aber hinreichend detailliert sind, um Hypothesen zu biologischen Fragestellungen auf intuitive Weise stützen oder verwerfen zu können. Dazu werden in Kapitel 4 ausgewählte Aspekte der Implementation in Form der Visualisierungs- und Analysesoftware VANTED[1] vorgestellt. Kapitel 5 hat die Bearbeitung biologischer Fragestellungen mit VANTED zum Thema und zeigt so die Nützlichkeit der entwickelten Methodik. Kapitel 6 fasst die Ergebnisse der vorliegenden Arbeit zusammen und gibt einen Ausblick auf sich aus dieser Arbeit ergebenden weiteren Forschungsarbeiten.

[1] VANTED steht für „Visualization and Analysis of Networks containing Experimental Data“. Das Anwendungssystem sowie eine ausführliche Dokumentation sind unter der URL `http://vanted.ipk-gatersleben.de` veröffentlicht, der Quelltext ist unter anderem mittels CVS-Zugriff von SourceForge abrufbar (`http://vanted.sourceforge.net`).

2 Grundlagen

Ziel der vorliegenden Arbeit ist unter anderem die Entwicklung von Methoden zur vereinfachten Visualisierung und Analyse von Experimentdaten im Bereich Systembiologie. Dazu wird im ersten Teil dieses Kapitels ein Überblick über die für die Systembiologie essentiellen Teilbereiche Genomics, Proteomics und Metabolomics gegeben. Zur Analyse der für diese Gebiete anfallenden experimentellen Messdaten werden oft statistische Methoden eingesetzt. Die für das Verständnis von im weiteren Verlauf der Arbeit benötigten statistischen Grundbegriffe sowie Grundlagen zum Design biologischer Experimente und zu Hypothesentests werden im zweiten Unterabschnitt dieses Kapitels vermittelt. Für die Systembiologie sind biologische Netzwerke und Klassifikationshierarchien häufig integraler Bestandteil biologischer Fragestellungen, formal lassen sich entsprechende Netzwerke im Computer als Graphen modellieren. Die Grundlagen dazu sowie damit zusammenhängende Aspekte wie Graphlayout und Informationsvisualisierung werden im dritten Unterabschnitt dieses Kapitels vorgestellt.

2.1 Grundlagen der Systembiologie

Die Fachgebiete Genomics, Proteomics und Metabolomics, welche sich mit der Analyse von Genen, Proteinen und Metaboliten beschäftigen, sind für das Verständnis biologischer Fragestellungen von großer Bedeutung. Das Ziel der Systembiologie ist, diese oft isoliert betrachteten „Welten“ zu einem Gesamtbild zusammenzuführen. Zum Verständnis der Biologie auf Systemebene sollten dazu nicht nur interessante Einzelphänomene, sondern Struktur und Dynamik vielfältiger Ursache–Wirkungsbeziehungen auf zellulärer und auf Ebene der Organismen untersucht werden [1]. Immer häufiger wird dazu ausgehend von einer Gesamtübersicht über die umfangreichen Datensätze eine „Top–Down“ Vorgehensweise zur Betrachtung interessanter Einzelphänomene im Kontext relevanter Abhängigkeiten verwendet.

Es folgt eine kurze Beschreibung der Forschungsgebiete Genomics, Proteomics und Metabolomics, dazu jeweils relevante Klassifikationshierarchien, biologische Netzwerke sowie ausgewählte Hochdurchsatzmessmethoden.

2.1.1 Genomics

Die molekulare Genetik beschäftigt sich mit den biophysikalischen und biochemischen Grundlagen des Lebens. Das Forschungsgebiet Genomics hat die umfassende Entschlüsselung der Bedeutung der genetischen Information zum Ziel. Äußerst komplexe und unvollständig verstandene molekulare Regelkreise bestimmen ausgehend von der die genetischen Informationen tragenden DNA die Funktion der einzelnen

Zellen und somit auch das Leben ganzer Organismen. Die DNA liegt bei Eukaryoten[1] im Zellkern und bei Prokaryoten[2] in der kernlosen Zelle vor. Sie hat eine von Watson und Crick 1953 vorgestellte [2] dreidimensionale Struktur, bestehend aus zwei in Form einer Doppelhelix verschlungenen Ketten von vier verschiedenen Nukleotiden, den Basen Adenin, Thymin, Guanin und Cytosin (siehe Abbildung 2.1). Nukleotide sind kleine Biomoleküle, bestehend aus einer Phosphorsäure, einem Zuckermolekül und einer von fünf verschiedenen Basen. Beide Nukleotidketten werden durch Paare von Wasserstoffbrücken zusammengehalten, welche sich nur zwischen den Basen Adenin und Thymin sowie zwischen Guanin und Cytosin bilden. Die Ermittlung der Abfolge und Position der Basenpaare nennt man Sequenzanalyse, welche neben der Analyse der im Folgenden beschriebenen Genexpression zu den wichtigsten Werkzeugen des Forschungsbereichs Genomics zählt.

In Lebewesen wird ein Teil der genetischen Information zur Proteinsynthese verwendet. Als (proteincodierendes) Gen bezeichnet man eine Abfolge von Nukleotiden, welche zur Bildung eines Proteins verwendet wird. Die Gesamtheit aller Gene bildet das Genom. Die Proteinsynthese beginnt mit der Transkription der DNA. Dazu wird die DNA „abgelesen“ und RNA-Stränge synthetisiert. Auf die Transkription eines Gens folgt die Translation. Hierbei wird die RNA mithilfe der Ribosomen in eine Aminosäuresequenz übersetzt. Der gesamte Prozess von Transkription, Translation, der Faltung der entstandenen Aminosäuresequenz und weitere, hier nicht betrachteten Zwischenschritte zur Bildung eines Proteins, wird als Genexpression bezeichnet.

Im Bereich Genomics ist die Analyse der DNA-Sequenz nur ein erster Schritt hin zum besseren Verständnis zur Bedeutung der genetischen Information. Das Wissen über die Aufgaben verschiedener Gene schwankt oft stark. Um die Funktionen unterschiedlicher Gene leichter vergleichen und beispielsweise die Auswirkungen der Änderung von Genexpressionen auf höherer Ebene einheitlich beschreiben zu können, wurden eine Reihe von Klassifikationshierarchien entwickelt.

Klassifikation von Gen-Funktionen

Das Gene-Ontology-Projekt [5] (GO) hat sich das Ziel gesetzt, eine Basis für die einheitliche und konsistente Beschreibung von Gen-Funktionen zu schaffen. Dazu werden beständig neue GO-Terme definiert und hierarchisch in Beziehung gesetzt. Die drei Hauptkategorien der GO sind „Molekulare Funktionen“, „Biologische Prozesse“ und „Zelluläre Komponenten“. GO-Terme können dabei gleichzeitig mehreren übergeordneten GO-Termen zugeordnet sein.

Ein weiteres Klassifikationssystem von Bedeutung ist die BRITE-Hierarchie, entwickelt als Teil der KEGG-Datenbank [6, 7]. KEGG steht für „Kyoto Encyclopedia of Genes and Genomes“ und BRITE für „Biomolecular Relations in Information

[1] Eukaryoten, aus dem Griechischen *eu* = gut und *karyon* = Kern, sind ein- oder mehrzellige Lebewesen mit Zellkern, Zellmembran und verschiedenen Zellorganellen. Eukaryoten werden in die Reiche Tiere, Pflanzen und Pilze unterteilt.

[2] Prokaryoten, aus dem Griechischen *pro* = vor und *karyon* = Kern, sind im Vergleich zu Eukaryoten einfacher aufgebaute einzellige Lebewesen ohne Zellkern. Hierzu zählen Bakterien und Archaeen (auch Urbakterien genannt).

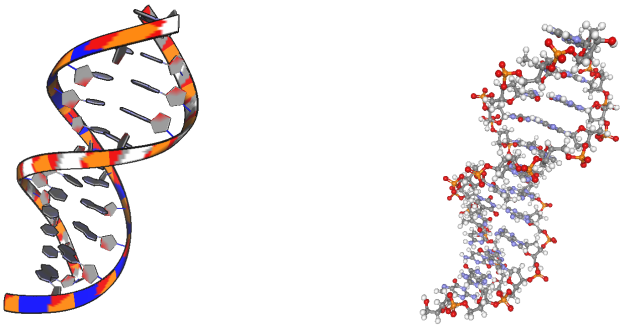

Abbildung 2.1: 3-D-Darstellung eines Ausschnitts der DNA als vereinfachte Darstellung (links) und als Stäbchenmodell (rechts). Zur Unterscheidung der verschiedenen Atome wird ein Farbcode verwendet. Gut erkennbar sind die beiden DNA-Stränge, die in Form einer Doppelhelix ineinander verschlungen sind. Die DNA-Strukturinformationen für diese Abbildung wurden der PDB-Datenbank [3] entnommen, zur Berechnung der 3-D-Darstellung wurde die Software BALLView [4] verwendet.

Transmission and Expression“. KEGG-BRITE verwendet KO-Identifikatoren (KO: KEGG Orthology) als Identifikator für die enthaltenen Datenbankeinträge. Jedem KO-Identifikator sind eine Reihe von Genen verschiedener Organismen zugeordnet. Neben Gen-Identifikatoren sind in der BRITE-Datenbank Verweise auf Stoffwechselwege sowie hierarchisch strukturierte Gen-Funktionsbeschreibungen enthalten. Von großer Nützlichkeit für die Bioinformatik ist, dass diese Datenbankinformationen nicht nur über eine Webschnittstelle, sondern auch über Programmierschnittstellen sowie als Datenbankdateien zum Download abrufbar sind.

Die Sequenzanalyse liefert ein statisches Bild der genetischen Sequenz eines Organismus. Mithilfe der eben vorgestellten Klassifizierungshierarchien kann diese Information aufbereitet und leichter verständlich gemacht werden. Von großer Bedeutung ist jedoch nicht nur das statische Bild, sondern die für jeden Organismus unterschiedliche, von der Umgebung und Zelldifferenzierung abhängige Genexpression. Erst die „Produktion“ unterschiedlicher Proteine ermöglicht es der Zelle, auf Umwelteinflüsse zu reagieren, sich anzupassen und bei höheren Lebewesen durch Ausdifferenzierung verschiedene Gewebe zu bilden. Auf Biochips basierende Analysen liefern essentielle Daten für das Verständnis dieser komplexen Vorgänge. Die Grundlagen dazu werden im Folgenden vorgestellt.

Biochip-Analyseverfahren

Zur Analyse der Genexpression werden hochparallele Messverfahren auf der Basis von Biochips verwendet. Ein Biochip besteht aus einer festen Oberfläche aus Glas, Metall oder Kunststoff und einer systematischen Anordnung von so genannten biomolekularen Sonden (engl. „probes“), beispielsweise Nukleinsäuren oder Proteinen [8]. In Abhängigkeit vom verwendeten Oberflächenmaterial, der Anzahl, Dichte

sowie der Größe der Sonden, spricht man von unterschiedlichen Technologieplattformen: Macroarrays, Microarrays, Oligonukleotid-Arrays und mikroelektrischen Arrays. Das Messverfahren für all diese Technologieplattformen basiert auf Wechselwirkungen der Bestandteile des Testpräparats mit Sonden auf dem Biochip. Es wird ein einzelnes Reaktionsgefäß verwendet, in welchem alle Wechselwirkungen unter den gleichen Bedingungen ablaufen. Auf Basis der beobachtbaren Veränderung des Biochips und der bekannten Position der Sonden können Rückschlüsse auf die Konzentration einzelner Substanzen im Testpräparat gezogen werden. Der bekannteste Hersteller von (DNA-Microarray) Biochips ist die Firma Affymetrix. Diese produziert und vertreibt eine Reihe von Biochips unter der registrierten Marke „GeneChip“.

2.1.2 Proteomics

Analysen zur Entstehung und Bedeutung der Proteine sind nach der Analyse der genetischen Sequenz der nächste Schritt zur Entschlüsselung des Lebens. Dieser Forschungsbereich wird „Proteomics“ genannt und hat die umfassende Analyse des Proteoms, also die Identifikation, Quantifizierung und Aufklärung der Bedeutung der Gesamtheit aller Proteine eines Organismus zum Ziel.

Proteine sind aus bis zu 20 verschiedenen Aminosäuren zusammengesetzte Makromoleküle. Sie gehören zu den wesentlichen Grundbausteinen einer Zelle. Die komplexen Proteinformen und Strukturen bestimmen deren spezifische Funktionen. Beispielsweise verleihen einige Proteine der Zelle ihre räumliche Struktur, andere sind die Grundlage für Stofftransporte, Signalweiterleitung oder Genregulation.

Die Proteinsynthese wird, wie im folgenden Abschnitt erläutert, vielfältig reguliert.

Genregulation, Signaltransduktionsnetzwerke

Biochemische Prozesse leiten im Rahmen der Genregulation Stimuli über Signalkaskaden weiter und ermöglichen es somit der Zelle, auf veränderte Lebensumstände zu reagieren. Im Falle mehrzelliger Lebewesen spielt sie bei der Ausdifferenzierung verschiedener Gewebe und Organe eine entscheidende Rolle. Die Signalverarbeitung im Rahmen der Genregulation basiert auf verschiedenen physikalischen oder chemischen Prinzipien. Träger der Informationen sind unter anderem Ionen, Proteine und Hormone. Den Startpunkt einer Signalkaskade bilden im Allgemeinen Rezeptorproteine an der Zellwand bzw. Zellmembran. Innerhalb einer Zelle laufen nach dem Eintreffen eines Signals eine Reihe von Protein-Protein-Interaktionen ab. Dabei wechseln Signalmoleküle zwischen mindestens zwei Zuständen und fungieren so als molekulare Schalter. Signaltransduktionsnetzwerke entstehen, wenn mehrere Signal-Kaskaden gemeinsam betrachtet werden. Man spricht von Genregulationsnetzwerken, wenn die Wechselwirkung von Genen und Proteinen im Mittelpunkt der Betrachtung steht.

Stets aktive, nicht regulierte Gene mit grundlegenden Aufgaben für den Organismus werden „Housekeeping-Gene“ genannt und sind beispielsweise die Basis grundlegender Stoffwechselvorgänge.

Protein-Protein-Interaktionsnetzwerke

Genregulation und Signaltransduktion zeigen, dass Protein-Protein-Interaktionen von großer Bedeutung für viele zelluläre Abläufe sind. Liegt der Fokus einer Untersuchung in den vielfältigen Protein-Protein-Interaktionen (PPI), sind Protein-Protein-Interaktionsnetzwerke (manchmal PIN für protein interaction network genannt) ein wichtiges Hilfsmittel. PPI-Netzwerke stehen zunehmend im Fokus der Bioinformatikforschung, da inzwischen eine Vielzahl an umfangreichen PPI-Netzwerken, beispielsweise für die Organismen *Saccharomyces cerevisiae* (Backhefe) [9], *Helicobacter pylori* (Stäbchenbakterium, das im menschlichen Magen vorkommt) [10], *Drosophila melanogaster* (Taufliege) [11] und *Caenorhabditis elegans* (Fadenwurm) [12], ermittelt werden konnten.

Als Interaktom bezeichnet man die Gesamtheit der Protein-Protein-Interaktionen innerhalb einer Zelle, welche in einem statischen PPI-Netzwerk abgebildet werden [13]. Dabei bleiben dynamische Einflussfaktoren vorerst unbeachtet. Es soll vielmehr ein Grundgerüst geschaffen werden, welches alle möglichen Protein-Protein-Interaktionen beschreibt.

Enzym-Klassifikation

Enzyme sind biochemische Reaktionen katalysierende Proteine. Seit den späten 60er Jahren werden neu entdeckte Enzyme systematisch benannt und mit einem Zahlensystem kategorisiert [14]. Beispielsweise wurde dem Enzym „Glucose-1-phosphate-phosphodismutase“ die Nummer „2.7.1.41“ zugeordnet. Die erste Zahl steht hier für die Hauptgruppe „Transferases“, die zweite für „Transfering phosphorous-containing groups“, die dritte Zahl schließlich für die konkrete Gruppe „Phosphotransferases with an alcohol group as acceptor“. Die vierte Zahl bezeichnet schließlich die dem eingruppierten Enzym zugeordnete Nummer. Der aktuelle Datenbestand ist durch die Expasy-Enzym-Datenbank öffentlich abrufbar [15].

Analyseverfahren

Ermittlung von Protein-Protein-Interaktionen Zur experimentellen Analyse der Interaktionen von Proteinen werden experimentelle Protokolle angewandt. Zu den wichtigsten Protokollen gehören das Y2H-Verfahren (Yeast two-hybrid system), das AP-MS-Verfahren (affinity purification coupled to mass spectrometry) sowie die bereits vorgestellten biochipbasierten Verfahren.

Y2H ist eine *in-vivo*-Methode (lat., „am lebenden Organismus“) zum Nachweis von Protein-Protein-Wechselwirkungen in Hefe-Zellen. AP-MS und Biochip-Analysen sind *in-vitro*-Verfahren (lat., „im Glas“), welche Protein-Protein-Interaktionen außerhalb des lebenden Organismus detektieren.

Anfangs wurden diese Verfahren zur detaillierten Analyse ausgewählter biologischer Prozesse verwendet [16, 17], inzwischen werden sie zur Analyse des Interaktoms verschiedener Organismen angewandt [18].

Analyse des Proteoms Während die genetische Sequenz weitgehend unverändert in allen Zellen vorliegt, ist die Zusammensetzung des Proteoms abhängig vom Al-

ter der Zelle, dem Gewebe und anderen Faktoren wie biotischem und abiotischem Stress [19].

Zur Analyse werden gelbasierte (1-D- / 2-D-Gelelektrophorese) und nicht gelbasierte Verfahren (Flüssigkeitschromatografie) zur Auftrennung der Probe verwendet. Die sich an die Auftrennung anschließende Identifizierung erfolgt mithilfe der Massenspektrometrie (MS) [20]. Die MS-Ergebnisse werden mithilfe von Datenbanken (zum Beispiel SwissProt [21]) mit den Massen von *in silico* (Bezeichnung für reine Computeranalysen) verdauten Proteinen verglichen und so identifiziert. Der Erfolg der Identifizierung ist somit abhängig vom Umfang der in der verwendeten Datenbank vorhandenen Daten.

2.1.3 Metabolomics

Die Biochemie eines Organismus bestimmt das Leben auf mikroskopischer und somit auch auf der makroskopischen Ebene. Auf mikroskopischer Ebene laufen in einer Zelle ständig unzählige biochemische Reaktionen ab, welche chemische Substanzen umwandeln. Das Ziel des Forschungsbereichs „Metabolomics“ ist die systematische Untersuchung des Metaboloms [22], also aller Metabolite. Metabolite sind Zwischenprodukte biochemischer Stoffwechselwege. Metabolomics erlaubt einen direkten Rückschluss auf die Physiologie einer Zelle und ist nach Genomics und Proteomics der nächste Schritt hin zum vollständigen Verständnis für die komplexen Vorgänge des Lebens.

Biochemische Reaktionen werden zu einem großen Teil von Enzymen katalysiert. Im Allgemeinen wird durch Enzyme die Aktivierungsenergie für spezifische Reaktionen herabgesetzt. Durch die Katalyse werden Reaktionen stark beschleunigt oder in einer spezifischen Umgebung überhaupt erst ermöglicht. Die Funktion bestimmter Enzyme ist von der Anwesenheit weiterer Moleküle oder Metallionen, den Kofaktoren, abhängig. Diese können beispielsweise Energie liefern, Elektronen abgeben oder aufnehmen. Kofaktoren werden so Bestandteil enzymatisch katalysierter Reaktionen und gehen als Koenzyme unverändert oder als Kosubstrate verändert aus der Reaktion hervor. Die meisten biochemischen Reaktionen sind umkehrbar, bei fehlender äußerer Einwirkung stellt sich daher nach einiger Zeit ein biochemisches Gleichgewicht ein. Da auf eine lebende Zelle ständig das Gleichgewicht störende äußere Faktoren einwirken, laufen Reaktionen in der Natur jedoch häufig in eine bestimmte, vorherrschende Richtung ab.

Thema des nächsten Abschnitts sind Reaktionsnetze und Stoffwechselwege, welche durch Zusammenfassung einer Reihe von Reaktionen und die Charakterisierung der beteiligten Substanzen entstehen.

Biochemische Reaktionsnetze und Stoffwechselwege

Die an einer biochemischen Reaktion beteiligten Substanzen werden in Ausgangs- und Endsubstanzen unterteilt. Die Ausgangssubstanzen einer enzymatisch katalysierten Reaktion werden als Substrate, die Endsubstanzen als Produkte bezeichnet. Ein Reaktionsweg ist eine Sequenz von Reaktionen, bei der mindestens eine Endsubstanz einer jeden Reaktion als Ausgangssubstanz der nachfolgenden Reaktion

beteiligt ist. Wenn eine einzelne Endsubstanz oder mehrere Endsubstanzen einer Reaktion als Ausgangssubstanz(en) anderer Reaktionen beteiligt sind, entstehen gegenseitige Abhängigkeiten, welche in Form eines Netzwerkes abgebildet werden können. Ein beliebiger Ausschnitt aus der Vereinigung aller Reaktionswege (dem Stoffwechsel) wird Reaktionsnetz genannt. Für einen Organismus besonders wichtige Reaktionsnetze oder Reaktionswege werden Stoffwechselwege oder Pathways genannt und häufig isoliert betrachtet.

Besonders umfangreiche Quellen für biochemische Stoffwechselwege sind die Datenbanken KEGG [23, 6], BRENDA (Braunschweiger Enzym Datenbank) [24, 25], MetaCyc [26], Reactome [27] und BioCarta [28].

Klassifikation von Stoffwechselwegen

Zur Katalogisierung von Stoffwechselwegen werden diese in verschiedene Klassen eingeteilt.

Beispielsweise werden KEGG-Pathways in der KEGG-Bibliothek in sechs verschiedene Hauptgruppen einsortiert: „Metabolismus“, „Genetische Informationsverarbeitung“, „Informationsverarbeitung mit der Umgebung“, „Zelluläre Prozesse“, „Menschliche Krankheiten“ und „Arzneientwicklung“. Jede dieser Hauptgruppen ist in weitere Untergruppen unterteilt. Jede Untergruppe enthält eine Reihe von Pathways.

Analyseverfahren

Eine sehr häufig im Bereich Metabolomics angewandte Analysemethode ist GC-MS, eine sehr empfindliche Technik zur Analyse von Stoffgemischen [29]. Im ersten Analyseschritt wird dabei eine Gaschromatographie (GC) und im zweiten Schritt eine Massenspektrometrie (MS) durchgeführt. Bei der Gaschromatographie wird ein Trägergas (z. B. Helium oder Wasserstoff) durch eine Säule aus beschichtetem Quarzglas geleitet. Das zu untersuchende Stoffgemisch wird in die Säule eingespritzt und erhitzt. Es erfolgt eine Auftrennung nach dem Siedepunkt der Einzelsubstanzen. Am Ende der Säule befindet sich ein Detektor, welcher den Austritt von Substanzen detektiert. Erkennungsmerkmal ist hier die Retentionszeit, die nach dem Einsetzen des Präparats vergeht, bis der Austritt von Stoffen detektiert wird. Eine der GC ähnliche Technik zur Auftrennung eines Stoffgemischs ist LC (Liquid Chromatography). LC kann im Vergleich zu GC ein anderes Metabolitspektrum erfassen.

Durch Kombination der oben genannten Trennverfahren mit einem Massenspektrometer können sehr geringe Substanzmengen nachgewiesen sowie weitergehende Rückschlüsse auf die Struktur der im Präparat enthaltenen Substanzen gezogen werden. Bei der Massenspektrometrie entstehen Profile von Masse-Ladungsverhältnissen, welche beispielsweise datenbankbasiert mit bekannten Profilen abgeglichen werden können. Auf diese Weise erfolgt die Identifizierung und Quantifizierung der Metabolite und somit die Aufschlüsselung der Zusammensetzung des Präparats.

2.2 Grundlagen der Biostatistik

Nach der Vorstellung der für die Systembiologie wichtigen Forschungsgebiete Genomics, Proteomics und Metabolomics werden in diesem Kapitel die wichtigsten Grundlagen zur statistischen Auswertung von biologischen Experimenten vermittelt.

Nach der Motivation zur Anwendung statistischer Methoden beschäftigt sich der folgende Unterabschnitt mit den Grundlagen zur Vorbereitung eines Experiments, der Identifikation der relevanten Einflussfaktoren eines Experiments und deren Definition in einem Versuchsplan. Im Folgenden thematisiert der Unterabschnitt „Statistische Hypothesentests“ relevante Grundlagen zur Hypothesenbildung, zur Signifikanz von statistischen Tests und zu dabei relevanten Wahrscheinlichkeitsverteilungen. Schließlich werden ausgewählte, häufig verwendete statistische Tests vorgestellt.

2.2.1 Statistische Grundbegriffe

Aus statistischer Sicht ist ein Experiment eine geplante und kontrollierte Einwirkung eines Untersuchenden auf Untersuchungsobjekte. Im Rahmen einer statistischen Erhebung wird der Zustand des Untersuchungsobjekts oder werden Vorgänge, die in Beziehung zum Untersuchungsobjekt stehen, kontrolliert beobachtet und erfasst [30]. Statistische Methoden müssen zur Analyse der Untersuchungsergebnisse herangezogen werden, wenn die Ergebnisse eines Experiments aufgrund der Variabilität des Beobachtungsmaterials, variabler Beobachtungsbedingungen oder Messungenauigkeiten nicht beliebig oft und exakt reproduzierbar sind. Die aufgrund der Streuung der quantitativ erfassten Merkmale entstehende Ungewissheit hinsichtlich eindeutiger Schlussfolgerungen führt zur Definition der Statistik [30]:

> „Statistik ist eine Zusammenfassung von Methoden, die uns erlauben, vernünftige optimale Entscheidungen im Falle von Ungewissheit zu treffen.“

Man unterscheidet die (beschreibende) *deskriptive Statistik* und die im Rahmen dieser Arbeit im Vordergrund stehende (analytische) *induktive Statistik* [31]. Die deskriptive Statistik beschäftigt sich mit der Untersuchung und Beschreibung der Grundgesamtheit, welche die Menge aller für eine Fragestellung potenziell zu betrachtenden Objekte repräsentiert. Die analytische Statistik dient der Vorbereitung von Entscheidungen und schließt von einem möglichst repräsentativen Teil der Grundgesamtheit, also der Stichprobe, auf das Ganze.

Als Faktoren eines Experiments werden die Einflussgrößen bezeichnet, welche bei der Durchführung des Versuchs erfasst und gezielt variiert werden können. Nicht erfasste Einflussgrößen bilden den Versuchsfehler. Störfaktoren eines Experiments sind zur Minimierung des Versuchsfehlers erfasste Faktoren, an denen kein inhaltliches Interesse besteht. In einem einfaktoriellen Versuchsplan kommt nur ein einzelner Faktor vor, ein mehrfaktorieller Versuchsplan enthält mindestens zwei Faktoren. Sind im mehrfaktoriellen Versuchsplan alle möglichen Faktorkombinationen berücksichtigt, spricht man von gekreuzten Faktoren, deren Kombination einen gleichgerichteten

verstärkenden oder entgegengerichteten abschwächenden Einfluss auf die Zielgröße haben kann.

Bei der Versuchsplanung steht das Prinzip der Vergleichbarkeit dem Prinzip der Verallgemeinerungsfähigkeit gegenüber [30]. Die Faktoren eines Experiments müssen im Versuchsplan so angeordnet und kombiniert werden, dass der Versuchsfehler minimiert und die Vergleichbarkeit und Verallgemeinerungsfähigkeit den Ansprüchen an das Experiment genügt. Das Resultat dieser Planung ist das Experimentdesign.

Verallgemeinerungsfähige Aussagen über die Gesamtheit der möglichen Faktorstufen hinweg sind dann möglich, wenn bei jeder Durchführung des Versuchs die Auswahl der Faktorstufen zufällig erfolgt. Diese Zufallszuteilung stellt die Unabhängigkeit der Ergebnisse sicher und erlaubt durch die Ausschaltung bekannter und unbekannter systematischer Fehler eine unverfälschte Schätzung der interessierenden Effekte [30]. Leichter vergleichbare, aber weniger allgemeine Aussagen können dann gemacht werden, wenn die Faktorstufen eines Experiments bei jeder Wiederholung konstant gehalten werden. Wiederholte Messungen (Replikate) verbessern die Verallgemeinerungsfähigkeit sowie die Vergleichbarkeit der Ergebnisse.

Um statistisch gesicherte Aussagen über die Ergebnisse eines Experiments machen zu können, werden in der induktiven Statistik Hypothesen aufgestellt und mit statistischen Tests überprüft.

2.2.2 Statistische Hypothesentests

Auf der Basis von verschiedenen Stichproben berechnete Kennzahlen weisen aufgrund des zufälligen Experimentfehlers eine Streuung auf. Die nach einer bestimmten Vorschrift berechnete, im statistischen Test involvierte Kennzahl, wird Prüfgröße genannt. Ein statistischer Test ist in der Lage, *Prüfgrößenunterschiede* statistisch abgesichert nachzuweisen. Dazu wird die Nullhypothese H_0 aufgestellt, welche die Annahme über die Gleichheit zweier Merkmale beschreibt. Falls H_0 verworfen werden kann, wird der Alternativhypothese H_1 zugestimmt. Aus diesem Grund besteht die statistische Sicherheit nur in der Ablehnung, nicht in der Zustimmung zu H_0.

Eine wichtige Grundlage für einen statistischen Test ist das Wissen um die wahrscheinliche Verteilung einer Prüfgröße bei einer Wiederholung des Experiments. Diese Wahrscheinlichkeit entspricht der Häufigkeitsverteilung der Prüfgröße und kann bei stetigen Messwerten mit einer stetigen Verteilungsfunktion, der Testverteilung, dargestellt werden.

Zwei in der Natur besonders häufig annähernd anzutreffende Häufigkeitsverteilungen und somit besonders wichtige Testverteilungen sind die Normalverteilung und die Lognormalverteilung.

Das Auftreten eines normalverteilten Merkmals ist dann zu erwarten, wenn die Häufigkeitsverteilung der Merkmalsausprägungen durch das *additive* Zusammenwirken vieler voneinander unabhängiger und gleichermaßen wirksamer Faktoren bestimmt ist, keine verfälschende Selektion stattgefunden hat und eine sehr große Zahl von Messungen oder Beobachtungen vorliegt [30]. Die Normalverteilung ist durch den Mittelwert der Verteilung μ und die Standardabweichung σ vollständig bestimmt. Entspricht $\mu = 0$ und $\sigma = 1$, spricht man von einer Standardnormalverteilung.

In der Natur findet man neben normalverteilten Merkmalen oftmals Merkmale, die eine bestimmte Schranke nicht unter- oder überschreiten. Häufig sind diese Verteilungen nach links durch den Wert 0 begrenzt. Ein Beispiel hierfür sind Wachstumsgrößen. Wenn durch Logarithmieren der Beobachtungswerte annähernd normalverteilte Werte entstehen, bezeichnet man die Häufigkeitsverteilungen als Lognormalverteilung. Zurückführen lässt sich die logarithmische Normalverteilung auf das *multiplikative* Zusammenwirken vieler Zufallsgrößen.

Auf Basis der einem Test zugrundeliegenden Häufigkeitsverteilung wird ein Vertrauensbereich definiert, in dem die Prüfgröße mit einer vorgegebenen Wahrscheinlichkeit enthalten ist. Aus diesem Grund sind bei der Anwendung eines statistischen Tests zwei Arten von Fehlern unvermeidbar:

1. Mit vorgegebener Irrtumswahrscheinlichkeit α tritt der Fehler erster Art auf: H_0 wird abgelehnt, obwohl die Nullhypothese in Wahrheit zutrifft.

2. Mit unbekannter Wahrscheinlichkeit β tritt der Fehler zweiter Art auf: H_0 wird nicht abgelehnt, obwohl H_1 zutrifft.

Ein Fehler der ersten Art ist vergleichbar mit einem blinden Alarm, ein Fehler zweiter Art mit einem nicht ausgelösten notwendigen Alarm [31].

Die statistische Sicherheit einer Testentscheidung entspricht $1 - \alpha$. Mit einer Absenkung von α geht ein unbestimmter Anstieg für die Wahrscheinlichkeit eines Fehlers der zweiten Art und eine Vergrößerung des Vertrauensbereichs einher. Daraus folgt nach [30]: Sichere Aussagen sind unscharf; scharfe Aussagen sind unsicher.

Es muss ein Kompromiss gefunden werden, der zum Beispiel davon abhängig gemacht werden kann, wie folgenschwer ein Fehler erster Art ist. Nach Bayer [32] wird erfahrungsgemäß bei Routineuntersuchungen $\alpha = 0,05$ und bei wesentlichen Entscheidungen höchstens $\alpha = 0,01$ gewählt.

Test auf Normalverteilung und Identifikation von Ausreißern

Ein statistischer Test beruht unter anderem auf einer Verteilungsannahme über die Prüfgröße. Existiert kein Vorwissen über die Häufigkeitsverteilung des zu untersuchenden Merkmals, kann die beobachtete Stichprobenverteilung mit vorgegebener Irrtumswahrscheinlichkeit zumindest auf Nicht-Normalität überprüft werden. Wird dabei die Normalverteilungshypothese abgelehnt, können eine Reihe von statistischen Standardverfahren nicht sinnvoll genutzt werden.

Ein Test auf Normalverteilung, welcher im Vergleich zu komplexeren Ansätzen, wie beispielsweise dem $\mathcal{X}^2$-Anpassungstest (Details in [33]), leicht konservativere Ergebnisse liefert, wurde von David et al. 1954 vorgestellt [34]. Dazu wird der Quotient aus der Spannweite R (Differenz der größten mit der kleinsten Beobachtung) und der Standardabweichung s einer Stichprobe des Umfangs n im Rahmen einer vorgegebenen Irrtumswahrscheinlichkeit α mit tabellierten Grenzen verglichen. Liegt der ermittelte Wert außerhalb dieser Grenzen, wird die Hypothese einer Normalverteilung abgelehnt.

Als Ausreißer bezeichnet man Beobachtungen, die aufgrund des zufälligen Versuchsfehlers auffällig stark von den restlichen Messwerten abweichen. Ausreißer haben oft einen großen Einfluss auf den Mittelwert einer Stichprobe und verfälschen

deren Standardabweichung. Sie müssen vor der Datenanalyse erkannt und gesondert betrachtet werden, um fehlerhafte Rückschlüsse zu vermeiden. Zur Identifikation von Ausreißern in einer normalverteilten Stichprobe ($X = \{x_i, ..., x_n\}$) ist der Grubbs-Test [35, 36] geeignet: Beginnend mit der am weitesten vom Mittelwert der Stichprobe entfernten Beobachtung x_e wird der Testwert Z berechnet: $Z = (|\overline{x} - x_e|)/s$, mit $s = \sqrt{1/(n-1) \cdot \sum_{i=1}^{n} (x_i - \overline{x})}$. Mit vorgegebener Irrtumswahrscheinlichkeit α kann für die Normalverteilung ein Vertrauensbereich definiert werden, in dem mit einer Wahrscheinlichkeit $1 - \alpha$ die Beobachtung vorzufinden ist. Liegt die Beobachtung außerhalb des Vertrauensbereichs, so wird diese als Ausreißer betrachtet und aus der Stichprobe entfernt. Die Berechnung und der Vergleich werden solange wiederholt, wie die jeweils am weitesten vom Mittelwert entfernte, noch in der Stichprobe verbliebene Beobachtung außerhalb des Vertrauensbereichs liegt. Nachteile dieses Verfahrens sind, dass sich die Wahrscheinlichkeit für das Entdecken eines Ausreißers nach dem Entfernen einzelner Beobachtungen verändert und dass das Verfahren nur bei größeren Stichprobenumfängen verwendet werden kann.

Test auf gleiche Stichprobenmittelwerte

In den meisten Fällen, bei denen von den Stichprobenmittelwerten auf den Mittelwert der Grundgesamtheit geschlossen wird, ist die Standardabweichung der Grundgesamtheit σ unbekannt. Um hier statistisch gesicherte Aussagen und somit eine Konfidenzschätzung über den Mittelwert einer Stichprobe treffen zu können, muss ein Vertrauensbereich auf der Basis einer Verteilungsfunktion definiert werden. Unter der Voraussetzung, dass die zu vergleichenden Stichproben unabhängig und annähernd normalverteilt sind, erlaubt die von W. S. Gosset 1908 unter dem Pseudonym „Student“ vorgestellte [37] Verteilungsfunktion t eine Konfidenzschätzung für den Parameter μ, ohne Wissen von σ, aber mit dem Wissen um die Standardabweichung der Stichprobe s [33]. Eine weitere Grundannahme ist, dass die beiden Grundgesamtheiten die gleiche Standardabweichung σ aufweisen. Die *Student*-Verteilung, kurz t-Verteilung, ist eine stetige Verteilungsfunktion, die durch die wahrscheinliche Abweichung eines Stichprobenmittelwerts vom Mittelwert der Grundgesamtheit im Verhältnis zum Standardfehler der Stichprobe definiert ist [30]. Mithilfe der t-Verteilung kann unter Einhaltung einer Irrtumswahrscheinlichkeit α die Nullhypothese, die darin besteht, dass zwei Grundgesamtheiten den gleichen Mittelwert aufweisen, überprüft werden. Zur Überprüfung dieser Hypothese verwendet man den doppelten[3] t-Test. Im Falle einer zweiseitigen Fragestellung wird bei diesem Test sowohl ein positiver als auch ein negativer signifikanter Unterschied der Stichprobenmittelwerte erkannt. Welchs t-Test [38] ist eine Variante des doppelten t-Tests, bei der die Standardabweichungen σ_1 und σ_2 der beiden Grundgesamtheiten ebenfalls unbekannt, jedoch nicht unbedingt gleich sein müssen. Da die nicht gegebene Annahme einer gleichen Standardabweichung eine zusätzliche Unsicherheit birgt, erfolgt in dieser Variante eine Korrektur des sich aus dem Stichprobenumfang ergebenden Freiheitsgrades nach unten.

[3] Man spricht von einem doppelten t-Test, wenn zwei Stichproben miteinander verglichen werden. Ein einfacher t-Test dient hingegen zur Überprüfung einer Mittelwerthypothese über eine einzelne Stichprobe.

Signifikanztest auf Unabhängigkeit in der Kontingenztafel

Zur Auswertung von in Kategorien unterteilten Variablen werden Kontingenztafeln verwendet. Eine Kontingenztafel ist eine Kreuztabelle, welche die absoluten Häufigkeiten bestimmter Merkmalsausprägungen enthält. Um eine Kontingenztafel auf Unabhängigkeit der betrachteten Variablen zu überprüfen, kann beispielsweise der *Exakte Fisher-Test* verwendet werden. Im Gegensatz zum χ^2-Test, kann der Exakte Fisher-Test auch bei kleinen Fallzahlen eingesetzt werden. Motulsky [39] nennt für den χ^2-Test als Mindestzahl der Untersuchungsobjekte zwanzig und fünf als kleinsten annehmbaren Erwartungswert für die Anzahl der Ausprägungen in den Kategorienkombinationen. Weiterhin erlaubt der Exakte Fisher-Test die Betrachtung einseitiger und zweiseitiger Fragestellungen, wohingegen mit dem χ^2-Test ausschließlich zweiseitige Fragestellungen untersucht werden können. Um das Problem und dessen statistische Analyse mit dem Exakten Fisher-Test möglichst kurz und prägnant beschreiben zu können, wird hier nur der Fall der Einteilung der beiden Variablen in genau zwei Kategorien betrachtet, die Kontingenztafel entspricht dann einer Vierfeldertafel. Sind die Daten in mehr als zwei Kategorien unterteilt, kann eine generalisierte Form des hier beschriebenen statistischen Tests verwendet werden [40].

Die Vierfeldertafel enthält in Spalten die Kategorien der Variable 1 und in den Zeilen die Kategorien der Variable 2. Die Anzahl des Auftretens einer bestimmten Kombination von Variablenausprägungen wird in der Tabelle vermerkt. Zur Kontrolle beziehungsweise als Hilfsmittel für die Berechnung werden in einer zusätzlichen Spalte und zusätzlichen Zeile die Zeilen- und Spaltensummen vermerkt:

		Variable 1		
		A	nicht A	$\sum$
Variable 2	B	a	b	$a+b$
	nicht B	c	d	$c+d$
	$\sum$	$a+c$	$b+d$	$n=a+b+c+d$

Fisher zeigte, dass unter Voraussetzung der stochastischen Unabhängigkeit der betrachteten Variablen die Wahrscheinlichkeit p für das Auftreten einer Häufigkeitsverteilung $T = (a, b, c, d)$ mit $n = a + b + c + d$ mit folgender Formel bestimmt werden kann [41]:

$$p(T) = \frac{(a+b)!\,(c+d)!\,(a+c)!\,(b+d)!}{n!\,a!\,b!\,c!\,d!} \tag{2.1}$$

Gesucht ist nun die Summe der vom Zufall bestimmten Wahrscheinlichkeiten für die konkrete Beobachtung und jeder anderen „extremeren“ Beobachtung unter der Voraussetzung von konstanten Spalten- und Zeilensummen. Dabei wird die einseitige und zweiseitige Fragestellung unterschieden. Dazu muss ein Kriterium gefunden werden, anhand dessen alle gültigen Vierfeldertafeln in eine Reihenfolge gebracht werden. Als Basis für den Vergleich kann beispielsweise das Verhältnis von a zu b dienen. Ob die Anordnung absteigend oder aufsteigend erfolgt, hängt von der

betrachteten Nullhypothese ab. Für die einseitige Fragestellung werden die Vierfeldertafeln entsprechend der Anordnung in einer Richtung betrachtet. Die zweiseitige Fragestellung betrachtet extremere Häufigkeitsverteilungen nicht nur in eine, sondern in zwei Richtungen. Nach [40] wird dazu bei zweiseitiger Fragestellung zuerst die Wahrscheinlichkeit $p(T)$ für die Beobachtung T berechnet. Dann werden alle weiteren möglichen anderen Vierfeldertafeln mit gleichen Spalten- und Zeilensummen wie bei der konkreten Beobachtung erzeugt und die zugehörigen p-Werte bestimmt. Alle p-Werte, die kleiner oder gleich $p(T)$ sind, werden zu $p(T)$ addiert, um die Gesamtwahrscheinlichkeit für ein zufallsbestimmtes Auftreten von T oder einer extremeren (unwahrscheinlicheren) Beobachtung zu bestimmen.

Anwendung findet dieser Test beispielsweise in klinischen Studien, welche die Wirksamkeit eines Medikaments untersuchen (Variable 1: Einnahme Medikament/ Placebo, Variable 2: Verbesserung des Zustands/keine Verbesserung). Die Nullhypothese, welche statistisch gesichert abgelehnt werden soll, könnte beispielsweise lauten „Das Medikament führt zu keiner Verbesserung des Zustands“.

2.2.3 Korrelationsanalysen

Der erste Unterabschnitt befasst sich mit der Pearson-Korrelation, welche den linearen Zusammenhang zwischen zwei Merkmalen bestimmt. Die nachfolgend vorgestellte Spearman-Korrelation ist robust gegenüber Ausreißern und dient der Identifikation möglicherweise nicht-linearer Zusammenhänge. Anschließend wird ein Ansatz zur Evaluierung der statistischen Signifikanz der berechneten Kennzahlen vorgestellt.

Die Ergebnisse von Korrelationsanalysen sollten dabei stets zurückhaltend interpretiert werden. Eine beobachtete hohe Korrelation gibt keinen direkten Hinweis auf die tatsächliche Ursache für die Beobachtung. Beispielsweise kann es Abhängigkeiten zu nicht betrachteten Parametern geben. Aus den Ergebnissen lässt sich keine Beziehung zwischen Ursache und Wirkung ableiten.

Pearson-Korrelation

Pearson und Lee stellten 1902 eine Familienstudie vor, in der unter anderem die Größen von Geschwisterpaaren untersucht wurden [42]. Das dort vorgestellte, heute „Pearson-Korrelationskoeffizient r“ genannte Maß, ist dimensionslos und beschreibt den Grad des linearen Zusammenhangs zwischen zwei Merkmalen $X = \{x_1 ... x_n\}$ und $Y = \{y_1 ... y_n\}$. Der Korrelationsfaktor r liegt stets im Bereich von -1 bis 1. Positive Werte zeigen die Tendenz, das X und Y gemeinsam ansteigen. Für negative Werte von r sind große Werte von X mit kleinen Werten von Y verbunden.

Die Pearson-Korrelation[4] für die Messwerte zweier Messreihen lässt sich wie folgt ermitteln:

$$r = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2} \cdot \sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}} \tag{2.2}$$

[4]Wie bei anderen statistischen Kennzahlen werden zur Beschreibung der Grundgesamtheit griechische Buchstaben und für Stichprobenkennzahlen lateinische Buchstaben verwendet.

Eine wichtige Eigenart der Pearson-Korrelation ist deren Anfälligkeit für Ausreißer. Wenn statt des linearen Zusammenhangs der verteilungsunabhängige Zusammenhang zweier Merkmale von Interesse ist, so kann die im Folgenden beschriebene Spearman-Korrelation verwendet werden.

Spearman-Korrelation

Der von Spearman 1904 vorgestellte „Rangkorrelationskoeffizient r_s“ ist im Gegensatz zur Pearson-Korrelation ein parameterfreies Maß, welches den Zusammenhang zwischen zwei Merkmalen X und Y beschreibt [43]. Parameterfreie statistische Methoden können durchaus eine Parametrisierung erfordern. Sie machen keine Annahmen über die Wahrscheinlichkeitsverteilung der untersuchten Prüfgröße und sind daher auch anwendbar, wenn die bei anderen statistischen Aussagen notwendigen Verteilungsvoraussetzungen nicht erfüllt sind.

Die Berechnung von r_s erfolgt analog zu 2.2, wobei die beobachteten Messwerte durch deren entsprechende Rangzahlen ersetzt werden.

Test auf Signifikanz einer beobachteten Korrelation

Mit sinkendem Stichprobenumfang n steigt die Wahrscheinlichkeit für eine zufällig beobachtete Korrelation. Ob eine beobachtete Korrelation mit einer gewissen Wahrscheinlichkeit nicht zufällig aufgetreten ist, kann wie folgt ermittelt werden [30, 31]:

Die Nullhypothese (H_0) lautet H_0 ($\rho = 0$), wobei ρ der unbekannte Korrelationskoeffizient der Grundgesamtheit ist. H_0 kann erst dann verworfen werden, wenn die beobachtete Korrelation (r) größer als ein unter einer gewissen Irrtumswahrscheinlichkeit zufällig möglicher Wert ist. Mit einer 1926 von Fisher nachgewiesenen Beziehung [44] kann die Signifikanz einer beobachteten Korrelation überprüft werden, wobei diese für alle $\hat{t} \geq t_{n-2;\alpha}$ abgelehnt wird:

$$\hat{t} = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}} \tag{2.3}$$

$t_{n-2;\alpha}$ steht für einen Anteil $\alpha = [0..1]$ unter der Fläche der mit $n-2$ Freiheitsgraden parametrisierten t-Verteilungskurve ($\alpha = 1$ entspricht der gesamten Fläche). Die entsprechenden Funktionswerte können in Statistikbüchern nachgeschlagen oder von Computern mit vorgegebener Genauigkeit berechnet werden.

Wird eine Vielzahl an Korrelationen berechnet und auf Signifikanz getestet, erhöht sich bei einer Gesamtbetrachtung die Wahrscheinlichkeit für das Auftreten mindestens eines Fehlers erster Art. Soll diese Fehlerwahrscheinlichkeit auf $P = \alpha$ reduziert werden, kann der α-Wert wie von Motulsky vorgeschlagen entsprechend der Formel $\alpha' = 1 - \sqrt[n]{1-P}$ angepasst werden [39]. Durch diesen und ähnliche Ansätze [45, 46, 47] wird bei globaler Betrachtung die Wahrscheinlichkeit für Fehler der ersten Art effektiv reduziert. Problematisch ist, dass dadurch die Wahrscheinlichkeit von Fehlern der zweiten Art steigt und somit viele Unterschiede in den Datensätzen nicht mehr erkannt werden.

2.3 Grundlagen der Informatik

Im Anschluss an die Vorstellung grundlegender Definitionen der Graphentheorie werden in diesem Kapitel Graphoperationen, Datenstrukturen zur Speicherung von Graphen und Graphalgorithmen vorgestellt. Darauf folgt die Darstellung der Themen Informationsvisualisierung und Graphlayout sowie die Vorstellung grundlegender Methoden zur netzwerkintegrierten Datenvisualisierung.

2.3.1 Grundbegriffe der Graphentheorie

Im Titel dieser Arbeit und in den vorangegangenen Abschnitten wurde häufig der Begriff „Netzwerk“ verwendet. Dieser Begriff dient der Beschreibung von Strukturen, welche untereinander in Beziehung stehende gegenständliche oder gedachte Objekte beinhalten. Im folgenden Abschnitt wird die Definition des Begriffs Graph vorgestellt, der die formale mathematische Entsprechung des Begriffs Netzwerk darstellt. Die in diesem Kontext relevante Terminologie der Graphentheorie kann hier nur verkürzt dargestellt werden, ausführliche Betrachtungen finden sich beispielsweise in [48].

Ein *ungerichteter Graph* $G = (V, E)$ besteht aus einer Knotenmenge V und einer Kantenmenge E, wobei jeder Kante $e \in E$ von G zwei nicht notwendigerweise verschiedene Knoten aus V zugeordnet sind. Eine Kante wird in der Form $e = \{u, v\}$ mit u und v als Endknoten der Kante beschrieben. Im Folgenden werden die Mengen V und E stets als endlich betrachtet, die betrachteten Graphen sind daher ebenfalls endlich.

Grundlegende Eigenschaften ungerichteter Graphen: Verbindet eine Kante ein- und denselben Knoten, ist also $e = \{v, v\}$, so ist e eine Schlinge. Zwei die gleichen Endknoten verbindenden Kanten e und f mit $e = \{u, v\}$ und $f = \{u, v\}$ werden parallele Kanten genannt. Ein schlichter Graph besitzt keine Schlingen und keine parallelen Kanten.

In den vorangegangenen Ausführungen wurde keine Richtung für die Verbindung zwischen Knoten definiert. In vielen Fällen ist dies jedoch von Bedeutung, um beispielsweise die zeitliche Abfolge von Zustandsübergängen mithilfe eines Graphen modellieren zu können.

Ein *gerichteter Graph* $G = (V, E)$ besteht aus einer Knotenmenge V und einer Kantenmenge E, sodass jeder gerichteten Kante e eindeutig ein geordnetes Paar (u, v) zugeordnet ist: $e = (u, v)$. Der Knoten $u \in V$ heißt Anfangsknoten, der Knoten $v \in V$ Endknoten von e. Die Definition einer Schlinge und einer parallelen Kante gilt analog auch für gerichtete Kanten. Jedoch sprechen wir hier, statt von der Knotenmenge $\{u, v\}$, von dem geordneten Paar (u, v), welches eine Kante definiert. Weiterhin gibt es nun eine weitere Form einer möglichen Kantenbeziehung, das antiparallele Kantenpaar.

Grundlegende Eigenschaften gerichteter Graphen: Zwei Kanten $e = (u, v)$ und $f = (v, u)$ heißen antiparallel, wenn jeweils der Anfangsknoten der einen Kante der Endknoten der anderen ist. Auf Basis der nun definierten Reihenfolge der eine Kante beschreibenden Knotenmenge $e = (u, v)$ wird eine Vorgänger-/Nachfolgerbeziehung abgeleitet: Der Knoten u heißt Vorgänger von v, der Knoten v ist Nachfolger von u.

Die Menge der Vorgängerknoten wird mit $Vor(v)$ bezeichnet: $Vor(v) = \{u|(u,v) \in E\}$. Die Nachfolger sind $Nach(v) = \{w|(u,w) \in E\}$. Die Vereinigung der Mengen $Vor(v)$ und $Nach(v)$ bildet die Nachbarschaft von v: $\Gamma(v) = Vor(v) \bigcup Nach(v)$. $Ein(v) = \{(u,v)|(u,v) \in E\}$ bezeichnet die Menge der eingehenden Kanten des Knotens v und $Aus(v) = \{(v,w)|(v,w) \in E\}$ die Menge der ausgehenden Kanten des Knotens v. Auf dieser Basis lässt sich der Eingangsgrad eines Knotens v mit der Anzahl der eingehenden Kanten definieren: $g^-(v) = |Ein(v)|$. Der Ausgangsgrad entspricht analog $g^+(v) = |Aus(v)|$. Als *Grad eines Knotens* bezeichnet man $g(v) = |Ein(v) \bigcup Aus(v)|$. Hat ein Knoten v weder ein- noch ausgehende Kanten ($g(v) = 0$), so bezeichnet man v als isolierten Knoten. Ein Knoten v wird als Quelle in einem gerichteten Graphen G bezeichnet, wenn $g^-(v) = 0$ und $g^+(v) > 0$. Entsprechend spricht man von einer Senke v, wenn gilt $g^+(v) = 0$ und $g^-(v) > 0$.

Pfad, Weg, Zyklus und Kreis: Eine Folge von Knoten $w = (v_1, v_2, ..., v_k)$ eines (gerichteten) Graphen $G = (V, E)$ wird als *Pfad* bezeichnet, falls gilt: $\forall i \in 1, ..., k-1 : (v_i, v_{i+1}) \in E$. Kommt in einem Pfad w jeder Knoten nur einmal vor, so spricht man von einem Weg. Verbindet eine Kante $(v_k, v_1) \in E$ in einem Pfad den Endknoten v_k mit dem Anfangsknoten v_1, so handelt es sich bei w um einen Zyklus. Man spricht bei einem Zyklus von einem Kreis, wenn in der Knotenfolge bis auf die identischen Anfangs- und Endknoten jeder Knoten nur einmal vorkommt.

Abstand zweier Knoten, Erreichbarkeit: Unter dem Abstand $d(u,v)$ zweier Knoten u und v versteht man die minimale Anzahl an Graphkanten, welche die Knoten eines Weges von u nach v verbinden. Dabei gilt $d(v,v) = 0$ $(\forall v \in V)$. Existiert ein Weg von u nach v, dann ist v von u aus erreichbar, ansonsten gilt $d(u,v) = \infty$.

(induzierter) Teilgraph: Ein (gerichteter) Graph $G' = (V', E')$ ist ein Teilgraph von G, falls gilt: $V' \subseteq V$ und $E' \subseteq E \bigcap (V' \times V')$. Sind alle Kanten aus G in G' enthalten, gilt also $E' = E \bigcap (V' \times V')$, so nennt man G' einen induzierten Teilgraph von G. Ein induzierter Teilgraph $K{=}(V_K, E_K)$ von G heißt starke Zusammenhangskomponente von G, falls K stark zusammenhängend ist und kein stark zusammenhängender induzierter Teilgraph von G existiert, der K echt enthält.

Ein zyklenfreier gerichteter Graph wird *gerichteter azyklischer Graph, kurz DAG*[5] genannt. Ein (gerichteter) Graph $G = (V, E)$ ist ein *Baum*, falls es einen Knoten $w \in V$ gibt, von dem aus alle Knoten des Graphen erreichbar sind und $|E| = |V| - 1$ gilt. Als Wurzel eines Baums G bezeichnet man den Quellknoten in G. Die Senken eines Baums werden Blätter genannt. Die Knoten eines Baums, welche nicht zu den Blättern zählen, werden innere Knoten genannt.

Bisher wurde ein Graph durch eine Menge von „anonymen“ Knoten und Kanten definiert. Praktische Relevanz erhält eine Graphstruktur oft erst dann, wenn die Knoten oder Kanten eines Graphen benannt werden. Ein Graph, der jedem Knoten und jeder Kante einen Bezeichner zuordnet, wird *gelabelter Graph* genannt.

Operationen mit Graphen

Ein statischer Graph ist für die Modellierung vielfältiger Beziehungen nützlich. Nicht selten erfordert jedoch die Berechnung von Kenngrößen oder die Modellierung sich

[5] Aus dem Englischen „**d**irected **a**cyclic **g**raph“.

verändernder Sachverhalte strukturelle Umformungen eines Graphen. Eine Reihe von grundlegenden, häufig benötigten Operationen werden hier vorgestellt:

Das Entfernen einer Kante $e \in E$ aus einem (gerichteten) Graph $G = (V, E)$ erzeugt den Graph $G - e = (V, E \setminus \{e\})$. Das Entfernen einer Menge von Kanten $M \subseteq E$ erzeugt den Graphen $G - M = (V, E \setminus M)$.

Mit dem Entfernen eines Knotens $v \in V$ werden auch alle ein- und ausgehenden Kanten vom Graph G entfernt: $G - v = (V \setminus \{v\}, E \setminus Ein(v) \bigcup Aus(v))$.

Entsprechend wird das Entfernen einer Menge von Knoten $N \subseteq V$ aus G wie folgt definiert: $G - N = (V \setminus N, E \setminus \forall(v \in N)\{Ein(v) \bigcup Aus(v)\})$.

Die *Fusion* oder *Verschmelzung* von zwei Knoten u und v sei wie folgt definiert:

$$G_{uv} = (V \setminus u, \forall(e \in E) f(e)) \left| \begin{array}{l} e = (a, b) \\ a, b \in V \end{array} \right.$$

mit

$$f(e) = \begin{cases} (a, b) & \text{falls } a \neq u \text{ und } b \neq u \\ (v, b) & \text{falls } a = u \text{ und } b \neq u \\ (a, v) & \text{falls } b = u \text{ und } a \neq u \\ (v, v) & \text{falls } a = b = u \end{cases}$$

Unter der *Kontraktion* einer Kante $e = (u, v)$ ist das Entfernen von e aus G sowie die anschließende Fusion von u und v zu verstehen: $G/e = (V', E' \setminus e)$ mit $(V', E') = G_{uv}$.

Die *Vereinigung zweier Graphen* $G = (V, E)$ und $H = (W, F)$ ist definiert als: $G \bigcup H = (V \bigcup W, E \bigcup F)$.

Datenstrukturen für Graphen

Um Graphen effizient im Computer verarbeiten zu können, müssen diese auf geeignete Weise gespeichert werden. In Abhängigkeit von der Struktur, dem Umfang (Anzahl Knoten und Kanten) und dem geplanten Einsatzzweck (unter anderem die durchzuführenden Graph-Operationen) sind verschiedene Repräsentationen zur Speicherung von Graphen geeignet. Zwei gängige und häufig verwendete Repräsentationen werden hier vorgestellt. Weitere Möglichkeiten, beispielsweise die Verwendung von sortierten Kantenlisten, sind in [48] beschrieben.

Adjazenzmatrizen Ein gerichteter Graph $G = (V, E)$ wird in einer $|V| \times |V|$ - Matrix $A_G = (a_{ij})$ mit $1 \leq i \leq |V|$ und $1 \leq j \leq |V|$ gespeichert. Dabei entspricht $a_{ij} = |(v_i, v_j) \in E|$. Für schlichte Graphen gilt somit $a_{ij} = 0$ falls $(v_i, v_j) \notin E$ und $a_{ij} = 1$ falls $(v_i, v_j) \in E$. Adjazenzmatrizen können beispielsweise als zweidimensionale Arraystruktur implementiert werden (siehe Abbildung 2.2).

Ein Vorteil dieser Repräsentation ist, dass bestimmte Analysen direkt mithilfe von Matrixoperationen durchgeführt werden können. Enthält ein Graph vergleichsweise wenige Kanten und gleichzeitig viele Knoten, erweist sich der mit der Knotenzahl quadratisch ansteigende Speicherbedarf dieser Darstellung als problematisch.

Adjazenzlisten Eine weitere Möglichkeit zur Speicherung von (gerichteten) Graphen sind Adjazenzlisten. Für den gerichteten Fall wird dazu ein Feld A der Länge

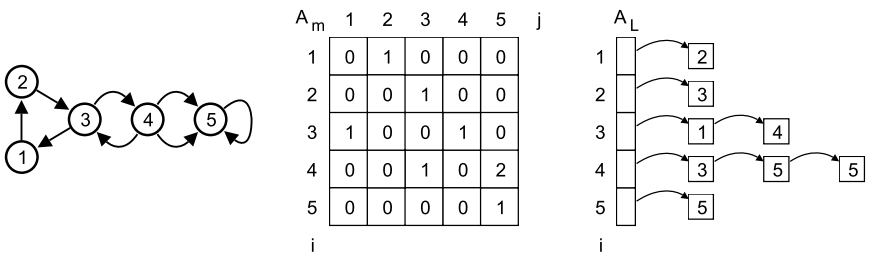

Abbildung 2.2: Ein gerichteter Graph, seine Adjazenzmatrix A_m und seine Adjazenzliste A_L.

$|V|$ verwendet, wobei jedem Eintrag A_i, $1 \leq i \leq |V|$ eine Liste der Nachfolger eines Knotens v_i zugeordnet wird (Nachbarschaftsliste): $A_i = Nach(v_i)$, $v_i \in V$.

Als Implementation kann für A ein eindimensionales Array verwendet werden, wobei jedem Arrayelement eine einfach verkettete Liste der Nachfolgerknoten zugeordnet wird (siehe Abbildung 2.2).

Ein Vorteil dieser Repräsentation ist der im Vergleich zur Matrixdarstellung verringerte Speicherbedarf. Der Speicherbedarf der Adjazenzliste wächst linear mit der Anzahl der Knoten und Kanten während der Speicherbedarf der Adjazenzmatrix quadratisch mit der Anzahl der Knoten ansteigt.

Die Liste der Nachfolger eines Knotens liegt in dieser Darstellung direkt vor. Die Ermittlung der Vorgängerknoten ist jedoch aufwendiger. Hierfür müssen alle Nachbarschaftslisten A_i durchgegangen werden. Dies lässt sich vermeiden, wenn neben der Adjazenzliste eine zweite invertierte Adjazenzliste geführt wird. Dazu wird ein Feld AI der Länge $|V|$ definiert, jedem Eintrag AI_i, $1 \leq i \leq |V|$ wird die Liste der Vorgängerknoten zugeordnet: $AI_i = Vor(v_i)$, $v_i \in V$.

Graphalgorithmen

Zur Beantwortung anwendungsbezogener Fragestellungen werden Graphalgorithmen verwendet. Eine Vielzahl von Algorithmen ist heutzutage bereits Teil von kommerziell oder frei verfügbaren Programmbibliotheken. Beispiele hierfür sind yFiles [49], Jung [50] und Gravisto [51]. In diesem Rahmen wird daher auf eine Beschreibung entsprechend grundlegender Algorithmen, wie sie beispielsweise zur Ermittlung der Nachfolger eines Graphknotens oder zum Finden der kürzesten Wege zwischen zwei Knoten oder zum Löschen von Knoten und Kanten benötigt werden, verzichtet. Es existieren jedoch häufig verschiedene Algorithmen und Datenstrukturen zur Lösung eines Problems. Diese müssen miteinander verglichen und bestimmten Kategorien zugeordnet werden können. Die hierfür notwendigen Vergleichbarkeitskriterien werden im Folgenden vorgestellt.

Vergleichbarkeitskriterien Neben Kriterien zur Wartbarkeit und zum Implementationsaufwand eines Algorithmus stehen im Allgemeinen Zeit- und Speicherkomplexität im Vordergrund der Betrachtung. Im Idealfall kann zur Ermittlung der Zeit-

Darstellungsart	Speicherplatzbedarf	Feststellung ob es eine Kante von i nach j gibt.	Bestimmung der Nachbarn (Nachfolger in gerichteten Graphen) von i	Bestimmung der Vorgänger von i (gerichtete Graphen)
Adjazenzmatrix	$O(n^2)$	$O(1)$	$O(n)$	$O(n)$
Adjazenzliste	$O(n+m)$	$O(g(i))$ bzw. $O(g^+(i))$	$O(g(i))$ bzw. $O(g^+(i))$	$O(n+m)$
Adjazenzliste mit invertierter Adjazenzliste (gerichtete Graphen)	$O(n+m)$	$O(g^+(i))$	$O(g^+(i))$	$O(g^-(i))$

Tabelle 2.1: Übersicht über die Komplexität von Graphdarstellungsarten. Verkürzt aus [48]. Betrachtet werden ungerichtete und gerichtete Graphen mit n Knoten und m Kanten.

komplexität eine Funktion $f(g)$ definiert werden, die für jede beliebige Eingabe g den Zeitaufwand eines Algorithmus zur Ermittlung des Resultats zurückliefert bzw. zur Ermittlung der Speicherkomplexität den benötigten Speicherbedarf zurückgibt. Praktisch ist dies jedoch nur in den seltensten Fällen möglich, sodass man sich darauf beschränkt, obere Schranken für häufig zu findende Eingaben („average case“ Komplexität [48]) zu suchen. Interessant sind weiterhin vom Inhalt häufiger Eingaben unabhängige Aussagen zur maximalen Komplexität („worst case“). Da sowohl die Zeit- als auch die Speicherkomplexität stark von der verwendeten Hardware abhängen, werden Komplexitätsaussagen stets abstrakt und relativ zueinander formuliert. Die Beschreibung der Eingabe eines Graphen wird dann beispielsweise auf die Anzahl der Knoten n oder die Anzahl der Kanten m abstrahiert.

Die Laufzeit eines Algorithmus oder der Speicherbedarf wird als Funktion $f : \mathbb{N} \rightarrow \mathbb{R}^+$ definiert. Mithilfe der O-Notation kann eine solche Funktion einer Klasse $O(g)$ zugeordnet werden. Dabei ist g ebenfalls eine Funktion von $\mathbb{N} \rightarrow \mathbb{R}^+$, wenn es eine Konstante $c > 0$ und $n_0 \in \mathbb{N}$ gibt, für die Folgendes gilt: $\forall n \geq n_0 : f(n) \leq c * g(n)$. Wurde beispielsweise für einen bestimmten Algorithmus eine relative Zeitkomplexität von $f(n) = 70+20*n$ ermittelt, benötigt der Algorithmus für eine Eingabelänge von $n = 10$ insgesamt 270 elementare Rechenschritte. Man spricht vereinfachend davon, dass der Algorithmus eine Zeitkomplexität von $O(n)$ hat, da beispielsweise mit $c = 55$ und $n \geq n_0 = 2$ stets gilt: $70 + 20 * n \leq 55 * n$.

Weitere häufig zu findende Komplexitätsklassen sind $O(n \log n)$, $O(n^2)$, $O(n^3)$. Algorithmen mit exponentieller Laufzeit ($O(a^n)$) mit $n > 1$) stoßen mit zunehmenden n sehr schnell an praktische Grenzen und sind somit nur in Ausnahmefällen oder für begrenzte Eingabegrößen einsetzbar.

Die O-Notation soll abschließend und beispielhaft zum Vergleich einer Adjazenzmatrix- mit einer Adjazenzlistenrepräsentation verwendet werden (vergleiche Tabelle 2.1).

2.3.2 Informationsvisualisierung und Graphlayout

Der Begriff Visualisierung ist abgeleitet aus dem Lateinischen (*videre*, sehen) und steht im Allgemeinen für eine Veranschaulichung und Darstellung abstrakter oder gegenständlicher Sachverhalte. Im wissenschaftlichen Bereich sind insbesondere die

medizinische Visualisierung, die wissenschaftliche Visualisierung und die Informationsvisualisierung etablierte Forschungsgebiete.

Die medizinische Visualisierung beschäftigt sich mit Methoden zur Darstellung von Lebewesen zum Zweck der medizinischen Diagnose. Die insbesondere im ingenieur- und naturwissenschaftlichen Bereich verbreitete wissenschaftliche Visualisierung (engl. „scientific visualization“) beschäftigt sich mit Methoden, welche physikalischen Prozessen zuordenbare gemessene oder simulierte Daten möglichst anschaulich darstellen. Der Bereich Informationsvisualisierung (engl. „information visualization“) beschäftigt sich mit Methoden zur Darstellung von abstrakten Daten, welche nicht unmittelbar physikalischen Prozessen zugeordnet werden können. Die im Titel dieser Arbeit erwähnte „Visualisierung von experimentellen Daten“ bezieht sich somit auf den Bereich Informationsvisualisierung, auf den im Folgenden näher eingegangen werden soll. Dabei soll sowohl der Prozess der Erzeugung einer grafischen Darstellung aus Daten als auch die entstandene Darstellung selbst Visualisierung genannt werden. Eine Einführung in diese Thematik findet sich in [52].

Ein bekanntes Sprichwort ist „Ein Bild sagt mehr als tausend Worte“. Eine gute Visualisierung hilft dem Betrachter oder dem Anwender einer interaktiven Visualisierung, komplexe Daten und Beziehungen zwischen den Daten leichter zu erfassen und zu verstehen. Zahlenmaterial wird je nach Anwendungsgebiet auf vielfältige Art und Weise visualisiert. Beispielsweise können Daten mit regionalem Bezug auf einer Landkarte durch eine unterschiedliche Färbung der Regionen in der Darstellung visualisiert werden, zur Darstellung von Zeitverläufen können Animationen oder Liniendiagramme verwendet werden. Auch Graphen können gezeichnet und somit visualisiert werden. Einen guten Überblick über dieses aktive Forschungsgebiet geben die Werke [53, 54, 55].

Drei Aspekte sind für eine interaktive Visualisierung von wesentlicher Bedeutung:

1. Die Auswahl oder Modellierung einer geeigneten Datenstruktur
2. Das Finden einer geeigneten grafischen Repräsentation der Daten
3. Die Auswahl oder das Design einer geeigneten Interaktionsform

In Kapitel 3 werden diese drei Aspekte berücksichtigt, um ein Gesamtkonzept zur netzwerkintegrierten Visualisierung von Messdaten zu erarbeiten. Im Folgenden sollen Grundlagen für diese Ausführungen vorgestellt werden.

Statische und dynamische Visualisierung

Eine Visualisierung lässt sich stets auch manuell erzeugen. Dazu nutzt ein Experte sein Hintergrundwissen, um auf der Basis eines konkreten Datensatzes eine anschauliche Visualisierung dieser Daten zu erstellen. Darstellungen in Büchern und auf Postern werden beispielsweise häufig auf diese Weise erzeugt.

Der große Nachteil einer solchen Vorgehensweise ist jedoch der damit verbundene Arbeitsaufwand. Wenn Zusammenhänge in den Daten noch unbekannt sind und Visualisierungen dazu genutzt werden sollen, Erkenntnisse über die Daten zu erlangen, dann ist es unpraktikabel, manuell eine Vielzahl an Visualisierungen zu erstellen,

welche jeweils unterschiedliche Aspekte hervorheben. Das Finden von *a priori* unbekannten Zusammenhängen in den Daten wird somit nur durch eine interaktive Visualisierung effektiv unterstützt.

Interaktionstechniken

Die Auswahl oder das Design einer geeigneten Interaktionsform ist ein weiterer wichtiger Bestandteil einer interaktiven Visualisierung. Da eine bestimmte Darstellungsform für einen Anwendungskontext gut, für einen anderen Kontext jedoch weniger gut geeignet sein kann, sollte ein Visualisierungssystem es dem Nutzer erlauben, die Darstellung im Idealfall interaktiv an die spezifischen Erfordernisse anzupassen.

Der zeitliche Aspekt einer Nutzerinteraktion sollte dabei nicht unbeachtet bleiben. Nach [56] lassen sich hier drei verschiedene zeitliche Ebenen unterteilen: Ereignisse, die im Abstand von bis zu 0,1 Sekunden wahrgenommen werden, können vom Anwender zeitlich nicht unterschieden werden. Für Animationen sollte die Darstellung daher mindestens zehnmal pro Sekunde aktualisiert werden. Eine Reaktion eines Visualisierungssystems wird dann vom Anwender als „unmittelbar“ erkannt, wenn die Verarbeitung des Befehls in weniger als einer Sekunde erfolgt. Die dritte zeitliche Ebene umfasst einen Zeitraum von 5-30 Sekunden (ca. 10 Sek.), ein Zeitraum, indem eine einzelne kognitive Aufgabe (jedoch keine Problemlösung) typischerweise durch den Nutzer bearbeitet wird. Beim Design und bei der Implementation eines interaktiven Visualisierungssystems sollten diese Ebenen Beachtung finden: Wird ein Befehl in weniger als 0,1 Sekunden bearbeitet, kann es sinnvoll sein, den Prozess der Objekttransformation zu verlangsamen und zu animieren [56]. Dauert die Bearbeitung eines Befehls länger als eine Sekunde, so wird die verspätete Reaktion des Programms als störend empfunden, eine Fortschrittsmeldung ist dann hilfreich.

Die wichtigsten Aspekte zur interaktiven Datenvisualisierung wurden von Shneiderman [57] als *Mantra der Informationsvisualisierung* formuliert: „Overview first, Zoom and Filter, then Detail-on-Demand“. Damit startet eine interaktive Visualisierung gewöhnlich mit einer Übersichtsdarstellung (Overview), welche allgemeine Eigenschaften der Daten geeignet repräsentiert. Mithilfe einer Zoom-Funktion können Ausschnitte der Darstellung vergrößert/verkleinert dargestellt werden, weniger relevante Bereiche werden ausgeblendet oder vereinfacht dargestellt. Entsprechend dem aktuellen Anwendungskontext können Filter dazu genutzt werden, nicht relevante Informationen auszublenden. Außerdem sollte ein interaktives Visualisierungssystem Details der Daten entsprechend den Anforderungen des Nutzers darstellen (Detail-on-Demand). Im Folgenden soll auf diese und weitere grundlegende Interaktionstechniken näher eingegangen werden. Prinzipiell können Nutzerinteraktionen drei verschiedenen Ebenen (vgl. Abbildung 2.3) zugeordnet werden [56]: Datentransformation (Data Transformation), visuelle Zuordnung von Daten zu Attributen der Visualisierung (Visual Mappings) und die Darstellungsebene (View Transformation).

Nutzerinteraktion auf Ebene der Datentransformation Auf dieser Ebene hat der Nutzer einer interaktiven Visualisierung die Möglichkeit, die für einen bestimmten

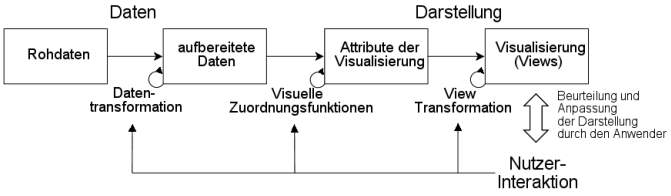

Abbildung 2.3: Beispiel für eine Pipeline zur Anpassung interaktiver Visualisierungen auf verschiedenen Ebenen (nach [56]).

Anwendungsfall relevanten Variablen auszuwählen. Die folgenden sechs Interaktions-Techniken sind dabei von besonderer Bedeutung [56]:

Eine Technik zur interaktiven Datenselektion sind *Dynamic Queries* (dynamische Abfragen). Diese ermöglichen es, Selektionsbedingungen mithilfe einer grafischen Benutzeroberfläche zu spezifizieren. Typische Benutzeroberflächenelemente sind Selektionsschaltflächen und Schieberegler. Alle Daten, die den gewählten Kriterien genügen, werden für die eigentliche Visualisierung berücksichtigt. Bei einem *Direct Walk* wechselt der Nutzer mithilfe von Verweisen (z. B. Hyperlinks) von Ansicht zu Ansicht um einen Ausschnitt der Daten zu visualisieren, zu analysieren oder zu modifizieren [56, 58]. Ein *Attribute Walk* (Attributnavigation) bezeichnet die Navigation von einem aktuell betrachteten Objekt hin zu Objekten mit einer ähnlichen oder gleichen Attributausprägung. Unter *Detail-on-demand* versteht man die Hinzunahme von zu visualisierenden Variablen auf der Basis von aktuell dargestellten, abstrakten Daten. Dies setzt im Allgemeinen einen größeren Platz zur Visualisierung der Daten voraus. Als *Brushing* bezeichnet man die gleichzeitige Visualisierung eines Objekts unter unterschiedlichen Gesichtspunkten [59, 60]. Dabei wird die Selektion eines Objekts in einer Ansicht in allen anderen Ansichten nachvollzogen. Eine weitere Technik zur Nutzerinteraktion auf Ebene der Datentransformation ist die *direct manipulation* zur interaktiven Anpassung von Objekteigenschaften.

Nutzerinteraktion zur Anpassung der Daten-Attribut-Zuordnung Auf dieser Ebene erfolgt die Zuordnung von Daten zu Attributen der Visualisierung. Beispielsweise erlaubt die *Dataflow*-Technik die Spezifikation des Datenflusses ausgehend vom relevanten Datenbestand hin zu verschiedenen Attributen der Visualisierung unter Verwendung eines interaktiven Diagramms [56].

Eine anwendungsbezogene Technik, die insbesondere in Tabellenkalkulationsprogrammen genutzt wird, sind *Pivot tables* [61]. Pivot-Tabellen erlauben dem Nutzer die schnelle und einfache Zuordnung von Daten zu Spalten und Zeilen einer Tabelle.

Nutzerinteraktion auf Darstellungsebene (View Transformation) Auch auf der Ebene der eigentlichen Darstellung können vielfältige Nutzerinteraktionen vorgesehen werden. Eine besonders wichtige und häufig eingesetzte Technik ist *Direct selection* (direkte Auswahl): Dem Anwender wird es ermöglicht, einzelne Objekte oder eine Gruppe von Objekten auszuwählen, die Selektion wird hervorgehoben dargestellt. Nachfolgende Nutzerkommandos beziehen sich auf die getroffene Auswahl.

Weitere Techniken dienen der Anpassung des Sichtbereichs durch Verschieben des sichtbaren Ausschnitts (*Camera movement*) und durch Verzerrung der Sicht, analog zur Wirkung einer Lupe (*Magic lenses*). *Overview and Detail* bezeichnet eine Interaktionstechnik, bei der ein Objekt in zwei oder mehr unterschiedlichen miteinander verknüpften Sichten dargestellt wird [62]. Eine Darstellung bietet beispielsweise einen Überblick über den Gesamtbestand der relevanten Daten, eine weitere Sicht stellt nur einen kleinen Teil der Daten detaillierter dar. *Zooming* ist eine wichtige Teilkomponente dieser Interaktionstechnik, die auch einzeln betrachtet von großem Nutzen ist.

Ästhetische Kriterien

Anschauliche Graphvisualisierungen erfordern die Einhaltung von Mindestanforderungen, welche an eine bestimmte Darstellungsart gestellt werden. Dies kann die Platzierung, also die Positionierung von Knoten in einer Gitternetzdarstellung betreffen oder beispielsweise die Kantenstützpunktpositionierung für eine orthogonale Darstellung eines elektronischen Schaltplans. Weitere Mindestanforderungen ergeben sich aus dem Anwendungskontext.

Während bestimmte Mindestanforderungen eingehalten werden *müssen*, sind im Allgemeinen weitere ästhetische Kriterien bekannt, welche eingehalten werden *sollten*. Beispiele hierfür sind die Minimierung von Kantenkreuzungen, Kantenstützpunkten und der Kantenlängen [53]. Weiterhin sollten, falls möglich, Symmetrieaspekte eines Graphen betont werden. Als problematisch erweist sich, dass sich Ästhetikkriterien widersprechen können und somit verschiedene Visualisierungen eines Graphen als schön und zweckmäßig eingeschätzt werden. Der Anwender sollte daher sinnvollerweise die Möglichkeit haben, verschiedene Visualisierungen zu erzeugen.

Standardverfahren zur Graphvisualisierung

Eine automatisierte Graphvisualisierung $\gamma = (\gamma_V)$ basiert auf einer automatisierten Platzierung der Graphelemente in einer Ebene ($\gamma_v : V \mapsto \mathbb{R}^2$) sowie auf einer Darstellung, welche die Platzierungsinformationen zum eigentlichen Zeichnen der Graphelemente verwendet. Für gewöhnlich werden Knoten dabei durch Punkte, Kreise oder andere geometrische Objekte symbolisiert. Graphvisualisierungen verwenden dabei im einfachsten Fall gerade Linien, um Kanten darzustellen. Kanten eines gerichteten Graphen werden durch eine auf den Zielknoten zeigende Pfeilspitze symbolisiert.

Etwas komplexer sind Graphvisualisierungen $\gamma = (\gamma_V, \gamma_E)$, welche Polygone zur Darstellung von Kanten unterstützen. Hierbei wird jeder Kante eine möglicherweise leere endlich Menge an Koordinaten zugewiesen. Diese Koordinaten definieren Streckenzugstützpunkte einer solchen Darstellung: $\gamma_E = E \mapsto (\mathbb{R}^2)^*$.

Im späteren Verlauf dieser Arbeit werden hauptsächlich kräftebasierende und hierarchische Layoutverfahren genutzt. Die Grundprinzipien dieser beiden Klassen von Algorithmen werden daher nun kurz vorgestellt.

Ebenenweise Graphvisualisierungen Für eine Visualisierung $\sigma = (\sigma_V, \sigma_E)$ gerichteter Graphen ist es wünschenswert, die Richtung der Kanten deutlich und somit möglichst einheitlich darzustellen. Enthält ein Graph keine Zyklen, so ist es möglich, alle Kanten in eine Richtung zeigen zu lassen. Enthält ein Graph Zyklen, so ist es erstrebenswert, eine möglichst große Zahl an Kanten in eine Richtung zeigen zu lassen. Bereits 1977 und 1980 stellten Warfield [63] und Carpano [64] erste Arbeiten zu diesem Thema vor. Größere Bekanntheit erlangte jedoch die Arbeit von Sugiyama und Koautoren von 1981 [65]. Der Sugiyama Algorithmus umfasst vier Schritte, die detailliert unter anderem in [65, 53] dargestellt sind und auf die hier nur kurz eingegangen wird. Die folgende Beschreibung basiert auf der Annahme, dass die Knoten einer Hierarchieebene horizontal angeordnet werden und untergeordnete Hierarchiestufen jeweils unter der zugehörigen höheren Ebene dargestellt werden.

1. *Entfernen von Zyklen:* Enthält der darzustellende Graph keine Zyklen, so kann dieser Schritt übersprungen werden. Sind Zyklen vorhanden, so müssen in dieser Phase so lange ausgewählte Kanten entfernt oder umgedreht werden bis ein zyklenfreier Graph entsteht. Nach Abschluss von Phase 4 müssen diese Änderungen am Graphen rückgängig gemacht werden. Die Ermittlung der minimalen Zahl an Kanten, die umgedreht werden müssen, ist ein Problem der Komplexitätsklasse NP, welches praktisch nur für eine sehr kleine Zahl an Kanten genau gelöst werden kann. Heuristiken werden daher zur angenäherten Lösung dieses Problems eingesetzt [53].

2. *Zuordnung der Knoten zu Ebenen:* Resultat von Phase 1 ist ein gerichteter, azyklischer Graph (DAG). Es wird nun jedem Knoten des Graphen eine Ebene so zugeordnet, dass alle Kanten in eine Richtung zeigen. Zeigt eine Kante nach dieser Zuordnung nicht direkt auf die nachfolgende Ebene, so werden solange temporäre Hilfsknoten eingefügt bis jede Kante auf die nächste Ebene verweist.

3. *Verringerung der Anzahl an Kantenkreuzungen:* Die Anzahl der Kantenkreuzungen in der entstehenden Darstellung ist direkt und ausschließlich von der Reihenfolge der Knoten in einer Ebene abhängig. In einem *planaren Graphen* ist die minimale Kreuzungszahl gleich null. Diese Eigenschaft lässt sich mit linearem Aufwand überprüfen [66]. Bei nicht-planaren Graphen werden zur Minimierung der Kreuzungszahl Heuristiken eingesetzt, beispielsweise die Barycenter-Heuristik [67].

4. *Zuweisung der Koordinaten:* Die vertikale Knotenpositionierung ergibt sich direkt aus der dem jeweiligen Knoten zugewiesenen Ebene. Die horizontale Positionierung muss so erfolgen, dass die in der vorherigen Phase berechneten Reihenfolgen eingehalten werden. Abschließend werden die temporären Knoten zur Zuweisung von Knotenstützpunkten verwandt und dann entfernt. Umgedrehte Kanten werden in ihre ursprüngliche Richtung gedreht und falls Kanten entfernt wurden, so werden diese wieder eingefügt.

Basierend auf der Graphstruktur und den berechneten Koordinaten für Knoten und Kantenstützpunkte erfolgt das Zeichnen des Graphen.

Kräftebasierende Graphvisualisierungen Graphvisualisierungen $\sigma = (\sigma_V)$, welche auf einem physikalischen Kräftemodell basieren, werden in der Praxis auf Grund ihrer Flexibilität sehr häufig eingesetzt. Diese Klasse von Graphvisualisierungsalgorithmen ist sowohl für gerichtete als auch für ungerichtete Graphen einsetzbar. Kantenstützpunkte werden dabei im Allgemeinen nicht berechnet. Detaillierte Beschreibungen hierzu finden sich beispielsweise in [68].

Ein kräftebasiertes Verfahren besteht aus zwei Komponenten, dem Kräftemodell und einem Optimierungsverfahren zum Finden einer Platzierung der Knoten. Dabei sollen sich die auf die Knoten einwirkenden, simulierten Kräfte möglichst aufheben und somit die Summe der Kräfte möglichst gering sein. Das Kräftemodell kann dabei für eine Vielzahl von Kriterien erweitert werden. Es können Kräfte definiert werden, welche zu einer Vermeidung von Überlappungen von Knoten oder zur Verringerung der Anzahl von Kantenkreuzungen führen. Modifizierte Kräftemodelle können das Layout von ausgewählten isomorphen Subgraphen konstant halten und diese deutlich hervorheben [69]. Zur algorithmischen Lösung des Optimierungsproblems werden Gradientenmethoden, Simulated Annealing oder evolutionäre Verfahren angewandt.

Spezielle Layoutverfahren für biologische Netzwerke

Die Layoutergebnisse von Standardverfahren wie beispielsweise zirkuläre, orthogonale, baumbasierende und kräftebasierende Verfahren sind für ansprechende Darstellungen biologischer Netze oft nicht gut geeignet [70]. Dies gab den Anstoß zur Entwicklung einer Reihe von spezialisierten Algorithmen, welche im Folgenden kurz vorgestellt werden.

Karp & Paley Karp and Paley [71] entwickelten einen Divide-and-Conquer-Algorithmus zur Identifikation von Subgraphtopologien (linear, kreisförmig, verzweigt). Die Platzierung kurzer linearer Subgraphstrukturen erfolgt horizontal oder vertikal entlang einer Linie. Für längere lineare Subgraphen wird ein in regelmäßigen Abständen die Richtung änderndes lineares Layout verwendet (Snake-Layout). Ein Kreislayout wird für zirkuläre Subgraphen angewandt, für verzweigte Strukturen werden Baumlayouts verwandt. Anschließend erfolgt die Platzierung von Kofaktor- und Enzymknoten. Die Systeme EcoCyc, BioCyc und MetaCyc enthalten eine Implementierung dieses Verfahrens [72, 73].

Becker & Rojas Becker und Rojas entwickelten basierend auf den Ideen von Karp und Paley ein kräftebasierendes Layout, das mit einer Reihe von Heuristiken erweitert wurde [74]. Zu Beginn erfolgt eine Suche nach dem längsten Zyklus im Graphen. Die nicht zu zyklischen Subgraphen gehörenden Knoten mit mindestens zwei Verbindungen zu Knoten einer zyklischen Substruktur werden innerhalb der kreisförmig gelayouteten Subgraphen platziert. Die verbleibenden äußeren starken Zusammenhangskomponenten werden rekursiv mit demselben Ansatz verarbeitet. Die Platzierung der identifizierten Subgraphen erfolgt mithilfe eines kräftebasierten Verfahrens, welches Knotenüberlappungen vermeidet.

Wegner & Kummer Wegner und Kummer stellten eine Weiterentwicklung des Algorithmus von Becker und Rojas vor [75]. Die Autoren argumentierten, dass kleinere zyklische Strukturen in metabolischen Pathways von größerer Bedeutung sind und deshalb beim Layout bevorzugt werden sollten. Ähnlich wie im Ansatz von Karp et al. [76], berücksichtigt dieser Algorithmus eine Liste von vordefinierten Kofaktoren (zum Beispiel ATP, NADP...). In Vorbereitung auf das eigentliche Layout werden Kofaktoren und andere Knoten, welche Teil von mehr als einem Zyklus sind, so in mehrere Knoten aufgespalten, dass jeder dieser Knoten nur noch eine Verbindung zu anderen Knoten hat. Die Anzahl der erlaubten Kantenkreuzungen kann auch für den gesamten Graphen festgelegt werden. Dazu werden gegebenenfalls weitere Knoten aufgespalten.

Schreiber Im Rahmen des BioPath-Projektes [70], welches das Ziel verfolgte, die grafische Qualität und den Detailreichtum des Boehringer-Posters biochemischer Reaktionswege elektronisch wiederzugeben, wurde ein hierarchisches Layoutverfahren entwickelt [77].

Zu Beginn erfolgt das Layout der Verbindung von Reaktions- zu Enzym- und Kofaktorknoten unabhängig vom Rest des Graphen. Diese Strukturen werden anschließend durch Reaktionsknoten ersetzt und ein hierarchisches Layout ähnlich dem Sugiyama-Verfahren [67] unter Berücksichtigung von Layoutconstraints (z. B. Abstände und Ausrichtung von Knoten) berechnet. Anschließend wird die zu Beginn erfolgte Ersetzung der Subgraphen durch Reaktionsknoten rückgängig gemacht.

Dogrusoz et al. Dogrusoz und Koautoren entwickelten ein Layoutverfahren für biochemische Reaktionsnetze unter Berücksichtigung von Zellkompartimentinformationen (Angaben zum Ort der modellierten Reaktionen) [78]. Um die Ablaufgeschwindigkeit des Verfahrens zu erhöhen, werden iterativ Teile des Graphen, welche reine Baumstrukturen bilden, entfernt und erst anschließend in zwei Stufen kräftebasierte Knotenplatzierungen berechnet. Die Knotenplatzierung, die sich aus dem ersten Layoutdurchlauf ergibt, dient der Bestimmung der zur Darstellung der Kompartimente benötigten Flächen (Rechtecke). Nachfolgend werden initial entfernte Knoten wieder hinzugefügt und die Knotenplatzierung unter Berücksichtigung der Kompartimentflächen berechnet. Dieses Layout ist im PATIKA-System (Pathway Analysis Tool for Integration and Knowledge Acquisition System) implementiert [79].

2.3.3 Netzwerkintegrierte Datenvisualisierung

Methoden und Software, welche Forscher bei der Interpretation von experimentellen biologischen Daten unterstützen, sind ein wichtiges Forschungsgebiet der Bioinformatik. Beispiele für solche nützlichen Verfahren sind Scatterplots [80] und Clustering-Verfahren mit Visualisierung der Resultate. Zur Visualisierung von Daten können eine Reihe grafischer Attribute wie Objektposition, Größe, Farbe, Textur oder die äußere Form modifiziert werden, um auf verschiedenen Skalen erfasste Merk-

male zu repräsentieren[6]. Messdaten im Bereich Biologie werden größtenteils quantitativ ermittelt, Genexpressionsdaten aber manchmal vereinfacht unter Verwendung einer ordinalen Skala (Reduzierung auf die Level 'hoch reguliert', 'unverändert', 'runter reguliert') dargestellt.

Card [56] befasste sich mit der Effektivität unterschiedlicher grafischer Attribute zur Datenvisualisierung. Er stellte fest, dass die Grafikattribute Position und Größe (welche beispielsweise in Diagrammen und beim Shape-Coding Verwendung finden) sehr gut zur Visualisierung von quantitativ und nominal erfassten Merkmalen geeignet sind. Eine Graustufeneinfärbung von Objekten ist gut zur Visualisierung quantitativer Daten und sehr gut für ordinale Daten geeignet. Vergleiche zwischen quantitativen und ordinalen Werten können gut mithilfe von Farbskalen oder durch Variation der Objekttextur visualisiert werden. Unterschiedliche Objektformen sind nur schlecht zur vergleichenden Visualisierung von quantitativen und ordinalen Daten geeignet.

Im Folgenden werden die zwei Standardmethoden Farb- und Größencodierung zur Visualisierung von experimentellen Daten im Kontext biologischer Netzwerke vorgestellt. Um den Vergleich der unterschiedlichen Visualisierungsformen zu erleichtern, werden für beide Visualisierungsformen Beispieldaten aus der Veröffentlichung [81] verwendet.

Farbcodierung (Color-Coding)

Farbcodierungstechniken werden oft zur Visualisierung von besonders umfangreichen experimentellen Datensätzen, wie sie beispielsweise bei der Analyse der Genexpression anfallen, verwendet. Im einfachsten Fall wird das Resultat einer einzigen Messung visualisiert. Es kann aber auch das Verhältnis zweier Messungen visualisiert werden (Abbildung 2.4, links). Die Farbe eines Netzwerkelements oder neben Graphknoten oder Kanten platzierter kleiner geometrischer Objekte wird durch eine Transformationsfunktion oder eine diskrete Zuordnungstabelle festgelegt. Häufig werden unterschiedliche Farben zur Darstellung der Minima und Maxima verwendet. Eine dritte Farbe, meist weiß oder schwarz, repräsentiert Nullwerte und somit Situationen, in denen keine Veränderung in Abhängigkeit eines Experimentfaktors festgestellt werden kann.

Beispiele für Softwaresysteme, welche eine statische Netzwerkvisualisierung und die Einbettung und Darstellung von Messwerten unter Zuhilfenahme von Farbcodierung unterstützen, sind KaPPA-View [82] und MapMan [83]. Dynamische Netzwerkvisualisierung und Farbcodierung werden beispielsweise von SimWiz [84] und vom im weiteren Verlauf der Arbeit detailliert beschriebenen System VANTED [85, 86] unterstützt.

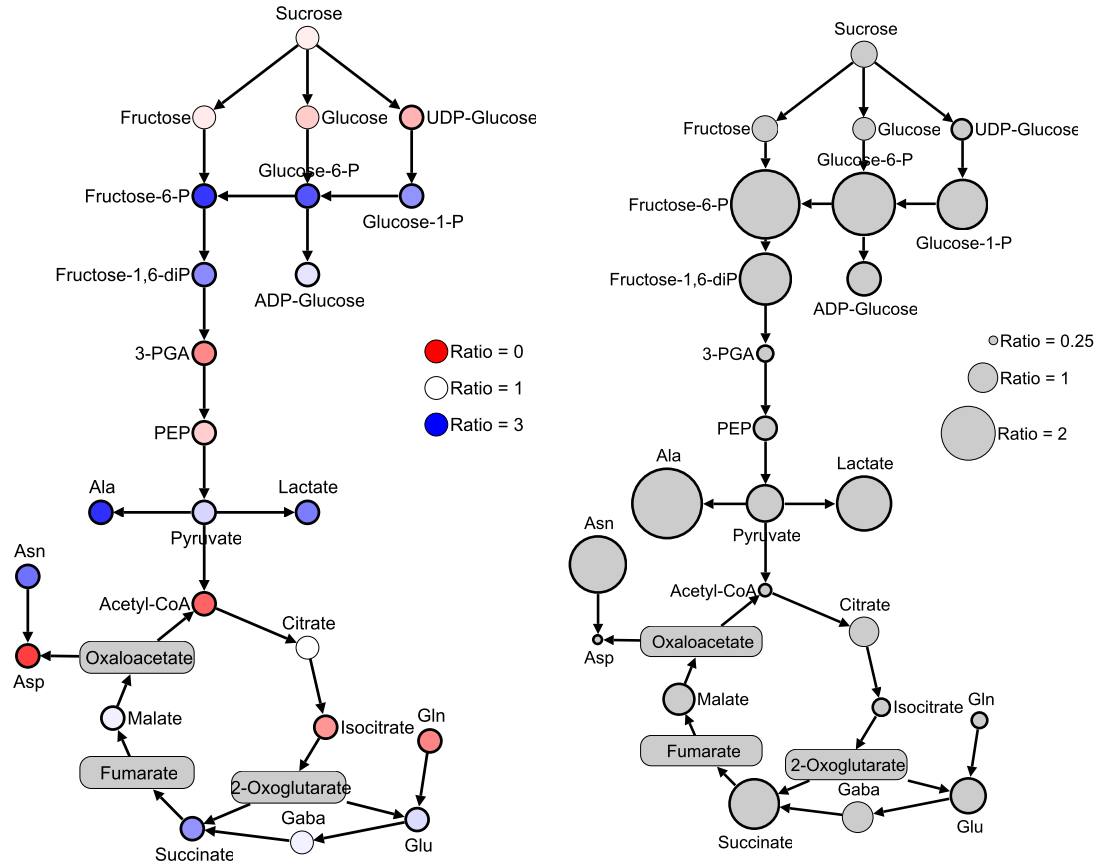

Abbildung 2.4: Links: Verwendung eines Farbcodes zur Visualisierung einer Verhältniszahl, ermittelt aus zwei unterschiedlichen Umweltbedingungen (erhöhte und normale Sauerstoffkonzentration) beim Wachstum von Sojabohnensamen. Details hierzu finden sich in [81]. Rechts: Visualisierung derselben Daten mithilfe von Größencodierung.

Form-/Größencodierung (Shape-Coding)

Unter Shape-Coding versteht man die Nutzung von unterschiedlichen Objektformen oder Größen zur Darstellung von Datenpunkten. Die Objektform selbst wird nur sehr selten zur Visualisierung von Experimentdaten variiert, normalerweise steht die Form eines Netzwerkelements in Verbindung zu anderen Aspekten wie dem Typ des Netzwerkelements. Im Allgemeinen werden Knotengrößen oder Kantenstärken variiert, um experimentelle Daten im Netzwerkkontext darzustellen. Die Variation der Kantenstärke ist beispielsweise gut geeignet, um metabolische Flüsse in einem Netzwerk darzustellen [87]. Wie im Falle der Farbcodierung werden auch hier Transformationsfunktionen oder Zuordnungstabellen verwendet, um einzelne Messwerte oder Verhältniszahlen zu visualisieren (Abbildung 2.4, rechts).

Animationstechniken

Einige Visualisierungssysteme, beispielsweise SimWiz [84] und KEGGanim [88], nutzen Animation in Verbindung mit Farb- oder Größencodierung. Animation ist jedoch mit einer Reihe von Nachteilen verbunden. Der Vergleich von Daten weit entfernter Zeitpunkte ist schwierig, außerdem können Animationen nicht in gedruckten Publikationen Verwendung finden. Stattdessen werden für den Druck ausgewählte Zeitpunkte separat dargestellt. Die Implementation von dynamischen Visualisierungssystemen ist aufwendiger, da die Geschwindigkeit der Animation und der User-Interaktionen auf die Erwartungen der Nutzer, unabhängig vom Umfang der dargestellten Daten, abgestimmt sein muss.

[6] Die wichtigsten Skalentypen sind Kardinalskala zur qantitativen Erfassung von Merkmalen (arithmetische Operationen sind möglich), Ordinalskala (eine Reihenfolge kann gebildet, aber keine Differenzen oder Verhältniszahlen berechnet werden) und die Nominalskala (Wertausprägungen können nur auf Identität geprüft werden).

3 Methodik

Massiv-parallele Analysetechniken im Bereich der Biologie erzeugen zunehmend umfangreichere experimentelle Datensätze. Dies ermöglicht bereits heute eine Top-Down-Sicht auf die Biochemie eines Organismus. Die Interpretation der Daten ist häufig durch die verfügbaren Analyse- und Visualisierungsmethoden beschränkt. Das Ziel der hier entwickelten Methodik ist, große Mengen von Daten mit hinreichender Detailgenauigkeit strukturiert im Computer zu speichern, um es dem Anwender auf dieser Basis zu ermöglichen, interaktiv leicht lesbare und verständliche Datenvisualisierungen zu generieren sowie verschiedene (statistische) Auswertungen der Daten vorzunehmen. Dazu werden Experimentdaten in Beziehung zu Graphknoten und -kanten, welche bestimmte biologische Einheiten repräsentieren, gesetzt und passende Analysemethoden benötigt. Die folgenden Hauptkomponenten werden dazu entwickelt und vorgestellt:

Datenmodelle und Datenmapping

- Datenmodell zur Abbildung und Strukturierung von Experimentdaten
- Graphmodell für biologische Netzwerke und Klassifikationshierarchien
- Methoden zur Verknüpfung von Daten- und Graphmodell

Experimentdaten- und Graphvisualisierung

- Methoden zur Visualisierung der netzwerkintegrierten Experimentdaten
- Auswahl geeigneter Graphlayoutverfahren für verschiedene Graphen

Interaktive Exploration

- Interaktionstechniken zur vereinfachten Anpassung der Visualisierung
- Navigationstechniken zur effizienten Graphvisualisierung und -exploration

Datenanalyse

- Methoden zur Analyse netzwerkintegrierter Experimentdaten

Diese Teilkomponenten sollen dabei nicht Teil festgelegter Workflows werden, sondern einen „Baukasten“ bilden. Einzelne Bestandteile sollen herauslösbar sein, um diese interaktiv für vielfältige Anwendungsszenarien einsetzen zu können. Dazu werden in diesem Kapitel sowohl neue Interaktionstechniken mit Bezug zur Pathway-Exploration entwickelt als auch einige der im vorherigen Kapitel vorgestellten grundlegenden Interaktionstechniken an den Anwendungskontext angepasst.

3.1 Definition und Verknüpfung von Experiment- und Netzwerkdaten

Um flexibel die Ergebnisse von ein- oder mehrfaktoriellen Versuchsplänen im Computer verarbeiten und beispielsweise biologischen Netzwerken zuordnen zu können, müssen Datenmodelle für Experimentdaten und Netzwerke definiert werden.

Im nachfolgenden Abschnitt wird ein die verschiedenen biologischen Domänen übergreifendes Datenmodell zur Verwaltung von Experimentdatensätzen präsentiert. In Kapitel 3.1.2 werden Graphmodelle für biologische Netzwerke und Klassifikationshierarchien vorgestellt. Anschließend wird in Kapitel 3.1.3 auf das Datenmapping, die Zuordnung von Experimentdaten zu Elementen des Graphmodells, eingegangen.

3.1.1 Datenmodell zur Verarbeitung von biologischen Experimentdaten

In Abhängigkeit vom biologischen Untersuchungsgegenstand, den verwendeten Messmethoden und den Experimentbedingungen unterscheiden sich die Angaben zur umfassenden Dokumentation eines Experiments von ihrer Struktur und ihrem Umfang her sehr stark. Aus diesem Grund gibt es derzeit keinen die verschiedenen biologischen Domänen übergreifenden Standard zur Experimentbeschreibung, sondern unterschiedliche für bestimmte Domänen geeignete Normierungsbemühungen. Für das Gebiet Genomics wurde MIAME [89] („minimum information about a microarray experiment“) und das zugehörige Objektmodell MAGE (microarray gene expression) entwickelt, für den Bereich Proteomics wurde PEDRo [90] vorgeschlagen und ArMet [91] fokussiert auf den Bereich Metabolomics. Entsprechende Dokumentationsangaben sind für die in dieser Arbeit betrachteten Analysen und Visualisierungen zu umfangreich und größtenteils unnötig, daher wird auf eine Beschreibung dieser Standards verzichtet und auch kein neues umfangreiches Datenmodell zur Dokumentation der Experimentdurchführung vorgeschlagen. Die objektorientierte Datenmodellierung erfolgt vielmehr mit dem Ziel, einen für die Visualisierung und Analyse der in den Bereichen Genomics, Proteomics und Metabolomics am häufigsten anfallenden Daten hinreichenden Satz von Klassen und Attributen zu definieren. Wichtige Grundlage zur Entwicklung des Modells waren Interviews bei interessierten Biologen am IPK-Gatersleben, anderen Forschungseinrichtungen, Universitäten sowie Literatur- und Datenbankrecherchen, beispielsweise hinsichtlich der Inhalte der Datenbanken FLAREX [92] und KEGG Expression [93]. Von Bedeutung sind Anzahl und Kombination der Versuchsfaktoren in typischen Experimentplänen sowie die von Experimentatoren als essentiell betrachteten Attribute zur Beschreibung der erfassten Daten.

Datenmodell für die Verwaltung vollständiger Experimentdatensätze

Experimentelle Messdaten werden durch die Klasse *measurement* repräsentiert und durch Zuordnung zur experimentdatenübergreifenden Klasse *substance* sowie durch Einordnung in eine Hierarchie, beginnend mit der Klasse *experiment* über die Klas-

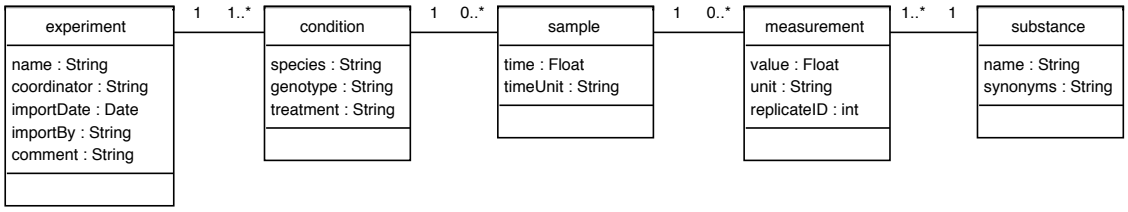

Abbildung 3.1: Datenmodell zur Verwaltung experimenteller Messdaten (UML-Klassendiagramm).

sen *condition* und *sample* näher beschrieben (einen Überblick gibt Abbildung 3.1). Die biologische Bedeutung und weitere Details zur Motivation der Modellierung einzelner Klassen sind in Tabelle 3.1 übersichtlich dargestellt. In den Tabellen 3.2 bis 3.6 sind die Klassenattribute beschrieben.

Die Klasse *substance* repräsentiert experimentdatenübergreifend im Rahmen von biologischen Experimenten numerisch quantifizier- und identifizierbare Substanzen, alternative Bezeichner können mithilfe des Attributs *synonyms* berücksichtigt werden. Die Klasse *experiment* repräsentiert Experimente und dient als zentraler Anknüpfungspunkt für die zur Beschreibung der Experimentfaktoren notwendigen Klassen. Sie steht mit der Klasse *condition* über eine 1:n-Verbindung in Beziehung, *condition* steht weiterhin in 1:n-Beziehung zu *sample*. Die Klassen *condition*, *sample* und *measurement* dienen zur Repräsentation der Experimentfaktoren. *condition* wird für nicht-numerische Experimentfaktoren wie unterschiedliche genetische Linien und Umweltbedingungen verwendet, *measurement* repräsentiert die numerischen Messdaten (Wert und Einheit mithilfe der Attribute *value* und *unit*), sowie Replikatinformationen. Die Klasse *sample* steht dazu mit *measurement* in 1:n-Beziehung. Die Zuordnung der Messwerte zu Substanznamen und -synonymen erfolgt mittels einer n:1-Beziehung zwischen den Klassen *measurement* und *substance*.

Für die genannten Klassenbeziehungen muss jedem übergeordneten Objekt jeweils mindestens ein untergeordnetes Objekt zugeordnet sein ($n > 0$). Dies gilt jedoch nicht für die Beziehungen zwischen den Klassen *condition* und *sample* sowie *sample* und *measurement*. Es dürfen *condition*-Objekte ohne zugehörige *sample*-Objekte und *sample*-Objekte ohne zugehörige *measurement*-Objekte existieren. Dies ist wichtig, da in der Praxis häufig einzelne Messwerte oder sogar ganze Messreihen aus verschiedensten Gründen nicht erfasst werden. Auch in solchen Situationen soll die Information über die im Experimentplan berücksichtigten Faktorstufen mithilfe von Objekten der Klasse *condition* und *sample* vollständig verfügbar sein. Fehlende Messreihen können dann optional mithilfe eines Platzhalters in die Darstellung einbezogen werden.

Klasse	**Biologische Bedeutung**	**Notwendigkeit**	**Abhängigkeiten**
substance	Repräsentation von im Rahmen biologischer Experimente gemessenen und identifizierten Substanzen	Notwendig zur Identifikation und Gruppierung von Messwerten, Substanzsynonyme ermöglichen erweitertes Datenmapping	1:n-Beziehung (n>0) zur Klasse *measurement*
experiment	Repräsentation der Ergebnisse ein- oder mehrfaktorieller Versuchspläne	Ermöglicht die Zusammenfassung von logisch zusammengehörigen Datensätzen, speichert zur späteren Identifikation notwendige Angaben	1:n-Beziehung (n>0) zur Klasse *condition*
condition	Repräsentation nichtnumerischer Experimentfaktoren wie Umweltbedingungen oder genetische Linien	Dient der Unterteilung und Identifikation der zum Experiment gehörenden Datensätze	1:n-Beziehung (n≥0) zur Klasse *sample*
sample	Repräsentation von im Experimentplan berücksichtigten Zeitpunkten	Dient der zeitlichen Untergliederung der Experimentdaten	1:n-Beziehung (n≥0) zur Klasse *measurement*
measurement	Repräsentation von Messwerten, zugehörigen Einheiten und Replikatangaben	Dient der Speicherung eines numerischen Messwerts und eines Replikatidentifkators, notwendig um Messwerte verschiedener Substanzen für statistische Analysen miteinander in Beziehung zu setzen	(keine untergeordneten Klassen)

Tabelle 3.1: Biologische Bedeutung, Angaben zur Notwendigkeit und Abhängigkeiten der im Datenmodell zur Verwaltung von Experimentdaten vorgesehenen Klassen.

Attribut	**Beschreibung**
name	Der Name oder Bezeichner einer gemessenen Substanz
synonyms	Möglicherweise leere Liste von Synonymen oder alternativen Identifikatoren

Tabelle 3.2: Attribute der Klasse *substance*

Attribut	Beschreibung
name	Titel des Experiments
coordinator	Name des für das Experiment verantwortlichen Wissenschaftlers
importDate	Datums-/Zeitattribut zur Speicherung des Zeitpunkts, an dem der Datensatz zusammengestellt wurde
importBy	Name der Person, die den experimentellen Datensatz zusammengestellt hat
comment	Freitextfeld zur Speicherung beliebiger Kommentare, welches beispielsweise zur Beschreibung experimenteller Hintergründe dienen oder Angaben über Literatur- und Datenbankreferenzen enthalten kann (optional)

Tabelle 3.3: Attribute der Klasse *experiment*

Attribut	Beschreibung
species	Name oder Identifikator des untersuchten Organismus
genotype	Bezeichnung des Genotypen des untersuchten Organismus
treatment	Angaben zur Behandlung, beispielsweise Wachstumsbedingungen

Tabelle 3.4: Attribute der Klasse *condition*

Attribut	Beschreibung
time	Optionale Angabe für den Zeitpunkt der Messung
timeUnit	Optionale Angabe über die Einheit der Zeitpunktangabe

Tabelle 3.5: Attribute der Klasse *sample*

Attribut	Beschreibung
value	Numerischer Messwert
unit	Einheit
replicateID	Identifikator des Replikats

Tabelle 3.6: Attribute der Klasse *measurement*

3.1.2 Datenmodell für biologische Netzwerke und Klassifikationshierarchien

Eine wichtige Komponente einer netzwerkintegrierten Datenvisualisierung ist ein geeignetes Graphdatenmodell. Dieses soll zur flexiblen Verarbeitung von biologischen Netzwerken, von Klassifikationshierarchien sowie für die Zuordnung von Experimentdaten zu Graphelementen geeignet sein.

Die Modellierung biologischer Netzwerke als Graph basiert auf der Zuordnung der darzustellenden Informationsobjekte (Enzyme, Substrate, Produkte, Reaktionen ...) zu Knoten und Kanten eines Graphen. Im Allgemeinen werden Informationsobjekte als Knoten und Beziehungen zwischen diesen als Kanten dargestellt. Bestimmten Objekten wie Reaktionen können je nach Modellierungsansatz Knoten oder Kanten zugeordnet werden. Manchmal wird in einem einzigen Modell einigen Reaktionen Knoten, anderen Kanten zugeordnet. Neben biologischen Netzwerken sollen auch Klassifikationshierarchien mit einem grundlegenden Modell abgebildet werden. Dies führt zu folgender Definition eines zur Darstellung verschiedenster biologischer Netzwerke und Klassifikationshierarchien geeigneten abstrakten Graphmodells:

Definition 1 (Mappinggraph)**.** Sei $G = (V, E)$ ein Graph mit einer endlichen Knotenmenge V und einer endlichen Menge gerichteter oder ungerichteter Kanten E, L eine endliche Menge von Bezeichnern, T_V die Menge der möglichen Knoten- und T_E der Kantentypen, sowie M die Menge der Experimentdaten. Die Funktion $l : V, E \mapsto L$ wird Labelfunktion, $t_V : V \mapsto T_V$ Knotentypzuordnungsfunktion, $t_E : E \mapsto T_E$ Kantentypzuordnungsfunktion und $z : V, E \mapsto M$ Datenzuordnungsfunktion genannt. Die Menge der Graphelemente von G zusammen mit l, t_V, t_E und z wird Mappinggraph MG genannt und $MG = (V, E, l, t_V, t_E, z)$ geschrieben.

Der Mappinggraph bildet einen essentiellen Baustein der hier vorgestellten Methodik, da er Experimentdaten und das Graphmodell der relevanten Netzwerke mithilfe der in Abschnitt 3.1.3 beschriebenen Datenzuordnungsfunktion miteinander in Beziehung setzt. Um eine flexible Basis für unterschiedliche Netztypen zu schaffen, sind die Elemente der Mengen T_V, T_E und deren biologische Entsprechung für MG nicht spezifiziert.

Mappinggraphen verschiedener biologischer Netzwerke und Klassifikationshierarchien

Einige der im weiteren Verlauf der Arbeit vorgestellten Analyse- und Visualisierungsmethoden sind nur für bestimmte Mappinggraphen geeignet. Daher wird die Definition des Mappinggraphen in den folgenden Abschnitten beispielhaft für einige biologische Netzwerke und Klassifikationshierarchien konkretisiert.

PPI-Netzwerke Ein Protein-Protein-Interaktionsnetzwerk wird wie folgt als Mappinggraph MG_{PPI} modelliert: Alle Graphknoten sind hier vom Typ $T_V = \{\text{Protein}\}$, die ungerichteten Kanten haben den Typ $T_E = \{Interaktion\}$. Jede Kante in MG_{PPI} bildet eine (prinzipiell ungerichtete) Interaktion zwischen zwei unter be-

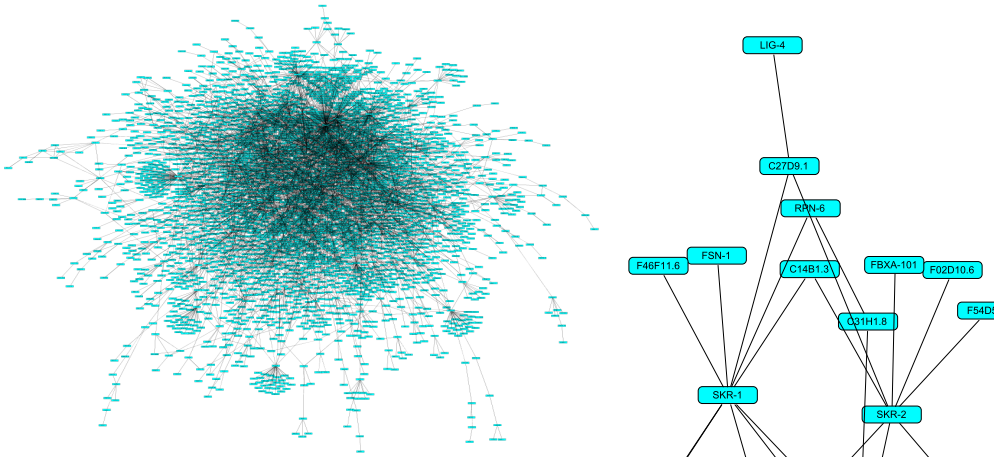

Abbildung 3.2: Links: die größte zusammenhängende Komponente des Protein-Protein-Interaktionsnetzwerkes des Wurms *C.elegans* (5418 Interaktionen und 2992 Proteine). Rechts: vergrößerter Ausschnitt. Datenquelle: GraphWeb [94], Zusammenstellung von Proteindaten aus der IntAct-Datenbank [95], Visualisierung mit VANTED.

stimmten Bedingungen interagierenden Proteinen ab. Abbildung 3.2 zeigt die Visualisierung eines Beispielgraphen.

Biochemische Reaktionsnetzwerke Für MG_{CR} sind folgende Knoten- und Kantentypen zulässig: $T_V = \{Metabolit,\ Kofaktor,\ Enzym,\ Reaktion,\ Pathwayverweis\}$ und $T_E = \{Reaktion, Link\}$. An einer biochemischen Reaktion beteiligte Substrate und Produkte werden als Knoten des Typs *Metabolit*, Kofaktoren und Enzyme als Knoten der Typen *Kofaktor* und *Enzym* modelliert und mit den zu den Reaktionen gehörenden Knoten des Typs *Reaktion* verbunden. Dadurch, dass die Produkte einer Reaktion als Substrate in eine weitere Reaktion eingehen können, entsteht ein Reaktionsweg und bei hinreichender Komplexität ein Reaktionsnetz.

Die meisten biochemischen Reaktionen sind reversibel. Daher ist die Reaktionsrichtung häufig unbestimmt. Falls keine vorherrschende Reaktionsrichtung im betrachteten Organismus bekannt ist, sind die Reaktionsknoten ausschließliche Quelle der gerichteten Kantenverbindung zu Substrat- und Produktknoten, ansonsten sind Substrate der Reaktion gerichtet mit Reaktionsknoten und diese gerichtet mit den Produktknoten verbunden. Enzym- und Kofaktorknoten sind mit den Reaktionsknoten durch ein antiparalleles Kantenpaar verbunden.

Optional verweisen Knoten des Typs *Pathwayverweis* auf ausgewählte andere biochemische Reaktionsnetzwerke. Knoten des Typs *Metabolit* sind mit Knoten des Typs *Pathwayverweis* verbunden, falls im verwiesenen biochemischen Reaktionsnetz *Metabolit*-Knoten mit gleichem Label vorkommen. Mit den Reaktionsknoten verbundene Kanten haben den Typ *Reaktion*.

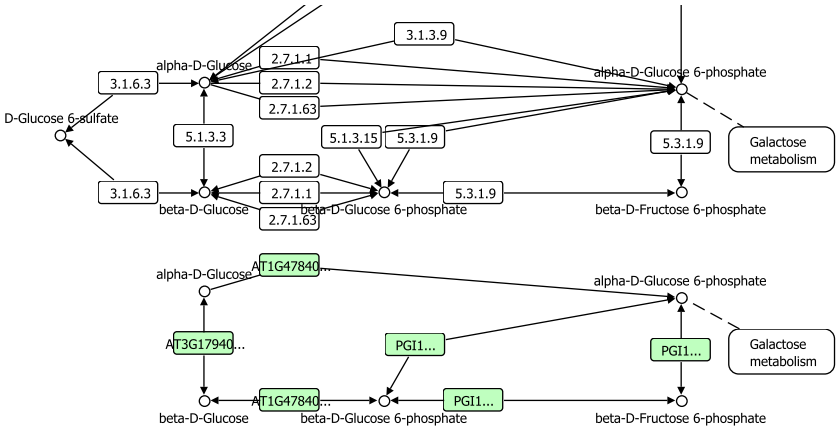

Abbildung 3.3: Zwei Ausschnitte aus dem KEGG Pathway Glycolysis, oben Referenzpathway (map00010) unten dessen organismusspezifische Variante (ath00010) für die Pflanze Ackerschmalwand (*Arabidopsis thaliana*). Dargestellt sind Knoten der Typen *Enzym* (z.B. 3.1.6.3), *Gen* (z.B. PGI1), *Metabolit* (z.B. Alpha-D-Glucose) und *Pathwayverweis* (Galactose metabolism). Kanten des Typs *Link* sind gestrichelt ohne Pfeilspitzen dargestellt.

KEGG-Pathways Die Kyoto Encyclopedia of Genes and Genomes (KEGG) Pathway Datenbank ist eine häufig verwendete Ressource für Forscher aus den Gebieten Biologie und Bioinformatik. Diese Datenbank enthält eine große Anzahl von metabolischen und regulatorischen Pathwayabbildungen, zwischen denen interaktiv navigiert werden kann. Referenzinformationen zu den dargestellten Informationsobjekten sind mithilfe von Webseitenlinks auf einfache Art und Weise erreichbar. Die Darstellung der Pathways als Bitmap-Bild entspricht einer statischen Visualisierung, es werden aber auch bestimmte dynamische Visualisierungsaspekte, wie Navigation und Einfärbung von Informationsobjekten, unterstützt. Ein Großteil der dargestellten Information wird vom KEGG-System auch in maschinenlesbarer Form, als KEGG-XML-Dateien (KGML) verfügbar gemacht. Diese Dateien können interpretiert werden, um auf dieser Basis dynamisch visualisierbare und editierbare Mappinggraphen zu konstruieren, welche für das in dieser Arbeit wichtige Datenmapping geeignet sind. Eine detaillierte Beschreibung der KEGG-XML-Struktur und deren Überführung in ein dynamisches Graphmodell wurde vom Autor bereits veröffentlicht [96]. Im Folgenden werden die wesentlichen Strukturelemente der KEGG-Pathways kurz vorgestellt: Für KEGG-Mappinggraphen MG_{KEGG} (Beispiel in Abbildung 3.3) gelten folgende, von der KGML-Struktur abgeleitete Knoten- und Kantentypen: $T_V = \{Ortholog,\ Enzym,\ Gen,\ Gengruppe,\ Metabolit,\ Pathwayverweis\}$ und $T_E = \{ECrel,\ PPrel,\ GErel,\ PCrel,\ Link\}$.

Die Knotentypen *Enzym*, *Gen* und *Metabolit* werden entsprechend ihrer biologischen Bedeutung verwendet. *Gengruppe* wird als Modellelement in KEGG-Pathways verwendet, um Beziehungen zwischen einer ganzen Gruppe von Genen zu anderen Knoten des Graphen vereinfacht darzustellen. Der Knotentyp *Ortholog* wird als ab-

strakte Entsprechung der einheitlichen Funktion einer Reihe verschiedener, in unterschiedlichen Organismen zu findenden Genen verwendet. Im KEGG-System gibt es für orthologe Gene mit gleicher Funktion KO-Einträge in der KEGG-BRITE-Datenbank. *Metabolit*-Knoten sind entweder wie in MG_{CR} mit *Enzym*- oder mit *Gen*-Knoten verbunden. In den KO-Referenzpathways des KEGG-Systems sind die Metabolite mit Knoten des Typs *Ortholog* verbunden. Knoten des Typs *Pathwayverweis* verweisen wie in MG_{CR} auf ausgewählte Pathways und sind gegebenenfalls mit Knoten des Typs *Metabolit* verbunden.

Die verschiedenen Kantentypen haben folgende Bedeutung: *ECrel* steht für eine Enzym-Enzym-Beziehung, bei der zwei Enzyme zwei aufeinander folgende Reaktionsschritte katalysieren, *PPrel* steht für eine Protein-Protein-Interaktion, *GErel* bedeutet Genexpression, *PCrel* steht für eine Interaktion zwischen einem Protein und einem Metabolit. Die Kanten des Typs *Link* dienen wie in MG_{CR} dem Verweis auf andere Mappinggraphen, hier vom Typ MG_{KEGG}.

Pathwayübersichtsgraphen Ein Pathwayübersichtsgraph MG_O ist ein Mappinggraph mit Knoten vom Typ $T_V = \{\text{Pathwayverweis}\}$ und Kanten des Typs $T_E = \{Link\}$. Jeder Knoten v_{MG_i} in MG_O repräsentiert einen anderen Mappinggraphen MG_i des Typs MG_{CR} oder MG_{KEGG}. Jede gerichtete Kante $e = (v_{MG_a}, v_{MG_b})$ in MG_O repräsentiert einen Verweis ausgehend von dem durch den Anfangsknoten repräsentierten Mappinggraphen (MG_a) zum durch den Endknoten repräsentierten Mappinggraphen (MG_b). Falls MG_b einen Verweis zu MG_a enthält, existiert in MG_O eine zu e anti-parallele Kante $f = (v_{MG_b}, v_{MG_a})$.

Erweiterte Pathwayübersichtsgraphen Ein *erweiterter Pathwayübersichtsgraph* MG_{OE} ist ein Pathwayübersichtsgraph, welcher zusätzlich die Elemente eines oder mehrerer KEGG-Pathways oder Elemente von biochemischen Reaktionsnetzwerken enthält. T_V und T_E ergeben sich jeweils aus der Vereinigung der zulässigen Knoten- und Kantentypen von MG_O und MG_{KEGG} bzw. MG_O und MG_{CR}. Details zur Konstruktion erweiterter Pathwayübersichtsgraphen sind Thema des Abschnitts 3.3.3.

Gene-Ontology-Hierarchie Im Rahmen des Gene-Ontology-Projekts, welches bereits in Kapitel 2.1.1 vorgestellt wurde, wird eine stetig größer werdende Zahl von GO-Termen zur einheitlichen und vergleichbaren Beschreibung der Bedeutung von Genen gepflegt und miteinander in Beziehung gesetzt. Diese GO-Terme bilden im Graphmodell einen gerichteten, azyklischen Graphen (DAG) mit den drei Quellknoten „biological process“ (GO: 0008150), „molecular function“ (GO: 0003674) und „cellular component“ (GO: 0005575).

GO-Mappinggraphen MG_{GO} können entweder als vollständige Repräsentation der Gene-Ontology-Hierarchie konstruiert oder anhand der GO-Annotation eines konkreten Datensatzes erstellt werden. Die genaue Vorgehensweise dazu wird in Abschnitt 3.1.3 erläutert. MG_{GO} enthält im ersten Fall ausschließlich Knoten des Typs $T_V = \{Klassifikation\}$, im zweiten Fall Knoten der Typen $T_V = \{Klassifikation,\ Gen\}$ (siehe Abbildung 3.4). Für die Kantentypen in MG_{GO} gilt $T_E = \{is\_a,$

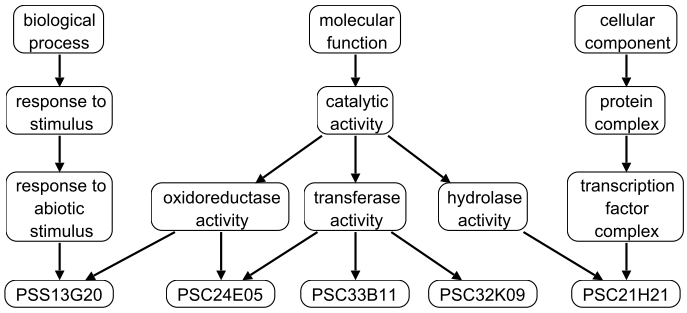

Abbildung 3.4: Fünf Gene (Senken im Mappinggraph) und eine in der Tiefe auf drei Stufen begrenzte, dazugehörige Gene-Ontology-Funktionsklassifikation.

part_of, Annotation}. Gerichtete Kanten sind je nach definierter Beziehung zwischen den GO-Termen vom Typ *is_a* oder *part_of*. Es gibt maximal drei Quellen im Graphen vom Typ *Klassifikation*, diese sind Startpunkt der in Kantenrichtung immer spezieller werdenden Genfunktionsbeschreibung. Basiert MG_{GO} auf einem Experimentdatensatz, ist von jedem Klassifikationsknoten über einen Weg mindestens ein *Gen*-Knoten erreichbar. Alle mit Knoten des Typs *Gen* verbundenen Kanten sind vom Typ *Annotation*.

KEGG-BRITE-Hierarchie Die KEGG-BRITE-Datenbank[1] spezifiziert eine Klassifikationshierarchie zur Beschreibung verschiedenster Aspekte biologischer Systeme. Sie enthält unter anderem Informationen zur Bedeutung von Genen und Enzymen sowie deren Zuordnung zu allgemeinen Funktionsklassen und KEGG-Pathways. KEGG-BRITE ergänzt die KEGG-Pathway-Hierarchie, welche auf molekulare Interaktionen und Reaktionen beschränkt ist.

Der Mappinggraph MG_{BRITE} wird entweder als vollständige Repräsentation der KEGG-BRITE-Hierarchie oder auf der Basis eines konkreten Datensatzes konstruiert (weitere Details dazu finden sich in Abschnitt 3.1.3). Werden neben der KEGG-BRITE-Datenbank weitere von KEGG angebotene Datenquellen (zum Beispiel die KEGG SOAP API) berücksichtigt, können neben *Enzym*- und *Gen*-Knoten auch *Metabolit*-Knoten als Startpunkt zur Konstruktion datenspezifischer MG_{BRITE} Mappinggraphen dienen. Für MG_{BRITE} gilt $T_V = \{$*Klassifikation, Pathwayverweis*$\}$, falls die vollständige Hierarchie unabhängig von einem Experimentdatensatz konstruiert wird und $T_V = \{$*Klassifikation, Pathwayverweis, Gen, Enzym, Metabolit*$\}$, falls eine von einem Datensatz abgeleitete Klassifikationshierarchie erstellt wird.

Definiert sind in MG_{BRITE} folgende Kantentypen: $T_E = \{$*Relation, Annotation, Link*$\}$. Gerichtete Kanten des Typs *Relation* verbinden Klassifikationsknoten untereinander und Klassifikationsknoten mit *Pathwayverweis*-Knoten. Basiert der Graph auf einem Experimentdatensatz, bilden die zur Generierung der Hierarchie verwendeten Knoten der Typen *Metabolit*, *Enzym* oder *Gen* die Senken im Graph. Sie sind über Kanten des Typs *Annotation* mit den Klassifikationsknoten und über Kanten

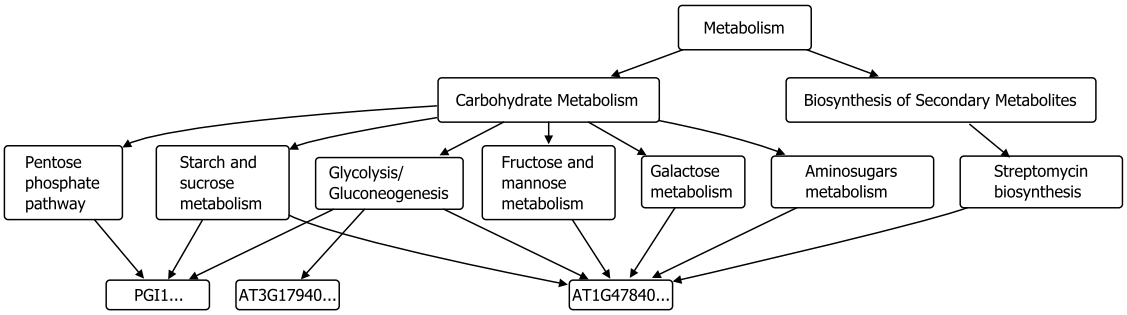

Abbildung 3.5: Die zu den drei verschiedenen *Gen*-Knoten aus Abbildung 3.3 gehörende KEGG-BRITE-Hierarchie.

des Typs *Link* mit *Pathwayverweis*-Knoten verbunden.

3.1.3 Zuordnung von Experimentdaten zu relevanten Netzwerken

Unabhängig vom Typ enthält jeder Mappinggraph die Datenzuordnungsfunktion z. Sie dient der Zuordnung bestimmter Teile eines oder mehrerer Experimentdatensätze zu Knoten und Kanten des Mappinggraphen. Zur Beschreibung der Funktion z sollen folgende drei Fragen beantwortet werden:

(1) Wie werden die vollständigen Experimentdatensätze bearbeitet, damit passende Ausschnitte einzelnen Graphelementen zugeordnet werden können?

(2) Welche alternativen Bezeichner für Graphelementlabel und Substanznamen im Experimentdatensatz sind relevant und wofür können diese verwendet werden?

(3) Wie sieht der Datenmappingalgorithmus aus und welche Komplexität hat er?

Graphelementorientiertes Experimentdatenmodell

Die im Rahmen eines Experiments gemessenen Substanzen sind im Allgemeinen nicht vollständig einem einzelnen biologischen Netzwerk zuordenbar. Es soll im Rahmen des Datenmappings möglich sein, einzelnen Graphelementen Teilmengen von Daten aus möglicherweise verschiedenen Experimenten zuzuordnen. Um dies zu ermöglichen, wird ein Datenmodell definiert, das zur Zuordnung von Teilen vollständiger Experimentdatensätze geeignet ist.

Für die Zuordnung von experimentellen Daten zu einzelnen Graphelementen wird als oberste strukturierende Einheit die Klasse *mapping* definiert (siehe Abbildung 3.6). Diese Klasse vereint die für das bisher betrachtete Experimentdatenmodell definierten Attribute der Klassen *experiment* und *substance*. Über eine $1 : n$-Beziehung ist die Klasse *mapping* mit den hinsichtlich der enthaltenen Attribute unveränderten Klassen *condition*, *sample* und *measurement* miteinander verbunden.

[1] http://www.genome.jp/kegg/brite.html

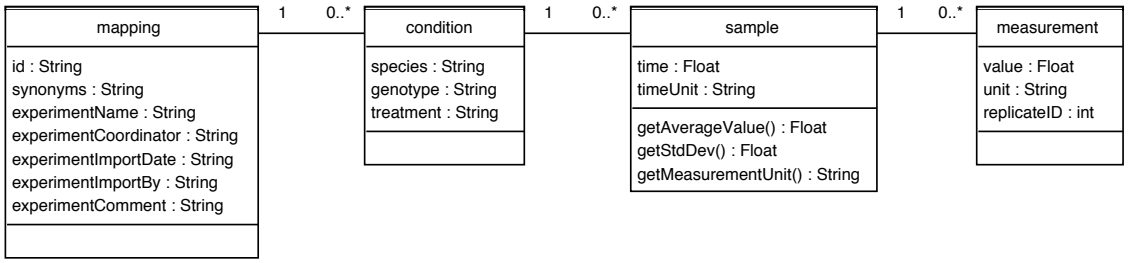

Abbildung 3.6: Für die Zuordnung von experimentellen Messdaten zu Graphelementen geeignetes Datenmodell (UML Klassendiagramm).

Objekte vom Typ *mapping*, *condition* und *sample* repräsentieren in diesem Modell die Informationen über eine einzelne gemesse Substanz. Der Substanzname wird im *mapping*-Attribut *id* gespeichert, Substanzsynonyme im Attribut *synonyms*.

Im ersten Schritt des Datenmappings wird aus jedem betrachteten *experiment*-Datensatz für die Zuordnung zu Graphelementen eine größere Menge von *mapping* Datensätzen generiert. Dabei wird für jede Kombination aus zusammengehörigen *substance* und *experiment* Einträgen ein passendes *mapping* Objekt generiert und diesem die entsprechende Teilmenge der *measurement*-, *condition*- und *sample*-Objekte zugewiesen. Im weiteren Verlauf des Datenmappings werden Objekte vom Typ *mapping* gegebenenfalls unter Berücksichtigung von alternativen Bezeichnern verschiedenen Graphelementen zugeordnet.

Alternative Bezeichner für Graphelemente und Experimentdaten

Durch Berücksichtigung von Synonymen für Graphelementlabel und Experimentdatenidentifikatoren verbessert sich die Flexibilität hinsichtlich der Zuordnung von Experimentdaten zu Elementen des Mappinggraphen.

Labelsynonyme für Knoten und Kanten des Mappinggraphen Da die Graphelementlabel der unterschiedlichen Typen der Mappinggraphen unterschiedliche IDs, Namen oder Synonyme der repräsentierten Informationsobjekte zeigen, wird die Datenzuordnung flexibel einsetzbar, wenn Labelsynonyme für Elemente des Mappinggraphen berücksichtigt werden. Dazu werden zwei Funktionen $s_V(l(v), t_V(v))$ und $s_E(l(e), t_E(e))$ mit $v, e \in MG$ definiert, die für Knoten und Kanten in Abhängigkeit vom Graphelementtyp passende Synonyme zurückliefert. $s(ge)$ liefert auf Basis dieser zwei Funktionen für jedes Graphelement $ge \in MG$ die passenden Synonyme zurück. Für einen Mappinggraph MG_{KEGG} kann s beispielsweise folgende Synonyme als Ergebnis zurückliefern: Für Knoten des Typs *Metabolit* enthält das Ergebnis die zugehörige Datenbank-ID, den Metabolitnamen und synonym verwendete Bezeichner. Für Knoten des Typs *Enzym* enthält das Ergebnis entsprechend die zum Knoten zugehörige EC-Nummer, den Enzymnamen und Datenbank-IDs von kodierenden Genen. Für Knoten des Typs *Pathwayverweis* liefert s die Vereinigung aller der im zugehörigen KEGG-Pathway-Mappinggraph enthaltenen Graphelementelabel (mit Ausnahme der enthaltenen *Pathwayverweis*-Knoten) sowie dazugehörige

Metabolit- und Enzymsynonyme zurück. Für Knoten anderen Typs und für Kanten in MG_{KEGG} liefern s_V und s_E jeweils die leere Menge als Ergebnis zurück.

Synonyme für Experimentdaten Durch Berücksichtigung von Synonymen für Substanzen im Experimentdatensatz wird die Flexibilität des Datenmappings weiter verbessert. Experimentdaten können dadurch, ohne dass die Substanznamen im Experiment verändert werden müssen oder verloren gehen, auf Mappinggraphen verschiedener Typen gemappt werden. Es ist dabei sinnvoll, den Begriff Synonym in diesem Kontext weiter zu fassen und neben alternativen Bezeichnern auch Datenbankidentifikatoren oder Datenannotationen zu berücksichtigen. Die Funktion $synonyms(m)$ mit $m \in M$, M Menge der Experimentdaten (Instanzen der Klasse *mapping*), liefert die zu jedem *mapping*-Objekt m zugehörigen und im Attribut *synonyms* gespeicherten Synonyme zurück (siehe Abbildung 3.6). Den zum *mapping*-Objekt gehörenden Substanznamen, also den Wert des Attributs *id*, liefert die Funktion $id(m)$ zurück.

Ein Biochipexperimentdatensatz kann beispielsweise Expressionsdaten für eine große Anzahl sogenannter Probe-IDs enthalten. Im Allgemeinen werden vom Anbieter für jeden Typ von Biochip Annotationsdateien bereitgestellt. Zu jeder Probe-ID können dann im Datensatz beispielsweise zugehörige Genidentifikatoren vermerkt werden. Unter Anwendung des im Folgenden beschriebenen Datenmapping Algorithmus kann ein entsprechend vorbereiteter Datensatz den Enzymknoten eines KEGG-Pathway-Mappinggraphen zugeordnet werden.

Datenmappingalgorithmus

Gegeben ist ein Mappinggraph MG und eine Menge ED von vollständigen Experimentdatensätzen, Objekte vom Typ *experiment* und die damit verbundenen Objekte der Typen *condition*, *sample*, *measurement* und *substance* (siehe Abschnitt 3.1.1). Ziel ist die sinnvolle Zuordnung von Experimentdaten zu Graphelementen. Das Ergebnis der zu MG gehörenden Funktion z entspricht dem Ergebnis des im Folgenden beschriebenen Datenmappingalgorithmus (vergleiche Pseudocode „Algorithm 1“).

Im ersten Schritt (Zeile 1) werden aus den Experimentdaten ED eine Menge M von Objekten m des Typs *mapping* konstruiert. Die Vorgehensweise dazu wurde zu Beginn dieses Abschnitts erläutert.

Für jeden Knoten und jede Kante, also jedes Graphelement $ge \in MG$ und jedes Objekt $m \in M$ wird überprüft, ob die Schnittmenge aus $A = l(ge) \cup s(ge)$ und $B = id(m) \cup synonyms(m)$ mindestens ein Element enthält. Ist dies der Fall, so gilt $m \in z(ge)$, ansonsten $m \notin z(ge)$ (Zeilen 2 bis 10). Nach der Überprüfung jedes Mappingobjekts $m \in M$ steht fest, welche Objekte dem jeweiligen Graphobjekt zugeordnet werden (Zeile 11).

Optional wird für jedes *mapping*-Objekt m_x, welches in den vorangegangen Schritten keinem einzigen Graphelement zugeordnet worden ist, ein neuer isolierter Knoten v_x erzeugt und dem Mappinggraphen hinzugefügt. Der Typ des Knotens entspricht der Art der betrachteten Substanz und kann vorgegeben oder von der *id* des Mappingobjekts abgeleitet werden. Der Datensatz wird anschließend v_x zuge-

Algorithm 1 Datenmapping

Eingabe: MG – Mappinggraph
Eingabe: ED – Experimentdaten, Objekte vom Typ *experiment*

```
1: M ⇐ generiere mapping-Objekte aus ED
2: for each Graphelement ge ∈ MG do
3:    Z ⇐ ∅
4:    for each m ∈ M do
5:       A ⇐ id(m) ∪ synonyms(m)
6:       B ⇐ l(ge) ∪ s(ge)
7:       if |A ∩ B| > 0 then
8:          Z ⇐ m ∪ Z
9:       end if
10:   end for
11:   z(ge) = Z
12: end for
```

ordnet: $z(v_x) = \{m_x\}$. Das Label des Knotens entspricht dem Wert des Attributs *id* ($l(v_x) = id(m_x)$). Dies ermöglicht dem Anwender, ursprünglich nicht im Mappinggraph berücksichtigte Elemente auf einfache Art und Weise in den vorhanden Graph zu integrieren. Außerdem ist dieser Schritt eine Voraussetzung zur Konstruktion von datenspezifischen Klassifikationsgraphen (MG_{BRITE} und MG_{GO}).

Zeitkomplexität Die Zeitkomplexität des Datenmappingalgorithmus ist quadratisch. Der zeitliche Aufwand steigt linear mit der Anzahl der zu berücksichtigenden Objekte des Typs *substance* und gleichzeitig linear mit der Anzahl zu Beginn in MG enthaltenen Graphelemente. Sind zu Beginn keine Graphelemente vorhanden, steigt beim automatischen Anlegen von Knoten der Aufwand linear mit der Anzahl der *substance*-Objekte.

Datenspezifische Klassifikationsgraphen Die Konstruktion des datenspezifischen Klassifikationsgraphen beginnt mit einem leeren Mappinggraphen. Die relevante Menge von Experimentdaten ED wird, wie oben beschrieben, im Datenmappingalgorithmus verarbeitet, wobei für jede Substanz eines Experimentdatensatzes ein neuer Knoten eines bestimmten Typs angelegt wird.

Zur Konstruktion von datenspezifischen MG_{BRITE} oder MG_{GO} Mappinggraphen werden nur neu angelegte Knoten der Typen *Gen*, *Enzym* oder *Metabolit* betrachtet. Diese Knoten sind Ausgangspunkt für das Erzeugen weiterer Knoten vom Typ *Klassifikation* und für MG_{BRITE} gegebenenfalls auch vom Typ *Pathwayverweis*. Für jeden Knoten werden die *id* und die gespeicherten Synonyme des zugeordneten *mapping*-Objekts ermittelt und für MG_{BRITE} in der KEGG-BRITE-Datenbank die zugehörige Klassifikation nachgeschlagen. Für MG_{GO} wird entsprechend die Gene-Ontology-Datenbank auf Einträge für die gefundenen Identifikatoren durchsucht. Die enthaltenen Angaben zur Klassifikation werden dazu genutzt, falls noch nicht vorhanden, entsprechende *Klassifikations-* oder *Pathwayverweis*-Knoten im Graphen anzulegen und untereinander und mit dem aktuell betrachteten Knoten auf geeig-

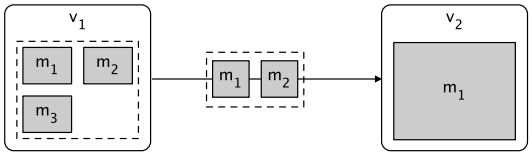

Abbildung 3.7: Für jedes einem Graphelement zugewiesene *mapping*-Objekt ist ein Bereich der Zeichenfläche des Graphelements zur Darstellung der Messwerte vorgesehen. Ein *mapping*-Objekt kann mehreren Graphelementen zugewiesen sein (m_1 allen drei Elementen, m_2 dem Knoten v_1 und der Kante).

nete Art über Kanten zu verbinden.

Die Vorgehensweise für andere Arten von Klassifikationsgraphen und andere Typen von Knoten ist ähnlich wie für MG_{KEGG} und MG_{GO}. Der Ablauf unterscheidet sich im Detail jedoch hinsichtlich der zu verwendenden Datenquellen für Klassifikationsinformationen und der damit einhergehenden Unterschiede in der Generierung der Kantenverbindungen.

Durch Zuordnung von experimentellen Daten zu Elementen des Mappinggraphen ergeben sich vielfältige Möglichkeiten zur Visualisierung von Experimentdaten direkt im Kontext der Graphvisualisierung. Dies ist Thema des folgenden Abschnitts.

3.2 Netzwerkintegrierte Visualisierung von Experimentdaten

Ein Nachteil vieler Visualisierungssysteme mit Funktionen zur Darstellung von Experimentdaten im Netzwerkkontext ist, dass diese zumeist nur einen einzelnen Datenpunkt unter Verwendung eines Farbcodes darstellen können (siehe Abschnitt 4.4).

Da einem einzelnen Graphelement eines Mappinggraphen mit der Zuordnungsfunktion z eine ganze Reihe von *mapping*-Objekten mit Messwerten, untergliedert nach Experimentbedingung (*condition*) und Zeit (*sample*), zugewiesen sein können, werden im Folgenden Methoden vorgestellt, die in der Lage sind, solche vergleichsweise komplexen Datenstrukturen zu visualisieren.

3.2.1 Aufteilung der Zeichenfläche zur Visualisierung mehrerer Datenmappings

Gemeinsame Grundlage der hier betrachteten Visualisierungsmethoden ist, dass das Darstellungsproblem beim Auftreten mehrerer *mapping*-Objekte vereinfacht wird. Die Anzahl der dem Graphelement $ge \in MG$ zugewiesenen *mapping*-Objekte sei $n_m(ge) = |z(ge)|$.

Für $n_m(ge) = 1$, wird die gesamte zur Datendarstellung verfügbare Fläche genutzt. Die verfügbare Zeichenfläche hängt für Graphknoten von der Größe der Knotendarstellung ab. Für Kanten wird eine rechteckige Fläche entlang der Graphkan-

tendarstellung in einer vom Nutzer vorgegebenen Größe verwendet (siehe Abbildung 3.7). Für $n_m(ge) > 1$ wird die verfügbare Zeichenfläche in rechteckige Abschnitte unterteilt. Die Anzahl der Abschnitte in x-Richtung n_x und somit die Breite der zur Darstellung der einzelnen *mapping*-Objekte verfügbaren Fläche („Slot"), wird entweder automatisch aus der jeweils zugewiesenen Anzahl der *mapping*-Objekte bestimmt ($n_x(ge) = \lceil\sqrt{n_m(ge)}\rceil$) oder vom Nutzer einheitlich für den gesamten Mappinggraph oder für einzelne Graphelemente individuell vorgegeben. Die Anzahl der Abschnitte in y-Richtung n_y ergibt sich aus $n_y(ge) = \lceil n_m(ge)/n_x(ge)\rceil$.

Die Daten der einzelnen *mapping*-Objekte werden dann fortlaufend, beginnend in x-Richtung, in einem der Slots der Zeichenfläche visualisiert. Ein kleiner Bereich zwischen den Slots bleibt frei, damit die Zuordnung mehrerer *mapping*-Objekte deutlich wird. Gilt beispielsweise $n_m(ge) = 5$, so wird die Zeichenfläche in $n_x(ge) = \lceil\sqrt{5}\rceil = 3$ Abschnitte in x-Richtung und $n_y(ge) = \lceil 5/3\rceil = 2$ Abschnitte in y-Richtung aufgeteilt. Slot 6 bleibt in diesem Fall frei.

3.2.2 Einfärbung der Zeichenfläche

Ist einem *mapping*-Objekt nur ein Sample zugeordnet, so kann die gesamte Zeichenfläche eines Slots gleichmäßig entsprechend dem Sample-Durchschnittswert eingefärbt werden. Enthält der *mapping*-Datensatz mehr als eine Experimentbedingung (*condition*) oder mehr als einen *sample*-Zeitpunkt, so wird die Fläche des einzelnen Slots vertikal unterteilt, um verschiedene Experimentbedingungen darzustellen. Um die Werte einzelner Zeitpunkte voneinander abzugrenzen, wird sie horizontal unterteilt (siehe Abbildung 3.8). Ist für eine bestimmte Kombination aus Experimentbedingung und Zeitpunkt kein Messwert verfügbar, so bleibt die entsprechende Fläche frei. Dadurch werden die Messdaten unterschiedlicher Experimentbedingungen für einen bestimmten Zeitpunkt stets untereinander und nicht versetzt dargestellt. Die einzelnen Rechtecke innerhalb des Slots werden dann mithilfe eines Farbcodes auf Basis des Sample-Durchschnittwerts für die jeweilige Kombination aus Zeitpunkt und Experimentbedingung eingefärbt. Die Variabilität (Standardabweichung) von Replikaten ist mit dieser Visualisierungstechnik nicht darstellbar.

Dadurch, dass häufig eine größere Zahl von Genen mit der Expression eines Proteins oder Enzyms in Zusammenhang stehen, erhöht sich entsprechend die Anzahl der benötigten Slots zur Darstellung der Daten. Entsprechend verkleinert sich die zur Darstellung eines einzelnen *mapping*-Datensatzes verfügbare Fläche. Die farbcodierte Einfärbung der Zeichenfläche ist daher besonders gut geeignet, um umfangreiche Experimentdaten kompakt direkt im Kontext der Visualisierung eines Mappinggraphen darzustellen.

3.2.3 Diagrammdarstellungen

Ein Experimentdatensatz, der aus einem *mapping*-Objekt mit zugeordneten Experimentbedingungen und Zeitpunkten besteht, kann detailliert mithilfe verschiedener Diagrammtypen (Balken-, Linien- oder Kreisdiagramme) dargestellt werden. Sind einem Graphelement mehrere *mapping*-Objekte zugeordnet, dann werden, wie in

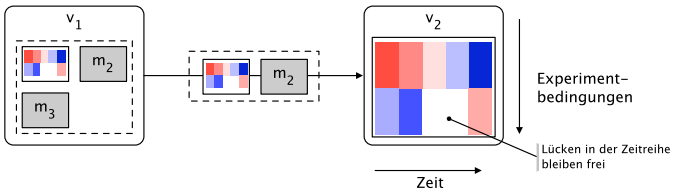

Abbildung 3.8: Beispiel für eine farbcodierte Darstellung der Sample-Durchschnittswerte des *mapping*-Objekts m_1 (siehe Abbildung 3.7). Der dargestellte Datensatz enthält zwei verschiedene Experimentbedingungen und Daten für fünf verschiedene Zeitpunkte.

Abschnitt 3.2.1 erläutert, mehrere Slots zur Darstellung genutzt. Innerhalb einer Knotendarstellung oder auf einer Kante werden dann mehrere Diagramme nebeneinander und gegebenenfalls übereinander angeordnet.

Eine Diagrammdarstellung erlaubt im Gegensatz zur Farbcodierung eine Visualisierung fast aller einem *mapping*-Objekt zugeordneten Experimentdaten-Details (genaue Messwerte, Variabilität der Replikate, Zeitpunkte, Einheiten der Messwerte und Zeitpunktangaben, siehe Abbildung 3.9). Die Vielzahl von Details erhöht aber auch den Flächenbedarf des Slots zur Darstellung des Diagramms. Diagrammdarstellungen können somit nur dann zur netzwerkintegrierten Datenvisualisierung eingesetzt werden, wenn entweder nur wenige Experimentdatensätze einzelnen Netzwerkelementen zugewiesen sind (für Abbildung 3.9 gilt: $n_m(v_4) = n_m(v_5) = 1$) oder genügend Platz zum Beispiel durch eine relativ große Knotendarstellung vorhanden ist. Die Unterscheidung unterschiedlicher Experimentbedingungen wird durch eine Farbcodierung ermöglicht oder kann bei Balkendiagrammen alternativ aus der Reihenfolge der Balken abgeleitet werden. Zeitreihen werden im Liniendiagramm durch einen in x-Richtung fortschreitenden Linienzug repräsentiert, bei Balkendiagrammen werden die Balken ebenfalls in x-Richtung fortschreitend nebeneinander platziert. Bei Kreisdiagrammdarstellung wird für jeden Zeitpunkt ein einzelnes Kreisdiagramm gezeichnet und innerhalb eines Slots nebeneinander platziert. Jedes einzelne Kreisdiagramm zeigt dann das Verhältnis zwischen den für den jeweiligen Zeitpunkt relevanten Experimentbedingungen. Die Absolutwerte sind aus dieser Darstellungsform nicht erkennbar. Sowohl bei Balken- als auch bei Liniendiagrammen können Fehlerbalken dazu genutzt werden, die Sample-Standardabweichung und somit die Variabilität unterschiedlicher Replikate zu visualisieren. Fehlende Sample-Daten führen bei Balken- und Kreisdiagrammen zu Lücken in der Darstellung. Bei Liniendiagrammen ist der Linienzug je nach Anwenderpräferenz durchgezogen oder unterbrochen.

In der Biologie werden häufig unterschiedliche Entwicklungsstadien eines Organismus innerhalb eines bestimmten Zeitraums untersucht. Durch unterschiedliche Einfärbung der Diagrammhintergrundfläche können solche Zeitabschnitte auf einfache Art und Weise gekennzeichnet werden. Auf Diagrammdetails wie die Beschriftung der Achsen kann dann teilweise verzichtet werden.

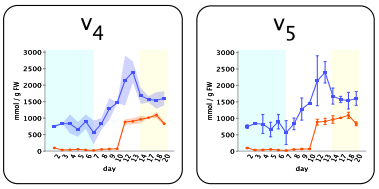

Abbildung 3.9: Unterschiedliche Diagrammdarstellungen eines Experimentdatensatzes. Links Liniendiagramm mit „Linienschatten“ zur Darstellung der Sample-Standardabweichung, rechts Fehlerbalken im Liniendiagramm.

3.3 Graphlayout, Interaktions- und Navigationstechniken

Zur Visualisierung von Graphen werden Layoutalgorithmen benötigt, welche die Knotenpositionen für die Darstellung bestimmen. In Kapitel 2.3.2 wurden eine Reihe von Standardlayoutverfahren und spezialisierte Layouts für biochemische Reaktionsnetze vorgestellt. Je nach Art des betrachteten Mappinggraphen und Ziel der Visualisierung können unterschiedliche Layoutverfahren genutzt werden, um ansprechende Layouts zu berechnen (siehe Abschnitt „Ästhetische Kriterien“, Kapitel 2.3.2). Im nachfolgenden Abschnitt werden den in Kapitel 3.1.2 beispielhaft vorgestellten Arten von Mappinggraphen geeignete Layoutalgorithmen zugeordnet. Weiterhin ist es, wie in Abschnitt 2.3.2 erläutert, sinnvoll, zur Analyse und Exploration von Experimentdaten dynamische Visualisierungen einzusetzen. Der mehrfache Aufwand zur manuellen Erzeugung von Visualisierungen wäre sonst viel zu groß, um bisher unbekannte Zusammenhänge mithilfe unterschiedlicher Darstellungen entdecken und analysieren zu können. In Abschnitt 3.3.2 werden daher für unterschiedliche Mappinggraphtypen geeignete Interaktionsformen vorgestellt.

Mappinggraphen der Typen MG_{CR} und MG_{KEGG} repräsentieren im Allgemeinen nur einen kleinen Ausschnitt aus der Gesamtheit der bekannten biochemischen und genetischen Zusammenhänge. Beide Typen von Graphen können daher Knoten des Typs *Pathwayverweis* enthalten, um auf andere, ausgewählte relevante Netze zu verweisen. Auf statische Visualisierung basierende Informationssysteme (zum Beispiel die webbasierte Schnittstelle zur KEGG-Pathway-Datenbank) unterstützen ausschließlich einen auf Hyperlinks basierenden Sprung von Pathway zu Pathway und somit keine Integration verschiedener relevanter Pathways in eine einzige Darstellung. Die mappinggraphbasierte Repräsentation von Pathways und anderen biologischen Netzen ermöglicht weitergehende, in Kapitel 3.3.3 vorgestellte Techniken der Pathwaynavigation und -exploration.

3.3.1 Layout biologischer Netzwerke und Klassifikationshierarchien

In den folgenden Unterabschnitten werden mit Ausnahme der erweiterten Pathwayübersichtsgraphen (MG_{OE}) allen bisher betrachteten Typen von Mappinggraphen

geeignete Layoutverfahren zugeordnet. Mappinggraphen des Typs MG_{OE} entstehen durch Nutzerinteraktion zur dynamischen Exploration des Datenbestands. Die Vorgehensweise zur Erzeugung und zum ansprechenden Layout der interaktiv bearbeiteten Graphen wird in Kapitel 3.3.3 besprochen.

PPI-Netzwerke und Pathwayübersichtsgraphen

Mappinggraphen der Typen MG_{PPI} und MG_O besitzen keine gerichteten Kanten beziehungsweise keine vorherrschende Kantenrichtung. Auf bestimmte Topologien spezialisierte Layoutverfahren können hier nicht sinnvoll verwendet werden, da entsprechende Graphen normalerweise keine speziellen Struktureigenschaften aufweisen. Ansprechende Darstellungen werden für diese Typen von Mappinggraphen insbesondere durch kräftebasierte Layoutverfahren ermöglicht (siehe 2.3.2). Die berechneten Darstellungen verfügen über eine vergleichsweise gleichmäßige Knotenverteilung und somit über kompakte Abmessungen. Die generelle Struktur des Graphen wird deutlich erkennbar.

Biochemische Reaktionsnetzwerke

Zum Layout biochemischer Reaktionsnetzwerke gibt es bereits eine Reihe von Methoden, welche in Kapitel 2.3.2 vorgestellt wurden. Sind für bestimmte Anwendungssysteme solch spezialisierte Layoutverfahren verfügbar, so liefern diese im Allgemeinen für Mappinggraphen des Typs MG_{CR} wesentlich bessere Ergebnisse als Standardlayoutverfahren. Problematisch ist, dass die vorgestellten Layoutverfahren größtenteils datenbank- oder anwendungsspezifisch entwickelt worden sind. Ist für einen konkreten Anwendungsfall kein spezialisiertes Layout verfügbar, so ist es sinnvoll, beispielsweise hierarchische oder kräftebasierende Standardlayoutverfahren als Startpunkt für eine weitere manuelle Verfeinerung des Layouts zu verwenden.

KEGG-Pathways

Für Mappinggraphen des Typs MG_{KEGG} ist ein manuelles Layout bereits gegeben (die von KEGG zum Download bereitgehaltenen Pathwaydateien enthalten bereits Layoutinformationen). Ein Vorteil des vorhandenen Layouts ist, dass der Experte, der das Layout bearbeitet hat, die aus seiner Sicht relevanten ästhetischen Kriterien direkt bei der Erstellung der Visualisierung berücksichtigen konnte und das Layout dadurch oft besonders ansprechend gestaltet ist.

Werden einem Mappinggraph MG_{KEGG} Experimentdaten zugeordnet und diese Daten mithilfe von vergrößerten Knoten und beispielsweise eingebetteten Diagrammdarstellungen visualisiert, so kann eine Layoutverfeinerung zur automatisierten Verbesserung der Visualisierung eingesetzt werden. Ein Ansatz zur automatischen Layoutverfeinerung, welcher sich auf die Beseitigung von Knotenüberlagerungen unter größtmöglicher Beibehaltung des gegebenen Layouts beschränkt, wurde von T. Dwyer und Koautoren auch in Form einer frei verfügbaren Beispielimplementation in Java und C++ unter der GPL-Lizenz veröffentlicht [97].

Gene-Ontology- und KEGG-BRITE-Hierarchien

Mappinggraphen der Typen MG_{GO} und MG_{BRITE} können automatisch auf ansprechende und übersichtliche Art und Weise mithilfe ebenenweiser Layoutverfahren, zum Beispiel mit dem in Kapitel 2.3.2 vorgestellten Sugiyama-Layout visualisiert werden. Die hier betrachteten Mappinggraphen besitzen keine Zyklen. Schritt eins des Sugiyama-Algorithmus kann daher übersprungen werden. Beim Typ MG_{GO} wird den *Klassifikations*-Knoten mit $l(v) \in \{$„biological process“, „molecular function“, „cellular component“$\}$, also den Quellknoten der Hierarchie, die Layoutebene eins zugeordnet. Für MG_{BRITE} wird den *Klassifikations*-Knoten ohne eingehende Kanten die Layoutebene eins zugeordnet. Um bei den auf Basis eines konkreten Datensatzes konstruierten Mappinggraphen die Knoten der Typen *Klassifikation* und *Pathwayverweis* deutlich von den Graphknoten mit zugeordneten Experimentdaten abzusetzen, werden Letztere einheitlich der untersten Layoutebene zugeordnet.

3.3.2 Interaktionstechniken

In Kapitel 2.3.2 wurden die wichtigsten Interaktionstechniken dynamischer Visualisierungssysteme vorgestellt. Die dort vorgestellten Techniken können auch zur Interaktion mit Visualisierungssystemen für Mappinggraphen sinnvoll eingesetzt werden. Dazu werden die Interaktionstechniken, wie im Folgenden beschrieben, an den Anwendungskontext angepasst.

Nutzerinteraktion auf Ebene der Datentransformation

Dynamic Queries Hierunter versteht man die Spezifikation von Selektionsbedingungen (Graphelementattribute wie Knotengröße, Position, Farbe, Experimentdatenzuordnung, Label) mithilfe einer grafischen Benutzeroberfläche. Diese Technik ist beispielsweise dazu geeignet, nicht relevante Knoten oder Kanten des Mappinggraphen zu selektieren, um diese entfernen zu können.

Direct Walk *Pathwayverweis*-Knoten werden dazu genutzt, die Visualisierung des aktuellen Mappinggraphen zu verlassen und die Visualisierung des dem Pathwayverweis zugehörigen Mappinggraphen aufzurufen.

Attribute Walk Ausgehend vom aktuell selektierten Knoten oder der selektierten Kante wird die Selektion erweitert. Der Nutzer wählt ein bestimmtes Attribut (zum Beispiel Knotentyp, Experimentdatenzuordnung), ausgehend davon werden alle anderen Elemente mit gleichem Attributwert zur Selektion hinzugefügt.

Detail-on-Demand Es wird dem Benutzer ermöglicht, bestimmte Details aus der Darstellung temporär oder dauerhaft zu entfernen. Selektierte Knoten, Kanten oder nicht relevante Teile der Experimentdaten (zum Beispiel bestimmte Zeitpunkte und Experimentbedingungen) können ausgeblendet werden. Umgekehrt ist es auch möglich, Details zur Darstellung auf Nutzeranforderung hinzuzufügen (zum Beispiel Achsenbeschriftung, Darstellung der Standardabweichung).

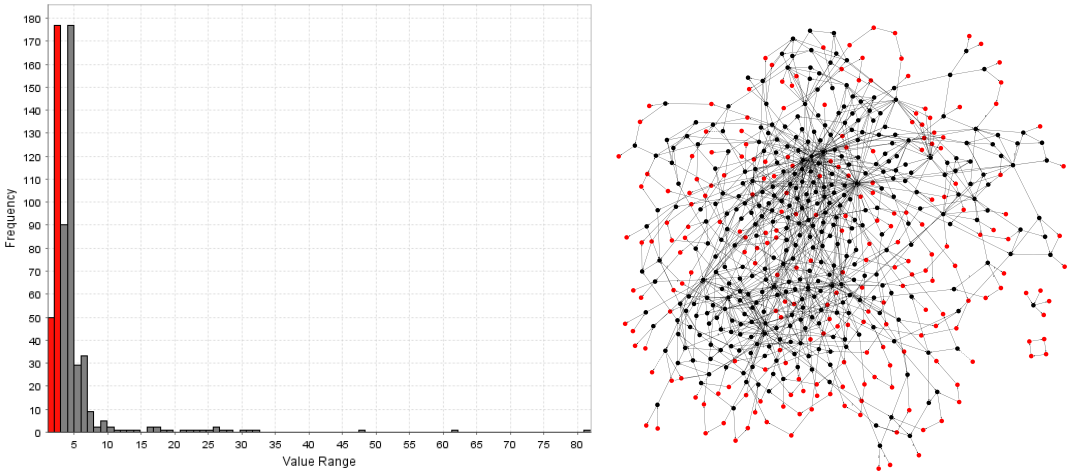

Abbildung 3.10: *Brushing*: Synchronisierte Auswahl von Graphelementen. In der Histogrammdarstellung, welche hier die Knotengradverteilung zeigt (x-Achse: Knotengrad, y-Achse: Anzahl Knoten mit entsprechendem Grad), können durch Mausklick auf die Diagrammbalken Knoten selektiert werden. Die Selektion wird in allen Graphansichten automatisch nachvollzogen (rot markierte Knoten und Diagrammbalken). Biochemisches Reaktionsnetz aus [98], Visualisierung mit VANTED.

Brushing Der Graph wird gleichzeitig in verschiedenen Ansichten dargestellt. Eine Ansicht zeigt beispielsweise die Graphstruktur, ein zweiter in einem Histogramm die Verteilung eines ausgewählten Knoten- oder Kantenattributs. Dann können Graphelemente in einer Ansicht auf Basis der Knotenattributausprägung selektiert werden. Die Auswahl wird in allen anderen Ansichten nachvollzogen (Beispiel siehe Abbildung 3.10).

Direct Manipulation Positionen und Größen der Knoten werden mithilfe von Tastaturkurzbefehlen direkt manipuliert. Die Knoten- und Kantenlabel ausgewählter Elemente können mit der Tastatur ohne weitere Zwischenschritte direkt editiert werden.

Nutzerinteraktion zur Anpassung der Zuordnung von Mappinggraph-Daten zu Attributen der Visualisierung

Color-Coding Eine grafische Benutzeroberfläche erlaubt die Änderung der Parameter einer Transformationsfunktion. Diese bestimmt den zum darzustellenden Datenpunkt zugehörigen Farbwert zur Knotenfüllung oder Kantenfärbung.

Shape-Coding Es wird eine Benutzeroberfläche bereitgestellt, welche es dem Nutzer ermöglicht, die Parameter der Transformationsfunktion zur Zuordnung von Knotenform und Größe sowie die Art der Kantendarstellung festzulegen.

Diagrammdarstellung Mithilfe einer Benutzeroberfläche können die verschiedenen in diesem Kapitel vorgestellten Diagrammdarstellungen zur Visualisierung der Experimentdaten ausgewählt werden. Bestimmte Charakteristika wie Anzeige der Achsenbeschriftungen, verwendete Farben, Anzeige der Fehlerbalken usw. können vom Nutzer spezifiziert werden.

Nutzerinteraktion auf Darstellungsebene

Direct Selection In der grafischen Darstellung des Mappinggraphen können Knoten und Kanten mit der Maus durch Anklicken oder durch Aufspannen eines Selektionsrechtecks direkt ausgewählt werden. Nachfolgende Nutzerkommandos beziehen sich soweit wie möglich auf die aktuelle Selektion.

Overview-and-Detail Ein Mappinggraph kann in mehreren voneinander getrennten Ansichten gleichzeitig dargestellt werden. Eine Darstellung zeigt den gesamten Graphen, eine weitere zeigt einen Teil des Graphen. In der Übersichtsdarstellung wird der in der Detailansicht dargestellte Ausschnitt gekennzeichnet. Diese Interaktionsform basiert zum Teil auf der *Zooming*-Technik, welche es dem Nutzer erlaubt, den Mappinggraphen mit einem vom Nutzer vorgegebenen Skalierungsfaktor darzustellen. Ein vom Nutzer veranlasster Wechsel des Skalierungsfaktors kann neben den Veränderungen auf Darstellungsebene weitergehende Anpassungen der Darstellung auf Ebene der Datentransformation (Details-on-Demand) verursachen und auf der Ebene der Zuordnung von Daten zu Attributen der Visualisierung (z. B. Anpassung der Diagrammdarstellung) auslösen.

3.3.3 Pathwaynavigation und -integration

Da ein biologisches Netzwerk oft nur einen Teilausschnitt aus der Gesamtheit der bekannten, in Graphmodellen abgebildeten Zusammenhänge darstellt, ist es sinnvoll, Knoten des Typs *Pathwayverweis* dazu zu nutzen, um auf andere relevante Mappinggraphen zu verweisen. Von den vorgestellten konkretisierten Typen von Mappinggraphen enthalten MG_O, MG_{OE}, MG_{CR}, MG_{KEGG} und MG_{BRITE} Knoten dieses Typs. Mappinggraphen, welche kein biologisches Netzwerk, sondern eine Klassifikationshierarche repräsentieren (MG_{BRITE}) sollen hier jedoch nicht betrachtet werden. Für diese ist kein Pathwayübersichtsgraph (MG_O bzw. MG_{OE}) definiert.

Knoten des Typs *Pathwayverweis* können unter Verwendung der *Direct-Walk*-Technik dazu genutzt werden, um interaktiv zwischen verschiedenen Mappinggraphvisualisierungen zu wechseln. Sind *Pathwayverweis*-Knoten mit anderen Knoten verbunden, geben sie außerdem einen sichtbaren Hinweis darauf, dass bestimmte Reaktionspartner, allgemein bestimmte Elemente des Mappinggraphen, ebenfalls in anderen Mappinggraphen vorkommen. Ähnlich wie beim Navigieren zwischen verschiedenen verlinkten Webseiten, kann auch die Struktur der Verlinkung unterschiedlicher Mappinggraphen unübersichtlich sein. Außerdem ist es manchmal sinnvoll, die Informationen mehrerer Mappinggraphen in ein Graphmodell zu integrieren. Aus diesen Gründen werden hier vier über die *Direct-Walk*-Technik hinausgehende Navigations- und Explorationstechniken entwickelt und vorgestellt. Die folgende

Auflistung gibt einen Überblick:

(1) Die Navigation startet mit einem Pathwayübersichtsgraphen und somit einer schematischen Übersichtsdarstellung der Verlinkung der mithilfe von *Pathwayverweis*-Knoten repräsentierten Mappinggraphen. Der Übersichtsgraph dient als Basis für die Erweiterung der Darstellung, indem einzelne *Pathwayverweis*-Knoten entfernt und Elemente des dazugehörigen Mappinggraphen hinzugefügt werden.

(2) Startpunkt der Navigation kann ein bestimmtes biologisches Netzwerk sein. Die Darstellung wird auf Nutzeranforderung hin erweitert, indem wie im ersten Fall *Pathwayverweis*-Knoten durch die Elemente des referenzierten Mappinggraphen ersetzt werden. Dieser Prozess kann mehrfach in Gang gesetzt werden, sodass je nach Auswahl der Pathwayverweise vielfältige Strukturen von verlinkten, integrierten Mappinggraphen (Sternstruktur, Kettenstruktur) entstehen können.

(3) Sind die Elemente mehrerer Mappinggraphen in eine Darstellung integriert, können diese neu platziert werden. Beispielsweise können die integrierten Netze kreisförmig oder direkt nebeneinander angeordnet werden.

(4) Nach der Betrachtung der Details eines integrierten Mappinggraphen können die Elemente entfernt und durch einen *Pathwayverweis*-Knoten ersetzt werden.

Diese Navigations- und Explorationstechniken werden im Folgenden detailliert beschrieben.

(1) Erweiterung der Pathwayübersicht

Eine erste Variante der visuellen Pathwaynavigation entspricht dem Ansatz „von der Übersicht ins Detail“. Ausgangspunkt ist der Pathwayübersichtsgraph MG_O.

Um die Graphdarstellung übersichtlich und verständlich zu halten, soll sichergestellt werden, dass ein bestimmter Mappinggraph nur einmal in die Darstellung integriert werden kann. Dies wird dadurch erreicht, dass nach jeder Integration eines Mappinggraphen *Pathwayverweis*-Knoten mit identischem Label, welche auf ein- und denselben Mappinggraphen verweisen, fusioniert werden.

Die interaktive Pathwayintegration verläuft dann im Detail wie folgt: Der Nutzer selektiert einen interessierenden *Pathwayverweis*-Knoten $v_i \in MG_O$ und startet die Integration des zugehörigen Mappinggraphen MG_i, beispielsweise mit einem Doppel-Klick oder durch Auswahl eines Menübefehls. Die Höhe und Breite des gelayouteten Mappinggraph MG_i wird ermittelt und mit der Größe des Knotens v_i verglichen. Ist die Größe des Knotens v_i hinreichend, wird mit der Integration von MG_i direkt fortgefahren. Ist dies nicht der Fall, wird die Größe des Knotens v_i so angepasst, dass diese den Abmessungen des Layouts von MG_i entspricht. Der weitere Integrationsvorgang wird nun unterbrochen. Der Nutzer hat nun die Möglichkeit, das Layout so anzupassen, dass es nach der Pathwayintegration zu keinen Überschneidungen kommt. Dies wird dadurch erleichtert, dass die vergrößerte

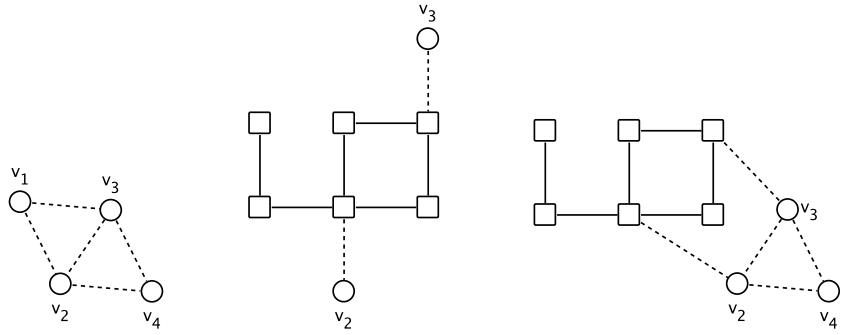

Abbildung 3.11: Erweiterung der Pathwayübersicht. Startpunkt der Navigation ist die Auswahl des *Pathwayverweis*-Knotens v_1 im Pathwayübersichtsgraph MG_O (links). Der Mappinggraph MG_1 (mittig) wird in MG_O integriert, v_1 wird entfernt. Durch Fusion von *Pathwayverweis*-Knoten mit identischem Label in MG_{OE} entstehen Kantenverbindungen zwischen den beiden Ursprungsgraphen. Es entsteht der erweiterte Pathwayübersichtsgraph MG_{OE} (rechts).

Knotendarstellung von v_i eine visuelle Vorschau auf das spätere Resultat der Integration gibt. Bei ausreichender Knotengröße von v_i verläuft die Erweiterung von MG_O um MG_i dann wie folgt (siehe auch Abbildung 3.11):

1. MG_O wird mit MG_i vereint: $MG_{OE} = MG_O \cup MG_i$. Dabei werden sämtliche in MG_i und MG_O enthaltenen Elemente Teil von MG_{OE}. Beispielsweise sind *Pathwayverweis*-Knoten aus beiden Graphen, die das gleiche Label haben, nach diesem Schritt unverändert in MG_{OE} enthalten.

2. Das Layout der Graphelemente aus MG_i wird in MG_{OE} so angepasst, dass die neu integrierten Graphelemente ihre ursprüngliche relative Positionierung beibehalten, aber sämtlich innerhalb der Grenzen der Knotendarstellung von v_i liegen.

3. Der Knoten v_i wird aus MG_{OE} entfernt.

4. Es erfolgt eine Knotenfusion der *Pathwayverweis*-Knoten mit gleichem Label. Die Position der fusionierten Knoten wird so festgelegt, dass diese der Position der Verweisknoten mit demselben Label im Ursprungsgraphen MG_O entspricht.

Der neu entstandene Graph MG_{OE} kann durch das nachfolgend beschriebene Verfahren um weitere Mappinggraphen erweitert werden. Die Graphvisualisierung kann alternativ wieder dadurch vereinfacht werden, dass der integrierte Pathway „zugeklappt“ und wieder durch einen *Pathwayverweis*-Knoten repräsentiert wird.

(2) Schrittweise Erweiterung der Pathwaydarstellung

Die zweite Variante zur visuellen Pathwayexploration startet mit der Darstellung eines gegebenen Mappinggraphen MG_i, welcher schrittweise zur Darstellung weiterer

Pathways erweitert wird. Betrachtet werden Mappinggraphen der Typen MG_{CR}, MG_{KEGG} und MG_{OE}.

Die interaktive Pathwayintegration verläuft dann im Detail wie folgt ab: Der Nutzer selektiert einen interessierenden *Pathwayverweis*-Knoten v_j im Mappinggraph MG_i und startet die Integration wie beim vorherigen Navigationsverfahren mit einem Doppel-Klick oder mithilfe eines Menübefehls. Wie im vorherigen Fall wird die Größe der zu v_j zugehörigen Pathway-Darstellung von MG_j ermittelt und mit der Größe des Knotens v_j verglichen. Ist v_j nicht hinreichend groß, wird die Größe von v_j angepasst, ansonsten wird mit der Integration wie folgt fortgefahren (vgl. auch Abb. 3.12):

1. MG_i wird mit MG_j vereint: $MG'_i = MG_i \cup MG_j$, dabei werden sämtliche in MG_i und MG_j enthaltenen Elemente in MG'_i einbezogen.

2. Das Layout der Graphelemente aus MG_j wird in MG'_i so angepasst, dass die neu integrierten Graphelemente ihre ursprüngliche relative Positionierung beibehalten, aber sämtlich innerhalb der Grenzen der Knotendarstellung von v_j liegen.

3. Aus MG_i und MG_j übernommene Knoten, welche mit v_j oder v_i verbunden sind, werden, falls vorhanden, mit einem aus dem jeweils anderen Pathway übernommenen Knoten, welcher dasselbe Label aufweist, verbunden. Der neu angelegten Kante wird der Typ *Link* zugewiesen. Falls es mehrere mögliche Knoten als Ziel der Kante gibt, wird der Knoten mit der geringsten Entfernung ausgewählt, um die Übersicht zu verbessern.

4. Der *Pathwayverweis*-Knoten v_j wird vom Graphen MG'_i entfernt. Falls vorhanden, werden auf MG_i verweisende *Pathwayverweis*-Knoten v_i aus MG'_i entfernt. Anschließend erfolgt eine Knotenfusion aller *Pathwayverweis*-Knoten mit demselben Label. Die Position der fusionierten Knoten entspricht der Position der entsprechenden Verweisknoten im Ursprungsgraphen MG_i.

Elemente mit übereinstimmendem Label, die nicht vom Typ *Pathwayverweis* sind, werden nicht fusioniert und verbleiben mehrfach in der integrierten Darstellung. Dies ist jedoch für den Anwender keine ungewöhnliche Situation, da bereits einzelne Mappinggraphen mehrfach Elemente mit gleichem Label enthalten können. Der Vorteil dieser Vorgehensweise ist, dass das relative Layout der Knoten der integrierten Mappinggraphen erhalten bleibt und somit eine *Mental-Map-Preserving*-Darstellung [99] erreicht wird. Für Analysezwecke (z. B. Ermittlung der Gradverteilung) kann es aber sinnvoll sein, mehrfach auftretende Knoten eines bestimmten Typs (z. B. *Metabolit*) zu fusionieren.

(3) Anordnung von Pathways

Ein wichtiger Aspekt bei der interaktiven Darstellung verschiedener integrierter Mappinggraphen ist ein (semi-)automatisches Layout der integrierten Darstellung.

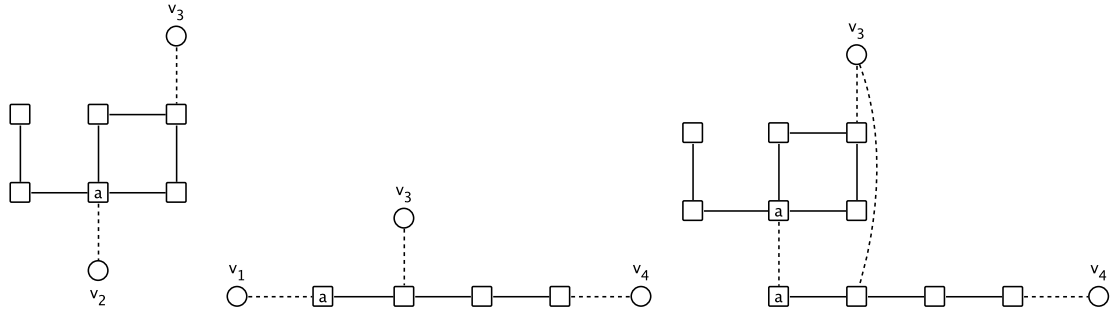

Abbildung 3.12: Erweiterung eines Pathwaygraphen: Mappinggraph MG_1 (links) wird mit den Elementen des Graphen MG_2 (mittig) erweitert, es entsteht MG_1' (rechts).

Mit zunehmender Zahl der integrierten Graphen nimmt die Anzahl der Netzwerkelemente zu. Ein manuelles Layout ist dann sehr aufwendig und somit unzureichend. Ein automatisches Layout der Gesamtdarstellung berücksichtigt nicht das gegebene, oft manuell von Experten optimierte Layout der einzelnen integrierten Mappinggraphen. Ein automatisches Layout wird daher im Allgemeinen zu einer unübersichtlicheren Darstellung führen, außerdem würde die *Mental Map* zerstört.

Eine mögliche Lösung für dieses Problem ist, das zum Beispiel für MG_{KEGG} gegebene Layout der integrierten Mappinggraphen zu erhalten, aber die relative Anordnung der integrierten Graphen manuell oder automatisiert vorzunehmen, um so eine verbesserte Gesamtdarstellung zu erhalten. Zu Beginn wird ein für die aktuelle Visualisierung eines Mappinggraphen MG relevanter Pathwayübersichtsgraph MG_O erzeugt. Für jeden in MG integrierten Mappinggraph MG_i wird ein *Pathwayverweis*-Knoten v_i in MG_O erzeugt. Kanten des Typs *Link* verbinden zwei Knoten in MG_O, wenn mindestens ein Verweis zwischen den zugehörigen Pathwaygraphen existiert. Die Größe der Knoten in MG_O wird so gewählt, dass diese der Größe der zugehörigen Pathwaydarstellung entsprechen. Der Pathwayübersichtsgraph MG_O kann dann automatisiert zum Beispiel mit einem kräftebasierten Verfahren gelayoutet werden. Je nach Struktur der Verbindungen in MG_O führt oft auch das Kreislayout zu guten Ergebnissen. Enthält MG_O nur wenige Knoten, kann auch ein manuelles Layout sinnvoll eingesetzt werden. Im letzten Schritt wird das Layout des Pathwayübersichtsgraphen genutzt, um die in MG integrierten Knoten neu anzuordnen. Die Knoten aus MG werden so verschoben, dass der Durchschnitt der Knotenpositionen der zu MG_i zugehörigen Knoten der Position des Knotens v_i in MG_O entspricht.

(4) „Zuklappen“ von Pathways

Zur weiteren Unterstützung einer interaktiven Navigation zwischen verschiedenen Pathways ist es nützlich, wenn die Elemente eines integrierten Mappinggraphen nach der detaillierten Darstellung und Betrachtung wieder durch einen dazugehörigen *Pathwayverweis*-Knoten ersetzt werden können. Eine solche Operation wird wie folgt durchgeführt: Alle in einen Mappinggraph MG integrierten Elemente eines aus-

gewählten Pathways (MG_i) werden in einen Knoten v_i vom Typ *Pathwayverweis* fusioniert, entstehende Schlingen werden anschließend entfernt. v_i wird auf die dem Mittel der Positionen der fusionierten Knoten entsprechende Stelle platziert. Die Größe von v_i wird so festgelegt, dass der Pathwaytitel als Label im fusionierten Knoten dargestellt werden kann.

3.4 Netzwerkintegrierte Analyse von Experimentdaten

Experimentdaten, die den Elementen des Mappinggraphen zugeordnet sind, können nicht nur zur netzwerkintegrierten Visualisierung genutzt werden, die Zuordnung ermöglicht außerdem statistische Analysen. Einerseits sind dabei „klassische“ Analysen, die ausschließlich Experimentdaten betrachten, sinnvoll durchführbar. Die Zuordnung der Daten zum Mappinggraphen kann dazu genutzt werden, die Ergebnisse der Analysen anschaulich direkt im Netzwerkkontext zu visualisieren. Beispielsweise können die Ergebnisse eines t-Tests mithilfe von Color- oder Shape-Coding sowie mithilfe von grafischen Symbolen innerhalb von Diagrammen dargestellt werden. Andererseits kann die Graphstruktur des Mappinggraphen die Basis für weitergehende Analyseverfahren bilden, indem gleichzeitig Experimentdaten und Netzwerkstruktur berücksichtigt werden.

Im Folgenden werden drei Methoden zur netzwerkintegrierten Analyse und Visualisierung der Ergebnisse vorgestellt. Der erste Unterabschnitt stellt Methoden zur (interaktiven) Korrelationsberechnung vor. Im zweiten Unterabschnitt wird die datenspezifische Gruppenzuordnung der Knoten eines Klassifikationsgraphen ausgewertet, es werden Häufigkeitsverteilungen bestimmt und netzwerkintegriert visualisiert. Im dritten Unterabschnitt wird diskutiert, wie diese Häufigkeitsverteilungen auf statistische Signifikanz überprüft und die Ergebnisse visualisiert werden können.

3.4.1 Korrelationsanalysen

Die wichtigsten Grundlagen zur Berechnung von Korrelationen wurden in Kapitel 2.2.3 vorgestellt. Zur Berechnung der Pearson-Korrelation wird vorausgesetzt, dass die zu korrelierenden Variablen annähernd normalverteilt sind, was mithilfe statistischer Tests überprüft werden kann (siehe Kapitel 2.2.2). Ist dies nicht gegeben oder zielt die Analyse auf die Entdeckung nicht-linearer Zusammenhänge ab, sollte statt dessen die Spearman-Rangkorrelation berechnet werden.

Zur Berechnung des Korrelationsfaktors r oder der Rangkorrelation r_s zwischen den beiden Graphelementen ge_1 und ge_2 ($ge_1, ge_2 \in MG$) zugeordneten Experimentdatensätzen müssen Paare von zu korrelierenden Werten gefunden werden ($X = (x_1...x_n)$, $Y = (y_1...y_n)$). Die Werte $x_1...x_n$ und $y_1...y_n$ werden dabei dem *value*-Attributen der dem Graphelementen ge_1 und ge_2 zugeordneten Objekte des Typs *measurement* entnommen. Die Zuordnung eines Graphelements ge zu *measurement*-Objekten basiert im ersten Schritt auf der Datenzuordnungsfunktion $z(ge)$. Jedes *measurement*-Objekt ist genau einem *sample*-, einem *condition*- und einem

mapping-Objekt zugeordnet. Details hierzu finden sich in Kapitel 3.1.3. Die *value*-Attributwerte zweier *measurement*-Objekte bilden genau dann ein Paar, wenn die folgenden Attribute der zugeordneten Objekte übereinstimmen (in Klammern Objekttyp): (mapping:) experimentName, (condition:) species, genotype, treatment, (sample:) time, timeUnit, (measurement:) replicateID. Im Folgenden wird von der „Korrelation zweier Graphelemente“ gesprochen, wobei die Korrelation der zugeordneten Experimentdaten gemeint ist. In Abschnitt 2.2.3 wurde gezeigt, wie der berechnete Korrelationsfaktor auf statistische Signifikanz geprüft werden kann. Das Ergebnis der entsprechenden Berechnung ist der Wert α, welcher mit einem vom Nutzer spezifizierten Grenzwert verglichen wird (z. B. $\alpha \leq 0.05$).

Hinsichtlich Auswahl der zu korrelierenden Graphelemente und Visualisierung der Ergebnisse sollen drei Varianten vorgestellt werden (siehe Abbildung 3.13): (1) Die Korrelationen zwischen einem ausgewählten Graphelement und allen anderen Graphelementen („1:n“). (2) Die Korrelation von benachbarten Knoten. (3) Die Korrelation aller möglichen Kombinationen von Knoten („n:n“). Nur die erste Variante (1:n) ist zur Analyse von Experimentdaten geeignet, welche Knoten oder Kanten zugeordnet sind, Varianten 2 und 3 sind auf die Analyse der Korrelation von Knoten beschränkt. In allen drei Modi wird zur Unterstützung einer interaktiven Datenanalyse die jeweils vorher aktive Visualisierungseinstellung gespeichert (Farbe, Größe, Stärke der Knoten- und Kantendarstellung und im n:n-Fall zusätzlich die Graphstruktur). Dadurch wird dem Nutzer ermöglicht, jederzeit die ursprüngliche Darstellung wiederherzustellen, zum Beispiel um mithilfe einer anderen Analysemethode Einsicht in die Daten zu bekommen.

1:n-Korrelationsanalyse Die 1:n-Analyse dient dazu, die Korrelation eines ausgewählten Graphelements zu allen verbleibenden Graphelementen zu berechnen und darzustellen. Die Ergebnisse der Korrelations- und Signifikanztests werden mittels Color- oder Shape-Coding visualisiert. Der Nutzer selektiert das gewünschte Graphelement und passt die Parameter dieser Methode den Erfordernissen an. Für die Farbocodierung wird eine Funktion parametrisiert, indem die den Korrelationsfaktoren -1, 0 und +1 zugeordneten Farben vom Nutzer spezifiziert werden. Im Falle des Shape-Codings werden Transformationsfunktionen zum Beispiel für Shape-Größe oder Schranken zum Wechsel der Knotenform (unterschiedliche Knotenformen für positiv oder negativ korrelierte Knoten) spezifiziert. Die Signifikanzschranke und der Visualisierungsmodus werden vom Nutzer spezifiziert (zum Beispiel dickere Umrandung der signifikanten Knoten oder breitere Linienzüge zur Darstellung signifikanter Kanten).

Korrelationen benachbarter Knoten Für diese Variante wird die Korrelation aller Paare benachbarter Knoten berechnet und die Ergebnisse durch Anpassung der Darstellung der verbindenden Kanten visualisiert. Der Nutzer wählt wie im 1:n-Fall die Parameter der zur Visualisierung der Ergebnisse verwendeten Farbzuordnungsfunktion. Die Kanten werden entsprechend dem Korrelationsfaktor (basierend auf den zugeordneten Daten für Anfangs- und Endknoten) eingefärbt. Kanten, die zwei

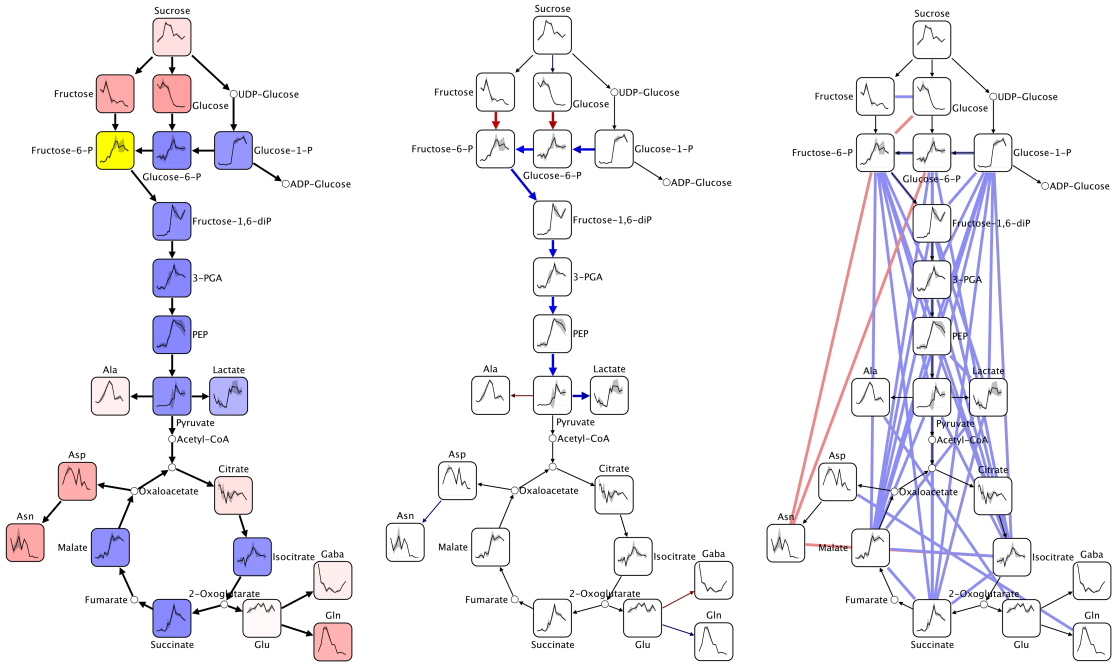

Abbildung 3.13: Interaktive Korrelationsberechnung und netzwerkintegrierte Visualisierung der Ergebnisse mittels Farbcodierung (positive Werte blau, negative rot). Dargestellt ist der zeitliche Verlauf (x-Achse) relativer Metabolitkonzentrationen (y-Achse) im Kontext der Glykolyse. Linkes Bild: Modus 1:n: Korrelation eines ausgewählten Graphelements (gelb gefärbter Knoten) zu allen anderen Graphelementen mit zugeordneten Experimentdaten. Mittleres Bild: Korrelation benachbarter Knoten: statistisch signifikante Korrelationen sind durch breiteren Linienzug hervorgehoben. Rechtes Bild: Modus n:n: Statistisch signifikante Beziehungen werden mittels neu angelegter Kanten visualisiert.

statistisch signifikant korrelierte Knoten verbinden, werden optional mittels Shape-Coding (Änderung der Breite des Linienzugs) hervorgehoben.

n:n-Korrelationsanalyse Die Korrelation wird hier wie im vorher beschriebenen Fall der Korrelation verbundener Knoten bestimmt, allerdings für alle möglichen Knotenpaare. Zur Visualisierung der Ergebnisse werden neue Kanten erzeugt. Um sicherzustellen, dass die ursprüngliche Graphstruktur sichtbar bleibt, werden die neu erzeugten Kanten in der Darstellung hinter den vorher vorhandenen Kanten angeordnet und mit größerer Linienstärke gezeichnet. Ohne Signifikanzschranke würde ein sehr unübersichtlicher vollständiger Graph entstehen. Aus diesem Grund werden Kanten nur dann neu angelegt und zur Visualisierung der Korrelation genutzt, wenn die beobachtete Korrelation statistisch signifikant ist (entsprechend der vom Nutzer vorgegebenen Schranke).

3.4.2 Histogrammfunktionen für Klassifikationshierarchien

Grundlage dieser Methode sind datenspezifische Klassifikationsgraphen MG_{GO} oder MG_{BRITE}. Es wird vorausgesetzt, dass durch eine Datenanalysefunktion f jeder Knoten $v \in MG$, für den gilt $t(v) \notin \{Klassifikation,\ Pathwayverweis\}$ und $|z(v)| > 0$, genau einer Gruppe c aus der Gesamtmenge der möglichen Gruppen C zugeordnet werden kann: $f(z(v)) = c$. Die Funktion f dient somit dem Datenclustering und kann beispielsweise in Form einer Self-Organizing Map [100] (kurz „SOM“) implementiert sein. Um nun einen Überblick über die klassenspezifische Gruppenzuordnung zu bekommen, wird für jeden Knoten vom Typ $t(v) \in \{Klassifikation,\ Pathwayverweis\}$ die Häufigkeit der Gruppenzuordnungen der von diesen Knoten aus erreichbaren Blätter bestimmt. Diesen Knoten werden über die Funktion z neu angelegte *mapping*-Datensätze zugeordnet: Den *mapping*-Objekten wird dazu für jede mögliche Gruppe $c \in C$ jeweils ein neu erzeugtes *condition*-Objekt zugeordnet, welches über ein *sample*-Objekt mit einem *measurement*-Objekt verbunden ist. Der Wert des Attributs *value* des *measurement*-Objekts entspricht der Häufigkeit der jeweiligen Gruppenzuordnung. Diese Vorgehensweise ermöglicht es, die Gruppierung der Experimentdaten im Kontext der zugehörigen Klassifikationshierarchie mittels der vorgestellten Visualisierungsmethoden darzustellen. Zum Beispiel können die *mapping*-Datensätze in Form eines Histogramms unter Verwendung eines Balkendiagramms oder die Verhältnisse der Gruppenhäufigkeiten mittels Kreisdiagrammen visualisiert werden. Weiterhin kann das Ergebnis mit dem im nachfolgenden Abschnitt vorgestellten Ansatz auf Signifikanz geprüft werden.

3.4.3 Signifikanzanalyse für Klassifikationshierarchien

Wurde, wie im vorherigen Abschnitt erläutert, ein Klassifikationsgraph konstruiert, so zeigt dieser für jeden Klassifikationsknoten die Häufigkeitsverteilung der Experimentdatengruppenzuordnung. Von Interesse sind dann häufig die Knoten, die eine auffällig unterschiedliche Verteilung im Vergleich zur gesamten Verteilung haben. Dazu wird eine Gruppe c_i ausgewählt und deren Häufigkeit der Zuordnung innerhalb der Klassifikationshierarchie statistisch untersucht. Es wird die Nullhypothe-

se aufgestellt, dass die Häufigkeitsverteilung der Zuordnung der Experimentdaten zur Gruppe c_i unabhängig von der Zuordnung der Experimentdaten zum jeweiligen Klassifikationsknoten v_i ist. Dazu wird für jeden Knoten v_i eine Vierfeldertafel $T = (a, b, c, d)$ erstellt:

		Gruppenzugehörigkeit	
		c_i	nicht c_i
Klassifikation	v_i	a = Anzahl der von v_i erreichbaren Blätter mit zugeordneter Gruppe c_i	b = Anzahl der von v_i erreichbaren und nicht zu c_i gehörenden Blätter
	nicht v_i zugeordnet	c = Anzahl der zur Gruppe c_i gehörenden von v_i aus unerreichbaren Blätter	d = Anzahl der nicht zur Gruppe c_i gehörenden und gleichzeitig von v_i unerreichbaren Blätter

Statistisch abgesichert lässt sich die beschriebene Fragestellung mit dem Exakten Fisher-Test beantworten (siehe Kapitel 2.2.2). Jedem Knoten vom Typ $t(v) \in \{\textit{Klassifikation, Pathwayverweis}\}$ wird mithilfe des Exakten Fisher-Test ein p-Wert zugeordnet, welcher der Summe der Einzelwahrscheinlichkeiten für das rein vom Zufall bestimmte Auftreten der zu berücksichtigenden Vierfeldertafeln entspricht. Je nach Anwendungsfall wird entweder der einseitige oder der zweiseitige Exakte Fisher-Test durchgeführt. Niedrige p-Werte sind dann ein Zeichen dafür, dass es eine stochastische Abhängigkeit zwischen der Gruppenzuordnung und der Zuordnung der Experimentdaten zum Klassifikationsknoten gibt. Liegt der p-Wert unter einem vom Nutzer spezifizierten Schwellwert (z. B. $p \leq 0,05$), dann wird die beobachtete Häufigkeitsverteilung als nicht zufällig und somit signifikant angesehen. Der Graph wird durch Entfernen aller Knoten, von denen aus kein signifikanter Knoten erreichbar ist, vereinfacht. Die verbleibenden signifikanten Knoten werden anschließend durch Color- oder Shape-Coding hervorgehoben dargestellt.

4 Realisierung

Die im vorherigen Kapitel vorgestellte Methodik wurde im Softwaresystem VANTED implementiert. Die Bereitstellung einer öffentlich verfügbaren Implementation ermöglicht neben der praktischen Anwendung der entwickelten Methodik eine erweiterte Methodenvalidierung. Externe Anwender helfen nicht nur dabei, Fehler in der Implementation aufzuspüren, die praktische Anwendung gibt wichtige Hinweise zur Relevanz der entwickelten Methoden und ermöglicht so die zielgerichtete Weiterentwicklung. Dazu wird zu Beginn dieses Kapitels der zur Implementation der Methodik verfolgte Softwareentwicklungsprozess skizziert und im Anschluss die Systemarchitektur der entwickelten Software vorgestellt. Darauf folgt die Vorstellung der wesentlichen Programmfunktionen auf der Basis eines Beispielworkflows und schließlich ein Vergleich des entwickelten Systems mit anderen Datenvisualisierungsprogrammen.

4.1 Der Softwareentwicklungsprozess

Die Wahl eines geeigneten Softwareentwicklungsmodells kann eine Rückkopplung in Gang setzen, welche einerseits zu einem positiven Einfluss auf die anwendergerechte Implementierung führt, andererseits aber auch wichtige Impulse für die Methodenentwicklung geben kann.

Ein traditionell weitverbreitetes Vorgehensmodell zur Softwareentwicklung ist das bereits 1970 von Royce diskutierte „Wasserfallmodell“ [101]. Der Name rührt daher, dass in der grafischen Darstellung die einzelnen Phasen kaskadiert angeordnet sind und im Normalfall Schritt für Schritt vorwärts durchlaufen werden. Werden Fehler erkannt, so wird versucht, diese in der vorangegangenen Phase zu korrigieren. Die Arbeitsschwerpunkte des Wasserfallmodells finden sich direkt oder zumindest in abgewandelter Form in vielen anderen Vorgehensmodellen wieder, da die einzelnen Schritte von grundlegender Bedeutung sind:

1. Anforderungsanalyse: In dieser Phase erfolgt die Anforderungsdefinition und Aufwandsschätzung. Ergebnis sind Lasten- und Pflichtenheft.

2. Systemdesign: Erarbeitung eines Softwaredesigns/-architektur, beispielsweise auf der Basis der Unified Modelling Language (UML).

3. Implementierung: Programmierung beispielsweise entsprechend dem objektorientierten Programmierparadigma (OOP).

4. Integrations- und Systemtest: Validierung und Verifikation.

5. Einsatz und Wartung: Bereitstellung der Software, Schulung, Fehlerkorrekturen, Dokumentation von Erweiterungen.

Zu Beginn des Projektes war es nicht möglich, alle für die Methodenentwicklung und Implementierung relevanten Anforderungen und Wünsche der potenziellen Nutzer detailliert zu dokumentieren. Ein wesentlicher Grund hierfür war, dass es zu diesem Zeitpunkt keine klaren Vorstellungen, sondern eher grob umrissene Entwicklungsziele für geeignete Visualisierungs- und Analysemethoden gab. Es war daher erwünscht, möglichst schnell vorzeigbare Ergebnisse zu erhalten, um davon abgeleitet weitere nützliche Ideen entwickeln zu können und weitere Anwender zu gewinnen. Eine lineare Methoden- und Softwareentwicklung, bei der die Anwender erst am Projektende das Resultat in Augenschein nehmen können, ist in diesem Fall ungeeignet. Der wesentliche Vorteil des stattdessen genutzten evolutionären Entwicklungsmodells besteht darin, dass zu Beginn die Implementation der Anwendung in der vollen Breite durch eine Konzentration auf die eigentlichen Kernanforderungen vermieden wird. Auf diese Weise entsteht eine so genannte „Nullversion“ der Software, welche frühzeitig dem Anwender zur produktiven Arbeit zur Verfügung gestellt wird. Aus der praktischen Anwendung und auch aus der vorangegangenen Anforderungsanalyse für die Nullversion heraus werden Änderungs- und Erweiterungswünsche gesammelt und in einer neuen Version umgesetzt. Ein möglicher Nachteil dieses Vorgehens ist die Inflexibilität der Nullversion gegenüber unvorhergesehenen Anforderungen. Durch Nutzung eines flexiblen Frameworks, in diesem Fall der Plug-in-basierten Entwicklungsplattform Gravisto, welche im nächsten Abschnitt noch näher vorgestellt wird, konnte dieses Problem umgangen werden. Der Aufwand zur zeitnahen Bereitstellung von neuen Softwareversionen wird durch den Einsatz von Techniken zum automatischen Softwareupdate (in diesem Fall Java Web Start) minimiert.

4.2 Systemarchitektur

Die sich aus der Methodik ergebenden Anforderungen hinsichtlich Datenhandling (Laden, Verarbeiten und Speichern von Hochdurchsatzexperimentdaten) und Interaktivität führen dazu, dass die Implementation als Desktopanwendung aus folgenden Gründen sinnvoll ist: webbasierte Systeme waren zu Beginn des Projekts zur Umsetzung der Anforderungen hinsichtlich Interaktivität ungeeignet. Obwohl inzwischen die Technik AJAX (kurz für Asynchronous JavaScript and XML) hochgradig interaktive webbasierte Anwendungen ermöglicht, besteht die Einschränkung hinsichtlich des Datenhandlings fort. Hochdurchsatzdaten und große Graphen erfordern immer noch vergleichsweise lange Download- und Uploadzeiten.

Im Forschungsbereich werden häufig unterschiedliche Rechnerarchitekturen und Betriebssysteme verwendet. Als Implementierungssprache wurde daher Java gewählt. Java unterstützt moderne objektorientierte Sprachkonstrukte und verfügt bereits in der Standard-API über Klassen und Methoden zur Verarbeitung von XML-Dateien, für den Zugriff auf Datenbanken und für den Datenaustausch mit Webservern. Als Implementationsbasis wurde das an der Universität Passau im Rahmen eines Programmierpraktikums entwickelte Framework Gravisto [51] gewählt. Es basiert auf dem Model-View-Controller-Konzept (MVC), welches Datenstrukturen, Darstellung und Interaktionselemente der Benutzeroberfläche als individuelle Komponenten betrachtet. Ein wichtiges Entwicklungsziel von Gravisto war die

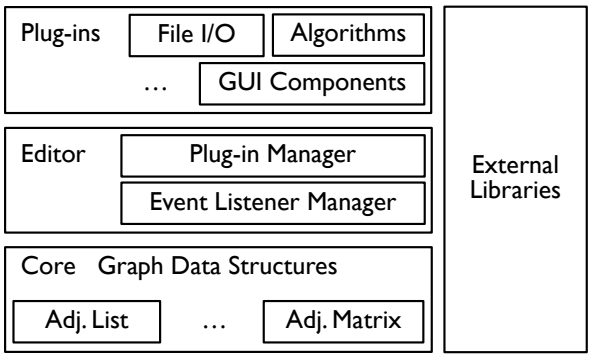

Abbildung 4.1: Überblick über die VANTED Systemarchitektur: Die unterste Ebene „Core“ definiert grundlegende Graphstrukturen, Plug-in- und Ereignis (Event)-Interfaces. Die „Editor“-Ebene implementiert den Standard-Plug-in-Manager und das Event-Management. Die „Plug-in“-Ebene enthält Nutzerkommandos und GUI-Komponenten.

Möglichkeit zur komponentenbasierten Erweiterbarkeit des Systems durch Plug-ins.

Das System besteht aus drei Ebenen, welche grundlegende Funktionalitäten gegeneinander abgrenzen (siehe Abbildung 4.1). Auf unterster Ebene „Core“ sind grundlegende Graphdatenstrukturen, Ereignisse sowie die Plug-in-Schnittstelle definiert. Darüber befindet sich die „Editor“-Ebene, welche das Anwendungsfenster zur Interaktion mit dem System bereitstellt. Teil dieser Ebene ist ein Plug-in-Manager zur Verwaltung der Systemerweiterungen. Jedes Plug-in stellt bestimmte Funktionalitäten bereit, beispielsweise Komponenten zur Erweiterung der grafischen Benutzeroberfläche oder Datei-Import/Export-Filter. Der Datenaustausch zwischen den Programmkomponenten ist entsprechend dem Observer-Entwurfsmuster gestaltet [102]. Programmkomponenten teilen dem Event Manager ihr Interesse an Informationen über Änderungen einer anderen Komponente mit und werden ab diesem Zeitpunkt zum Beispiel über Änderungen der Graphstruktur informiert. Auf oberster „Plug-in“-Ebene sind, mit Ausnahme grundlegender Editorkommandos, alle Benutzerkommandos, GUI-Komponenten sowie Datei- und Datenbankschnittstellen implementiert. Auf allen Ebenen sind für die Implementation verschiedener Funktionen externe Bibliotheken notwendig.

Anpassungen der Ebenen Core, Editor und Plug-ins Die im vorigen Kapitel vorgestellte Methodik wurde als Erweiterung des Gravisto-Systems implementiert. Dafür notwendige Änderungen im „Core“ waren minimal, beispielsweise wurde ein ID-Attribut zum Graphelementinterface hinzugefügt, um Elemente leichter vergleichen zu können.

Im Bereich Editor waren insbesondere zur Verbesserung der Benutzerfreundlichkeit umfangreichere Erweiterungen und Anpassungen notwendig. Unter anderem werden von Algorithmen bereitgestellte Menübefehle nicht mehr in ein einziges Hauptmenü einsortiert, sondern in ein vom jeweiligen Algorithmus spezifiziertes Hauptmenü, welches gegebenenfalls angelegt wird. Eine neue API ermöglicht die ein-

heitliche Behandlung und Präsentation von Fehlermeldungen sowie die Präsentation von Statusmeldungen bei Hintergrundprozessen. Da viele Aspekte der Benutzerschnittstelle ursprünglich eher prototypisch umgesetzt waren, wurde beispielsweise die Plug-in-Schnittstelle so erweitert, dass für Knoten- und Kantenattribute anstatt des oft unverständlichen Attributnamens eine Beschreibung (Text oder Bild) im Attributeditor angezeigt werden kann. Die vorherige Festlegung auf drei Knotenshapes (Kreis, Ellipse, Rechteck) wurde aufgehoben, Plug-ins können weitere Shapes zur Darstellung von Knoten bereitstellen. In VANTED können Plug-ins als *optional* markiert werden. Somit kann der Anwender für unterschiedliche Aufgabenstellungen nicht benötigte Programmfunktionen ausblenden und behält so leichter den Überblick über die umfassenden Programmfunktionen.

Die Beschreibung von Implementationsdetails zur Umsetzung des Mappinggraph-Konzepts und zur Verarbeitung von Experimentdaten erfolgt im Anschluss an den folgenden kurzen Überblick über die implementierten Erweiterungen auf Plug-in-Ebene. Gelistet ist jeweils das implementierte Java-Interface sowie die Anzahl der entsprechenden Erweiterungen und eine kurze Funktionsbeschreibung:

- `Algorithm` (112): Algorithmen zur Auswertung und Anpassung der Graphstruktur (z.B. Layout, Datenauswertung, Bildexport, Druck)
- `Attribute, AttributeDescription` (20): Definition von Knoten- oder Kantenattributen und Attributbeschreibungstexten
- `Input-/OutputSerializer` (8): Erweiterung der Dateischnittstelle
- `AttributeComponent` (2): Komponenten zur Datenvisualisierung im Graphkontext
- `GUIComponent` (6): GUI-Komponenten (Toolbars für Suche, Zoom, Zwischenablage)
- `Shape` (22): zusätzliche Knotenshapes (beispielsweise Elemente zur SBGN-konformen Darstellung von Graphen [103])
- `Tool` (2): Editortools zum Hinzufügen, Löschen und Modifizieren von Knoten und Kanten
- `ValueEditComponent` (25): GUI-Komponenten zur Änderung von Attributen
- `InspectorTab` (15): GUI-Komponenten (Seitenpanels)
- `View` (2): Graph-Views zur Darstellung der Graphstruktur sowie Statistikview zur Auswertung von Attributausprägungen und Darstellung als Histogramm

Implementation Mappinggraph Mappinggraphen sind im VANTED-System als attributierter Graph implementiert. Bereits im Gravisto-System definierte Attribute erlauben die Zuweisung von Labels, Knoten- und Kantentypen. Eine neu definierte Attributklasse dient der Zuordnung von Experimentdaten zu Graphelementen

(Details im nächsten Abschnitt). Anwender können mithilfe der Grapheditorfunktionen verschiedene Typen von Mappinggraphen manuell modellieren. Unterschiedliche Typen von biologischen Netzen können aber auch mithilfe verbesserter oder neu entwickelter Dateischnittstellen (GML, GraphML, DOT, KGML, SBML) geladen werden. Zur Manipulation von zugeordneten Experimentdaten wurden eine Reihe von Algorithmen (zur Auswertung, zum Entfernen von Daten, zur datenspezifischen Anpassung von Farben, Formen und Größen) implementiert. Die Datenvisualisierung in Form von Diagrammen wird durch neu entwickelte Attributkomponenten ermöglicht.

Implementation Experimentdatenmodell Die Ergebnisse eines ein- oder mehrfaktoriellen Experiments werden vom System aus kommaseparierten Textdateien (CSV) oder aus dem Inhalt von Microsoft-Excel-Dateien (XLS) generiert und als XML-Dokument umgesetzt. Die XML-Datenstruktur ist in Form eines XML-Schemas (XSD) definiert. Im Rahmen des Datenmappings werden daraus XML-Attribute generiert, welche den Zielknoten und -kanten zugeordnet werden. Diese speichern die zugeordneten Experimentdaten in Form von XML-Objektbäumen (DOM). Sie ermöglichen den XPATH-basierten Zugriff auf zur Visualisierung und Analyse benötigte Daten. Bei Bedarf, zum Beispiel beim Speichern des Graphen in einer Datei, erfolgt eine Serialisierung als XML-String.

4.3 Systemfunktionen aus Anwendersicht

VANTED ist ein in Java implementiertes Desktopprogramm mit grafischer Benutzeroberfläche für sämtliche Programmfunktionen (Bildschirmfoto in Abbildung 4.2). Die wichtigsten Funktionen werden im Folgenden anhand eines typischen Workflows vorgestellt (vergleiche Abbildung 4.3):

Experimentdaten können in das System mithilfe einer ausgefüllten Microsoft-Excel-Vorlage (XLS-Datei) geladen werden. Die Excel-Vorlage bietet den Vorteil, dass Informationen über Experimentbedingungen, Zeitpunkte und Replikatinformationen in der Datei an definierter Stelle eingegeben und automatisch eingelesen werden können. Alternativ können „unstrukturierte“ kommaseparierte Dateien eingelesen werden. Bei diesen erfolgt nach dem Laden eine Benutzerabfrage zur Annotation der Datenspalten. Das System unterstützt die gleichzeitige Auswahl mehrerer Datendateien, um mehrere Experimentdatensätze laden und vergleichen zu können. Im nächsten Schritt erfolgt die Auswahl eines geeigneten Zielnetzwerks. Ein vorhandener Mappinggraph kann mithilfe der Dateischnittstelle oder von einem Webserver (KEGG-Pathways, MetaCrop-Pathways) geladen werden. Alternativ können die Grapheditorfunktionen zum Zeichnen eines neuen Mappinggraphen verwendet werden. Anschließend erfolgt das Datenmapping. Optional werden dabei verschiedene Datenbanken für Synonyminformationen über Metabolite, Enzyme und Gene oder datenspezifische Annotationsdateien berücksichtigt. Die Mappinggraph-View erzeugt für zugeordnete Experimentdaten automatisch passend ausgewählte Visua-

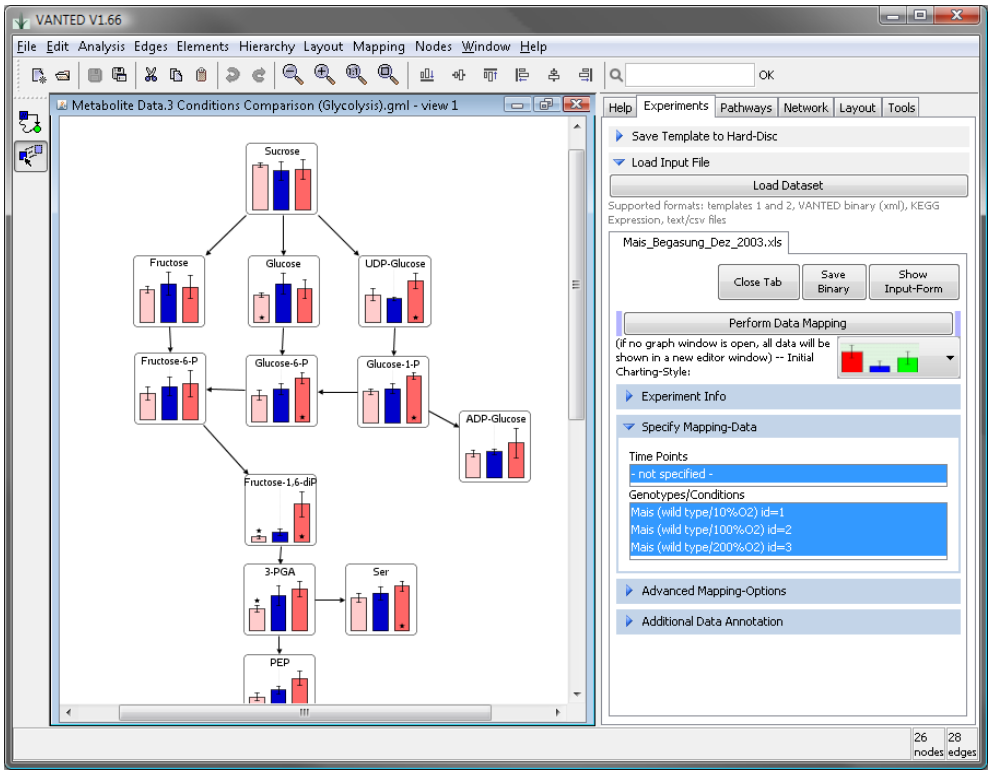

Abbildung 4.2: Grafische Benutzeroberfläche von VANTED.

lisierungskomponenten. Sind mehrere Experimentbedingungen und ein einziger Zeitpunkt im Datensatz enthalten, wird beispielsweise für jedes als XML-Attribut zugeordnete *mapping*-Objekt ein Balkendiagramm erzeugt. Ausreißer in den *sample*-Daten können mit statistischen Analysealgorithmen erkannt und entfernt werden (Kapitel 2.2.2). Die verschiedenen, in der Methodik vorgestellten Varianten der interaktiven Korrelationsanalyse (Kapitel 2.2.3) können vom Benutzer von einem „Statistik“-Seitentab (rechter Bereich des Anwendungsfensters) aus parametrisiert und gestartet werden. Ebenfalls in Seitentabs sind weitere interaktiv nutzbare Funktionen zum Vergleich von Sample-Durchschnittswerten (t-Test) und zum Graphlayout angesiedelt. Analysefunktionen, die keiner Parametrisierung bedürfen oder die nur selten mehrfach hintereinander ausgeführt werden, sind als Menübefehle implementiert. Die Ergebnisse der Arbeit mit dem Programm (Visualisierung eines Mappinggraphen inklusive der zugeordneten Experimentdaten) können als Datei gespeichert werden (GML, GraphML), gedruckt oder als Bild exportiert werden (JPG, PNG, PDF, SVG). Es ist ebenfalls möglich, eine „Clickable Imagemap“, also eine HTML Datei mit eingebettetem Bild und Links zu anderen Visualisierungen oder Informationsressourcen zu exportieren. Die grafische Pathwaydarstellung im MetaCrop System [98] basiert beispielsweise auf dieser Programmfunktion. Neben dem Sprung von Mappinggraph zu Mappinggraph können bei der Arbeit mit VANTED verlinkte Mappinggraphen in die aktuelle Ansicht integriert werden (siehe Kapitel 3.3.3).

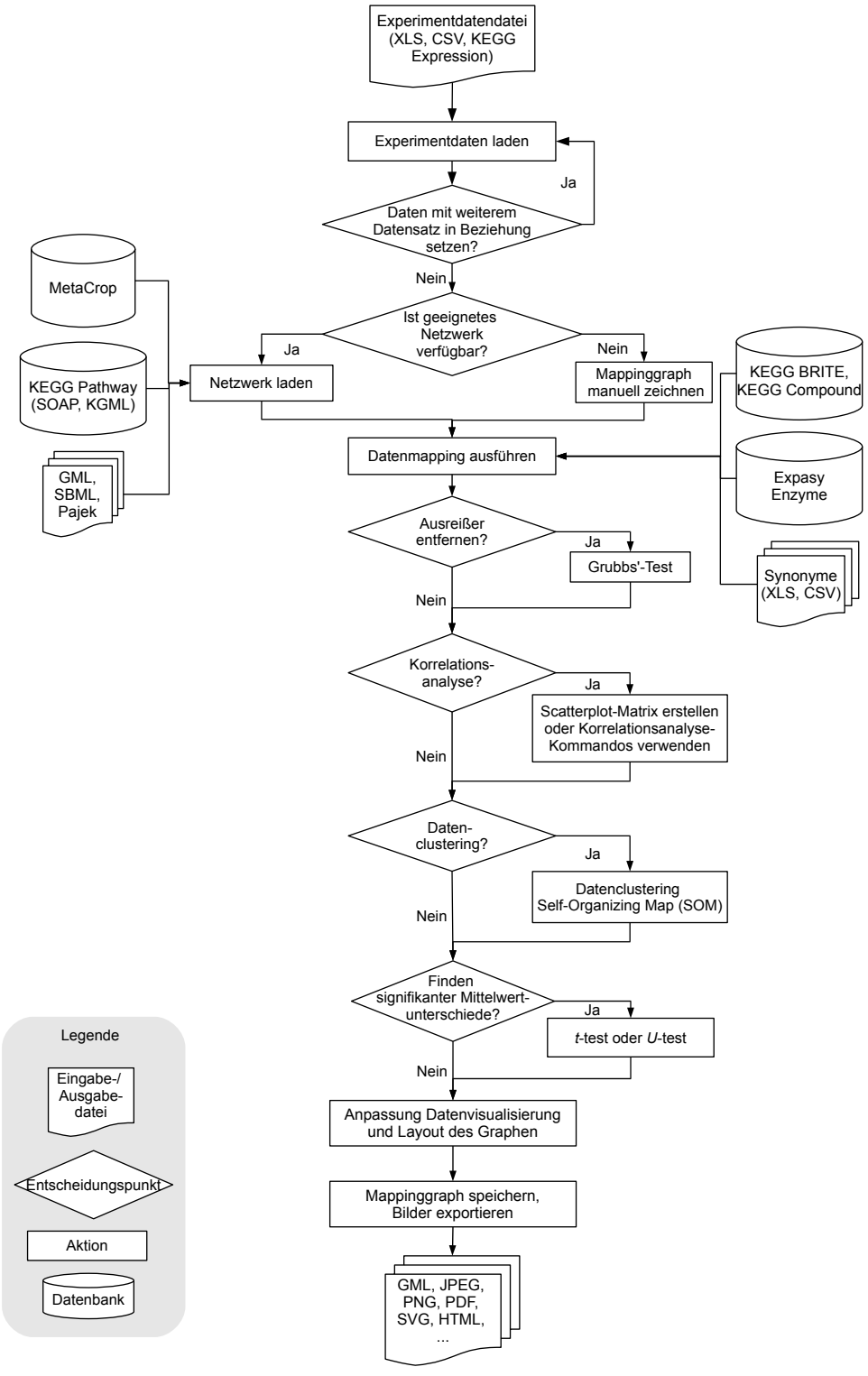

Abbildung 4.3: Überblick über einen typischen VANTED-Workflow. Die Reihenfolge der Arbeitsschritte kann vom Anwender frei gewählt werden. Aktionen können rückgängig gemacht und bei Bedarf neu parametrisiert wiederholt werden.

4.4 Vergleich mit anderen Systemen

Im Rahmen einer Vergleichsstudie [104] konnten mehr als 170 Anwendungsprogramme identifiziert werden, welche sich grob betrachtet mit dem Thema Graphvisualisierung im biologischen Umfeld beschäftigen. Von diesen, mithilfe von Literatur- und Webrecherchen gefundenen Tools, sind 51 als Open-Source-Software verfügbar, 24 werden kommerziell vertrieben, 13 sind webbasierte Anwendungen und 39 in der Literatur erwähnte Tools sind nicht mehr auf den angegebenen Webseiten oder anderweitig im Internet verfügbar. Aus dieser großen Liste von Tools entsprechen von ihrem Fokus her nur eine kleinere Anzahl von Tools zumindest teilweise den mit dieser Arbeit verfolgten Zielen, der flexiblen Visualisierung biologischer Netzwerke unter Berücksichtigung der betrachteten Anforderungen von Experimentatoren, insbesondere zur Visualisierung und Analyse von Experimentdaten. In Kooperation mit Anwendern aus der Industrie (BASF/Sungene) wurden 12 Anwendungsprogramme für den detaillierten Funktionsvergleich zu VANTED ausgewählt: BioTapestry [105], BioUML [106], CellDesigner [107], Cytoscape [108], ONDEX [109], PathBuilder [110], Path Visio [111], Pathway Builder [112], PathwayLab [113], Pathway Logic [114], VisANT [115] und VitaPad [116]. Diese Auswahl basiert auf Angaben von Webseiten oder Journalartikeln hinsichtlich Funktionalität und Programmfokus.

Im Folgenden werden die genannten Anwendungsprogramme kurz vorgestellt und am Ende des Abschnitts die wesentlichen Unterschiede zum VANTED-System herausgestellt. Einen Überblick zur Verfügbarkeit (Lizenz), zur getesteten Programmversion und über die wichtigsten Funktionalitäten der vorgestellten Tools gibt Tabelle 4.1.

Das **VitaPad** System konnte im Test nicht gestartet werden, da es versuchte, sich beim Start zwingend mit nicht mehr aktiven Datenbankservern zu verbinden. VitaPad als nicht mehr aktiv gepflegtes Projekt entfällt daher bei diesem Vergleich. Das am Institut für Systembiologie in Seattle, USA, entwickelte Tool **BioTapestry** dient hauptsächlich der Konstruktion und Untersuchung von genetischen Netzwerken und ist somit nicht zur Analyse von anderen biologischen Netzwerken oder Klassifikationshierarchien geeignet. Nur einfache Datensätze, keine Zeitreihen oder Daten unterschiedlicher Umweltbedingungen können geladen und mithilfe eines Farbcodes visualisiert werden. Die Software **BioUML** wurde am Institut für Systembiologie in Novosibirsk, Russland, entwickelt. Wichtiger Programmfokus sind Simulationsfunktionen und Datenintegration – insbesondere die Ableitung von biologischen Netzwerken auf Basis verschiedener Datenbanken. Obwohl der Quelltext zum Download verfügbar ist, bleiben die Bedingungen für dessen Nutzung offen (keine Angaben zur Lizenz). Die Anwendung unterstützt keine Klassifikationshierarchien oder die gleichzeitige Visualisierung von Daten verschiedener Umweltbedingungen oder Zeitserien. Das am Systembiologie Institut in Tokio, Japan, entwickelte Programm **CellDesigner** verfügt über vielfältige Funktionen insbesondere zur Simulation von Genregulationsnetzwerken. Es wird stetig weiter entwickelt und enthält wie VANTED zukunftsweisende Funktionen zum Zeichnen von Diagrammen entsprechend dem SBGN-Standard (System Biology Graphical Notation). Es verfügt darüber hinaus über eine Integration in die „Systems Biology Workbench“ (SBW) – ein Software-Framework für den Daten- und Funktionsaustausch zwischen verschiedenen Anwen-

dungen [117]. Obwohl das System selbst nicht im Quelltext verfügbar ist, besteht die Möglichkeit, die Software mithilfe von Plug-ins zu erweitern. Das von einem Konsortium verschiedener Institutionen entwickelte Open-Source-Tool **Cytoscape** enthält eine große Zahl von Programmfunktionen zur Arbeit mit verschiedenen biologischen Netzwerken oder sonstigen Graphen. Es ist hinsichtlich Visualisierung von Graphen und Daten flexibel (enthält zum Beispiel verschiedene Layoutfunktionen). Strukturierte Experimentdaten können nur mithilfe einer Farbcodierung dargestellt werden. Es sind keine eingebauten Netzwerkanalysefunktionen und auch keine Statistikfunktionen verfügbar. Es wird aber die Entwicklung von Plug-ins unterstützt. Diese Möglichkeit wird bereits von einigen externen Gruppen genutzt, um die Funktionalität des Systems zu erweitern. Das von Mitarbeitern verschiedener Universitäten, unter anderem in Bielefeld, Deutschland und Nottingham, Großbritannien, entwickelte Softwaresystem **ONDEX** ist auch unter GPL Lizenz im Quelltext verfügbar und enthält eine Reihe von Programmroutinen zur Verknüpfung von Daten einer größeren Anzahl von Datenbanken. Der Programmfokus liegt klar in der Generierung von biologischen Netzwerken. Genexpressionsdaten können mithilfe eines Farbcodes im Netzwerkkontext dargestellt werden. Die gleichzeitige Visualisierung von Zeitreihendaten oder Daten verschiedener Umweltbedingungen wird nicht unterstützt. Das vom Pandey Lab in Baltimore, USA, entwickelte System **PathBuilder** ist ein webbasiertes Open Source Tool (LGPL) zur Annotation von biologischen Pathways, welche in der angeschlossenen Datenbank gespeichert sind. PathBuilder enthält keine Funktionen zum Graphediting sowie keine Unterstützung zur Visualisierung von Experimentdaten. Das am Institut für Bioinformatik an der Universität Maastricht, Niederlande, entwickelte Tool **Path Visio** dient hauptsächlich der Visualisierung und dem Editing von Pathways im GPML- und GenMAPP-Dateiformat. Das Programm ermöglicht insbesondere den Zugriff auf Pathways des WikiPathways-Systems [118]. Das Programm ist in der Lage, Microarraydaten und Proteomicsdaten mithilfe einer Farbcodierung zu visualisieren. Dabei werden Zeitreihendaten, aber keine unterschiedlichen Umweltbedingungen unterstützt. Nachteilig ist, dass Experimentdaten nur vom angeschlossenen Datenbankserver und nicht aus lokalen Dateien geladen werden können. Weiterhin werden keine Netzwerkanalysen und auch keine Statistikfunktionen unterstützt. Das von der Firma Protein Lounge in San Diego, USA, vertriebene Tool **Pathway Builder** dient hauptsächlich der Generierung von Darstellungen von Signaltransduktionsnetzwerken. Im Test wurde festgestellt, dass dieses Tool nicht graphbasiert ist und somit auch kein Graphediting und keine Graphanalysefunktionen unterstützt. Experimentdaten können vom Tool ebenfalls nicht verarbeitet werden. Als nützliche Funktionen verbleiben der Zugriff auf vorbereitete, ansprechende Pathwaydarstellungen aus der enthaltenen Pathway(bild)datenbank sowie die Möglichkeit zum Zeichnen von eigenen Bildern, dabei ist jedoch die Größe der Zeichenfläche durch das Fehlen von Funktionen für Scrolling und Zoom fest vorgegeben. **PathwayLab** ist ein von der Firma InNetics AB in Linköping, Schweden, vertriebenes kommerzielles Anwendungsprogramm zur Analyse, Visualisierung und Simulation von biochemischen Pathways. Die Anwendung wurde als Programmerweiterung für Microsoft Visio entwickelt. Die Pathways werden mithilfe von Visio-Vorlagen als Grafik gezeichnet. Mithilfe von Dialogfenstern kann ein darauf basieren-

des Simulationsmodell zusammengestellt werden. Ergebnisse der Simulation können in Diagrammform unabhängig von der Pathwaydarstellung angezeigt werden, vorhandene Experimentdaten können jedoch nicht verarbeitet werden. Das Programm **Pathway Logic** des Forschungsinstituts SRI International in Menlo Park, Kalifornien, USA, dient der Generierung und Simulation von (biologischen) Netzwerken. Es fehlen Programmfunktionen zum Zeichnen von Netzwerken sowie zur Arbeit mit Experimentdaten. Es bietet jedoch einige Netzwerkanalysefunktionen (Graphvergleich und Suche nach Pfaden). Das an der Universität Boston, USA, entwickelte Tool **VisANT** dient der Visualisierung und Analyse von biologischen Netzwerken und Pathways. Es ist möglich, Experimentdaten mithilfe einer Farbcodierung oder durch Variation von Knotengröße oder Kantenstärke darzustellen. Außerdem werden Liniendiagramme zur Visualisierung von Zeitseriendaten unterstützt. Die Kombination unterschiedlicher Umweltbedingungen mit Zeitseriendaten wird jedoch nicht unterstützt. Es bietet eine vergleichsweise große Zahl von Visualisierungs- und Analysefunktionen, jedoch keine Funktionen zum Zeichnen von Graphen.

Im Vergleich zu den anderen Tools bietet VANTED eine große Anzahl an Visualisierungs- und Analysefunktionen. Das vom Funktionsumfang ähnlich umfangreiche Tool VisAnt kann gleichzeitig keine Umweltbedingungen und Zeitserien verarbeiten und bietet keine Möglichkeit zum eigenständigen Zeichnen von Pathways. Der Anwender ist auf externe Tools zum Zeichnen der Graphen angewiesen. ONDEX als ebenfalls sehr leistungsfähiges Tool hat einen klaren Fokus auf Datenintegration und Netzwerkgenerierung, bietet jedoch keine Möglichkeit zum Mapping von Experimentdaten verschiedener Umweltbedingungen oder von Zeitseriendaten. Neben VANTED bieten nur zwei weitere Programme Statistikfunktionen, diese sind jedoch aufgrund der fehlenden Unterstützung für komplex strukturierte Experimentdatensätze nicht vergleichbar: ONDEX bietet Histogramm- und Filterfunktionen, aber keine statistischen Tests zum Beispiel für den Vergleich von Sample-Durchschnittswerten. Pathway Lab verfügt über spezielle Simulationsfunktionen. Ein wichtiger Unterschied zu allen im Vergleich berücksichtigten Anwendungsprogrammen ist, dass VANTED Experimentdaten mit gleichzeitiger Berücksichtigung unterschiedlicher Umweltbedingungen, Zeitserien und Replikatangaben unterstützt. Keines der untersuchten Tools war in der Lage, flexibel unterschiedliche Diagrammformen wie Kreis-, Balken- oder Liniendiagramme zur Datenvisualisierung zu verwenden.

	BioTapestry	BioUML	CellDesigner	Cytoscape	ONDEX	PathBuilder	Path Visio	Pathway Builder	PathwayLab	Pathway Logic	VANTED	VisANT
Version	3.05	0.83	4.01	2.61	ib08	1.0	1.12	2.0	1.2	3.0	1.6	3.29
Lizenz	LGPL	n/a	©	LGPL	GPL	LGPL	Ap.2	$	$	GPL	GPL	©
Hintergrundbilder	·	·	·	·	·	·	·	✓	✓	·	✓	·
Export als Bild	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Graphbasiert	✓	✓	✓	✓	✓	✓	✓	·	·	✓	✓	✓
Editormodus	✓	✓	✓	✓	✓	·	✓	✓	✓	·	✓	·
Organismusspez. Auswahl	·	✓	·	✓	✓	·	·	·	·	·	✓	✓
Abfrage von Metadaten	·	✓	·	✓	✓	·	✓	·	·	·	✓	✓
Semantic Zoom	·	·	✓	·	✓	·	·	·	·	·	·	·
Links zu externen Quellen	·	·	✓	✓	✓	·	✓	✓	✓	·	✓	✓
Große Graphen	✓	·	✓	✓	✓	·	✓	·	·	·	✓	✓
Zoom, Panning	✓	·	✓	✓	✓	·	·	·	✓	·	✓	✓
Labels und Daten anzeigen	✓	✓	·	·	✓	·	✓	·	·	·	✓	✓
Laden von Experimentdatendateien	✓	✓	·	✓	✓	·	✓	·	·	·	✓	✓
Klassifikationshierarchien	·	·	·	·	✓	·	·	·	·	·	✓	✓
Datenmapping	·	·	·	✓	✓	·	·	·	·	·	✓	✓
Color-Coding	✓	·	✓	✓	✓	✓	✓	·	·	·	✓	✓
Shape-Coding	·	✓	✓	✓	✓	·	✓	·	·	·	✓	✓
Diagramme	·	·	·	·	·	·	·	·	·	·	✓	✓
Kompartimente	✓	✓	✓	·	✓	·	·	✓	✓	·	✓	✓
Zeitseriendaten	✓	·	·	·	·	·	✓	·	✓	·	✓	✓
Umweltbedingungen	·	·	·	·	·	·	·	·	·	·	✓	✓
Netzwerkanalysefunktionen	·	·	·	·	✓	·	·	·	·	✓	✓	✓
Statistikfunktionen	·	·	·	·	✓	·	·	·	✓	·	✓	·

Tabelle 4.1: Vergleichende Darstellung der wesentlichen Funktionalitäten verschiedener Visualisierungstools. ✓ – Funktion verfügbar, · – Funktion nicht verfügbar, © – kostenlos nutzbar (CellDesigner) oder kostenlos für nicht kommerzielle Nutzung (VisANT), kein Open Source, $ – kommerzielles Anwendungsprogramm, GPL/LGPL/Apache 2 – Open Source Lizenz, n/a – Lizenz nicht genannt.

5 Anwendungsbeispiele

Die interaktiven und explorativen Aspekte der Nutzung des VANTED-Systems können nur unzureichend in Schriftform oder mit wenigen Abbildungen dargestellt werden. Daher steht in diesem Abschnitt die Bearbeitung konkreter biologischer Fragestellungen mit dem Ziel der Erstellung von anschaulichen Abbildungen für Artikel, Poster oder Vorträge im Vordergrund. Es werden vier Anwendungsbeispiele vorgestellt: Das erste Beispiel zeigt, wie Metabolitdaten unterschiedlicher Pflanzenlinien übersichtlich dargestellt und somit leichter analysiert werden können. Das zweite Beispiel zeigt die Visualisierung von durch Simulation berechneten Kohlenstoffflussdaten. Dazu werden Simulationsergebnisse Graphkanten zugeordnet und mithilfe einer Kantenstärkencodierung visualisiert. Das dritte Anwendungsbeispiel zeigt, wie Experimentdaten unterschiedlicher omics-Ebenen (hier Proteomics und Metabolomics) in einem KEGG-Pathway integriert dargestellt werden können und wie ein datenspezifischer KEGG-BRITE-Mappinggraph den Überblick über die zur Datenauswertung relevanten Pathways verbessert. Im letzten Beispiel werden mittels Datenclustering nach ähnlichen Konzentrationsverläufen gruppierte Zeitreihendaten im Netzwerkkontext dargestellt.

5.1 Visualisierung und Analyse von Metabolitdaten transgener Pflanzen

Bohnen und andere Hülsenpflanzen sind eine ökonomisch wichtige pflanzliche Proteinquelle für die Nahrungsmittelproduktion. Die Steigerung des Proteingehalts von Bohnensamen ist daher eines der ehrgeizigen Ziele in der Pflanzenzucht. Im Rahmen eines Experiments wurden genetisch veränderte Linien einer Bohnenart (*Vicia narbonensis*) hergestellt. Dazu wurde das Pflanzengenom so verändert, dass das Enzym „phosphoenolpyruvate carboxylase“ (PEPC) exprimiert wird. Über einige Zwischenschritte werden so bestimmte Vorprodukte von Aspartate, Malate und anderer Metabolite des Zitronensäurezyklus beeinflusst [119]. PEPC kontrolliert den Kohlenstoffffluss und kann dadurch den Kohlenstoffumsatz im Bohnensamen verbessern [120]. Die Analyse von ausgewachsenen Samenkörnern transgener Pflanzen hat im Versuch eine signifikante Erhöhung der Proteinkonzentration um bis zu 20 % pro Gramm und ein höheres Trockengewicht ergeben. Die Kombination beider Effekte zeigt einen Proteinanstieg der Samenkörner um 40 % bis 50 % [119]. Isotopenindikatorexperimente konnten im Rahmen dieser Studie eine klare Stimulation der [14C]-CO_2-Aufnahme und dessen Verwendung für den Proteinaufbau nachweisen. Um die Veränderung der Prozesse, die zur Synthese von Aminosäuren und Proteinen führen, genauer charakterisieren zu können, wurden Metabolitprofile für die Glyko-

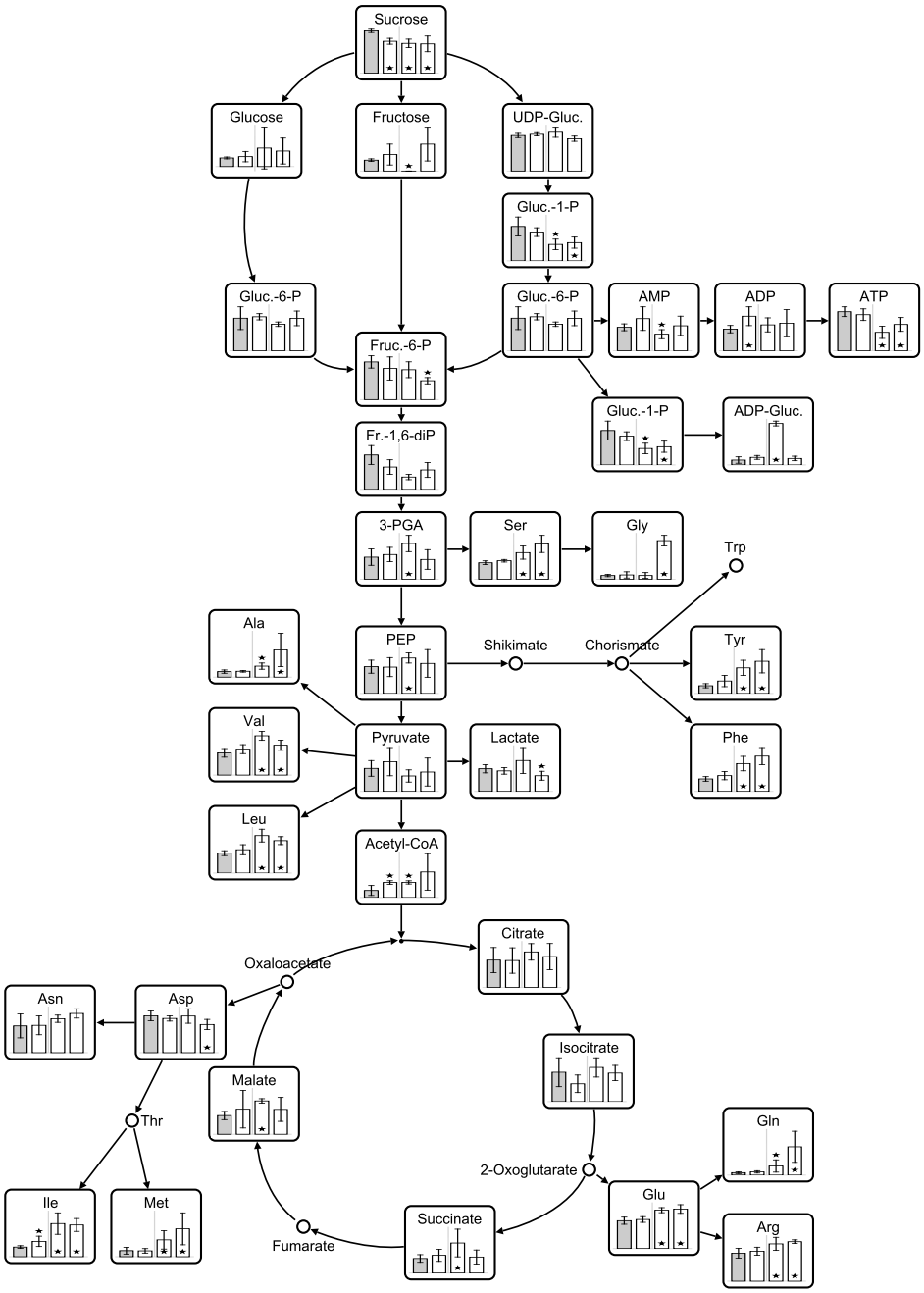

Abbildung 5.1: Vereinfachtes biochemisches Reaktionsnetzwerk (Glykolyse und Zitronensäurezyklus). Relative Metabolitkonzentrationen verschiedener Bohnenpflanzenlinien (*Vicia narbonensis*, Wildtyp in Grau, transgene Linien in Weiß) sind in Diagrammform visualisiert. Signifikante Mittelwertunterschiede der verschiedenen Linien im Vergleich zum Wildtyp wurden mit einem t-Test ($\alpha \leq 0,05$) ermittelt und mit Sternchen gekennzeichnet.

lyse, den Zitronensäurezyklus sowie von Zuckern und freien Aminosäuren erhoben. Die Metabolitkonzentrationen wurden dabei mit einer Flüssigkeitschromatographie in Verbindung mit einer Massenspektroskopie (LC-MS, siehe Kapitel 2.1.3) ermittelt. Der Effekt des Transgens zeigte sich dabei mit unterschiedlicher Intensität für die verschiedenen Pflanzenlinien [119]. Die Visualisierung in Abbildung 5.1 ermöglicht dem Biologen, einen schnellen Überblick über die verschiedenen Effekte auf den Metabolismus der Pflanzen zu bekommen. Details zu Standardabweichungen und Hinweise auf signifikante Mittelwertunterschiede können übersichtlich in die Diagrammdarstellung integriert werden. Insgesamt lässt sich nach visueller Inspektion der Daten feststellen, dass die Expression des bakteriellen PEPC-Enzyms zu einer Veränderung der Metabolitprofile führt und eine Verschiebung der Konzentrationen von Zucker und Stärke zu organischen Säuren, Aminosäuren und Proteinen stattfindet. Die Verbindung von Techniken zur Ermittlung von Metabolitprofilen in Verbindung mit den hier vorgestellten Methoden zur ansprechenden Visualisierung und statistischen Auswertung der Daten erlaubt es dem Anwender des VANTED-Systems auf einfache Art und Weise, verschiedene transgene Linien zu vergleichen. In bestimmten Fällen kann eine solche Darstellung auch dabei helfen, Ziele für weitere transgene Veränderungen eines Organismus zu finden.

5.2 Simulation des Metabolismus und Visualisierung des berechneten Kohlenstoffflusses

Simulationsmodelle des Pflanzenmetabolismus ermöglichen einen verbesserten Einblick in die komplexen biochemischen Prozesse in einer lebenden Zelle. Ziel ist es, Struktur, Dynamik und Verhalten des Modells näher zu untersuchen, um Hypothesen über das Verhalten in der Realwelt abzuleiten. Die im Rahmen dieser Arbeit entwickelten Visualisierungs- und Analysemethoden ermöglichen es, Simulationsergebnisse und das grafische Modell des Stoffwechselweges integriert zu betrachten. Dabei gibt es eine Reihe verschiedener Ansätze zur Simulation biochemischer Prozesse. Für dieses Beispiel wurde die „flux balance analysis“ (kurz FBA), eine Methode zur Analyse des Fließgleichgewichts, herangezogen. FBA hat den Vorteil, dass kein Vorwissen über die kinetischen Parameter der biochemischen Reaktionen benötigt wird. Es genügt das Wissen um die Stöchiometrie der beteiligten Reaktion, also den Mengen der Ein- und Ausgangsstoffe. In VANTED wird diese Information automatisch dem Datenbestand der KEGG-Compound-Datenbank entnommen. Weiterhin wird eine Zielfunktion benötigt, welche als Optimierungskriterium für die Auswahl des besten Fließgleichgewichts verwendet wird. In dem betrachteten Modell [121] wurde der zentrale Stoffwechsel des Nährgewebes (Endosperm) eines Gerstenkorns abgebildet. Die anhand von Literaturrecherchen und Datenbankabfragen erhobenen Daten wurden initial inklusive Literaturreferenzen und weiteren Detailangaben in die MetaCrop-Datenbank [98] übertragen. Die grafische Modellierung des biochemischen Reaktionsnetzwerkes erfolgte mit VANTED. Zur Simulation wurde die COBRA-Toolbox [122] verwendet, welches die von VANTED generierte SBML-Datei als Eingabe verwendet. Die Ergebnisse des Simulationslaufs werden au-

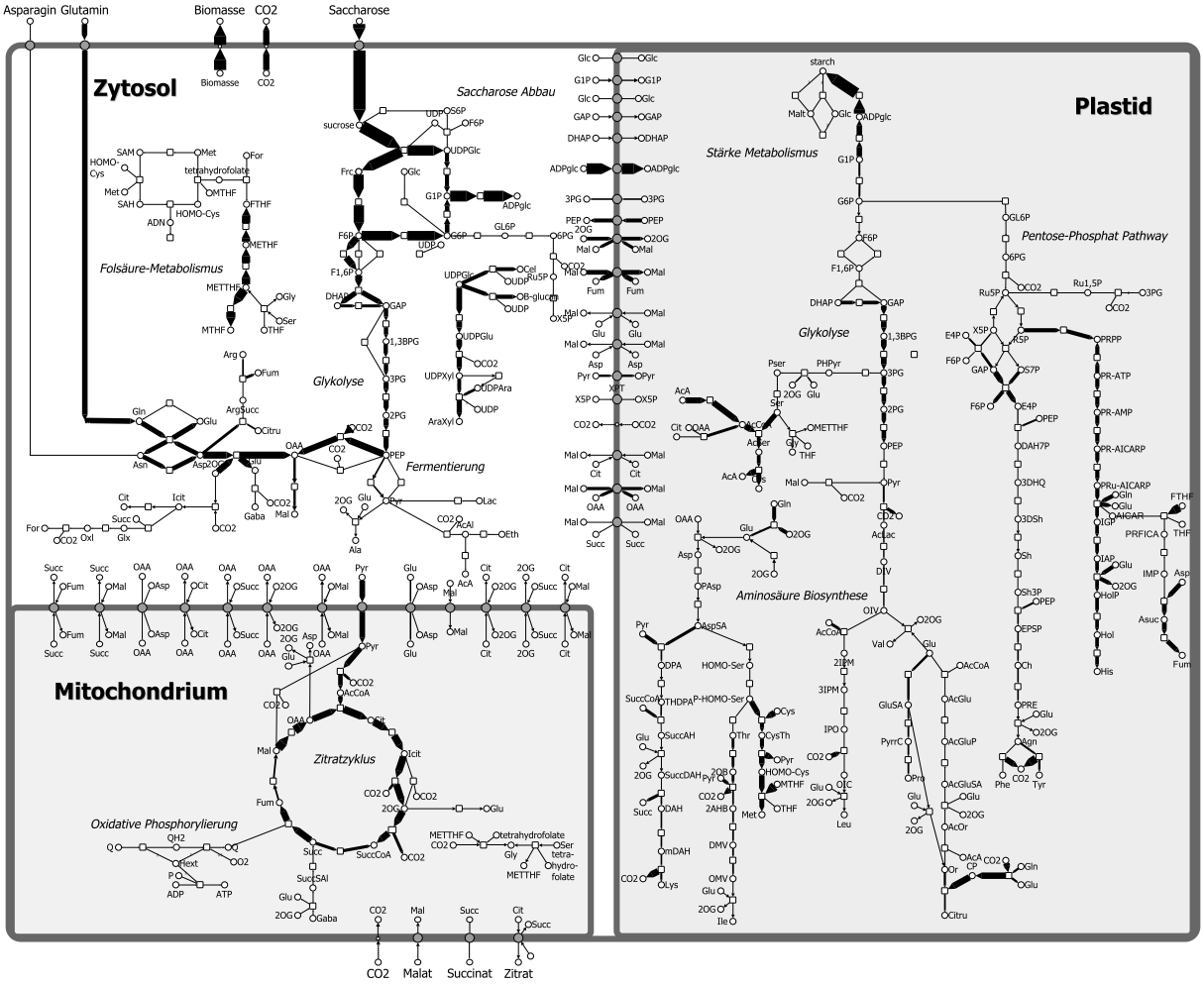

Abbildung 5.2: Visualisierung des Kohlenstoffflusses im Getreidesamen unter simulierten, optimalen Wachstumsbedingungen. Datenquelle für die Reaktionsdaten ist MetaCrop [98]. Details zur Integration der zur Modellierung und Simulation verwendeten Tools in VANTED, siehe [121].

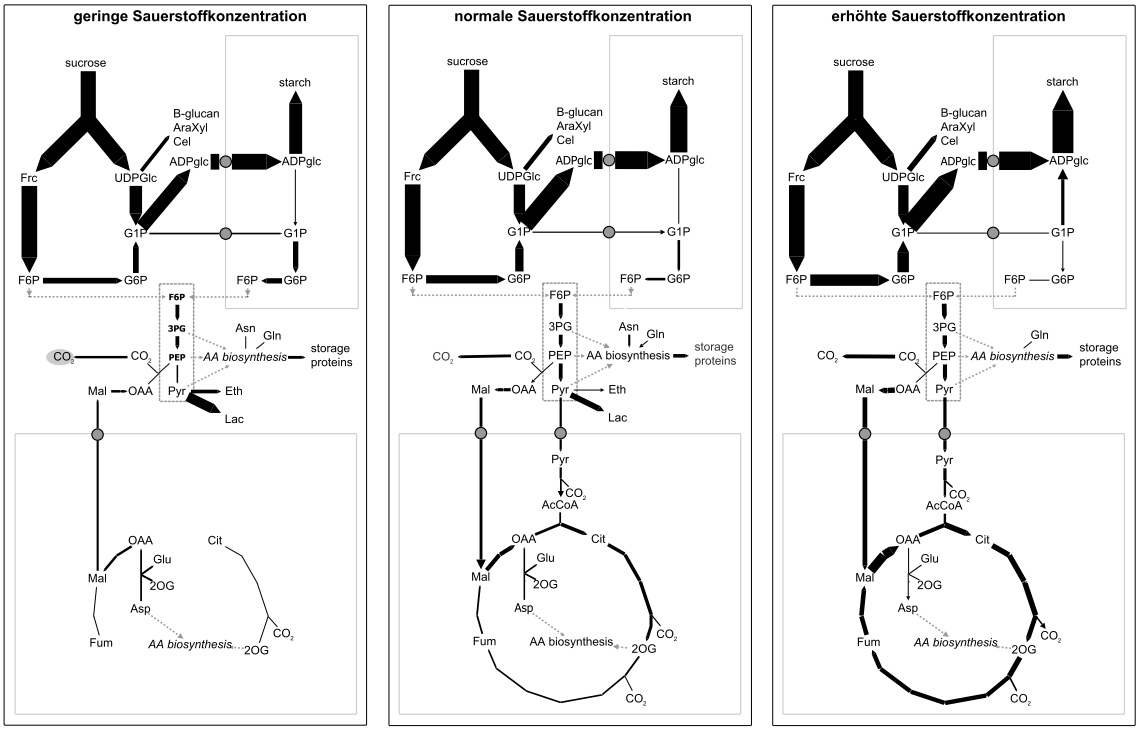

Abbildung 5.3: Detaildarstellung unterschiedlicher Wachstumsbedingungen (verringerte, normale und erhöhte Sauerstoffkonzentration in der Umgebungsluft). Im VANTED-System können die gemappten Ergebnisse unterschiedlicher Simulationsbedingungen interaktiv mit einem Schieberegler ausgewählt und nahezu verzögerungsfrei visualisiert (animiert) werden.

tomatisiert als kommaseparierte Textdatei geladen und mittels Datenmapping den Kanten des graphbasierten Simulationsmodells zugeordnet (siehe Abbildung 5.2). Da VANTED auch komplex strukturierte Experimentdatensätze unterstützt, können mehrere Simulationsläufe unter gleichen Simulationsbedingungen als eine Reihe von Replikaten betrachtet und unterschiedliche Parametrisierungen als unterschiedliche Experimentbedingungen abgebildet werden. Die Daten mehrerer Simulationen mit unterschiedlichen Zielparametern können gleichzeitig mittels Diagrammen visualisiert werden. Abbildung 5.3 zeigt, wie alternativ ausgewählte Simulationsbedingungen schrittweise interaktiv durchlaufen und mittels Kantenstärkecodierung anschaulich dargestellt werden können. Die Auswertung und Darstellung der Simulationsergebnisse hat gezeigt, dass diese den wichtigsten biochemischen Eigenschaften des Gerstenmetabolismus entsprechen und das modellierte Simulationsmodell somit das Potenzial bietet, grundlegende Zusammenhänge des Getreidekornmetabolismus korrekt zu simulieren [121].

5.3 Metabolit- und Enzymdaten von genetisch modifizierten Kartoffelpflanzen

Die Kartoffel ist nach verschiedenen Getreidesorten die wichtigste Pflanze zur Stärkeproduktion. Daher wird die Regulation der Umwandlung von Saccharose nach Stärke in der Kartoffelknolle bereits seit mehreren Jahrzehnten studiert (einen Überblick gibt [123]). Um die Bedeutung der Saccharoseaktivierung näher zu untersuchen, wurde eine Reihe genetisch veränderter Kartoffelpflanzenlinien gezüchtet, in die ein Hefeenzym zur Beeinflussung der Saccharoseverarbeitung eingeschleust wurde [124]. Insgesamt konnten in diesem Experiment mehr als 60 verschiedene Metabolite und Enzyme quantifiziert und identifiziert werden. Traditionell wurden lange Tabellen zur Darstellung der Messergebnisse verwendet [29, 124]. Mit VANTED können die Ergebnisse der transgenen Modifikationen übersichtlich direkt im Kontext des darunterliegenden biochemischen Reaktionsnetzwerks dargestellt werden. Dazu wurden die Metabolit- und Enzymdaten aus der Publikation [124] für VANTED aufbereit und in das System geladen. Der Experimentdatensatz enthält die Messdaten der Metabolite und Enzyme. Sieben *condition*-Objekte bilden den Wildtyp und sechs genetisch modifizierte Pflanzenlinien im Datenmodell ab. Die letzte genetisch modifizierte Linie unterscheidet sich von den vorherigen fünf in der Art der Expression des Transgens (das Hefeenzym wird in den ersten fünf Linien durch sogenannte konstitutive Promotoren aktiviert, in der letzten Linie durch einen mit Ethanol induzierbaren Promotor). Ein datenspezifischer KEGG-BRITE-Mappinggraph kann auf der Basis der im Datensatz enthaltenen Metabolit- und Enzymidentifikatoren konstruiert werden (siehe Abbildung 5.4). Durch eine Filteroperation in VANTED wurde für dieses Beispiel die Auswahl auf Pathways beschränkt, welche sowohl passende Enzym- als auch Metabolitknoten enthalten. Eine Histogrammfunktion vereinfacht die Auswertung der Daten und zeigt, dass die Pathways „Starch and sucrose metabolism“ und „Glycolysis / Gluconeogenesis“ mit elf zuordenbaren Substanzen den größten Anteil der Daten abdecken und daher im Rahmen einer Datenauswertung betrachtet werden sollten. Für dieses Beispiel wurden die Metabolit- und Enzymdaten ausschließlich dem Glykolysepathway zugeordnet (siehe Abbildung 5.5). Die Visualisierung zeigt, dass ganze Bereiche des Metabolismus auf ähnliche Art und Weise durch die genetische Modifikation beeinflusst werden, während bestimmte Bereiche ein anderes Muster zeigen. Aus der Darstellung wird ersichtlich, dass die Metabolitkonzentrationen in der induzierbaren Linie stark erhöht sind, während die Aktivität der zugehörigen Enzyme sich vom Wildtyp nur wenig unterscheidet. Möglicherweise sind die Enzymaktivitäten hoch genug, um die temporär erhöhten Hexosekonzentrationen zu verstoffwechseln [124]. Entsprechende Zusammenhänge aus langen Tabellen mit Zahlenkolonnen herauszulesen, würde direkt abrufbares umfassendes Wissen über den Metabolismus und die dort ablaufenden biochemischen Reaktionen erfordern, was erstrebenswert ist, aber auch für Forscher aus dem Gebiet der Biologie nicht immer vorausgesetzt werden kann.

Ein Teil der Metabolitdaten wurde für eine n:n-Korrelationsanalyse ausgewählt

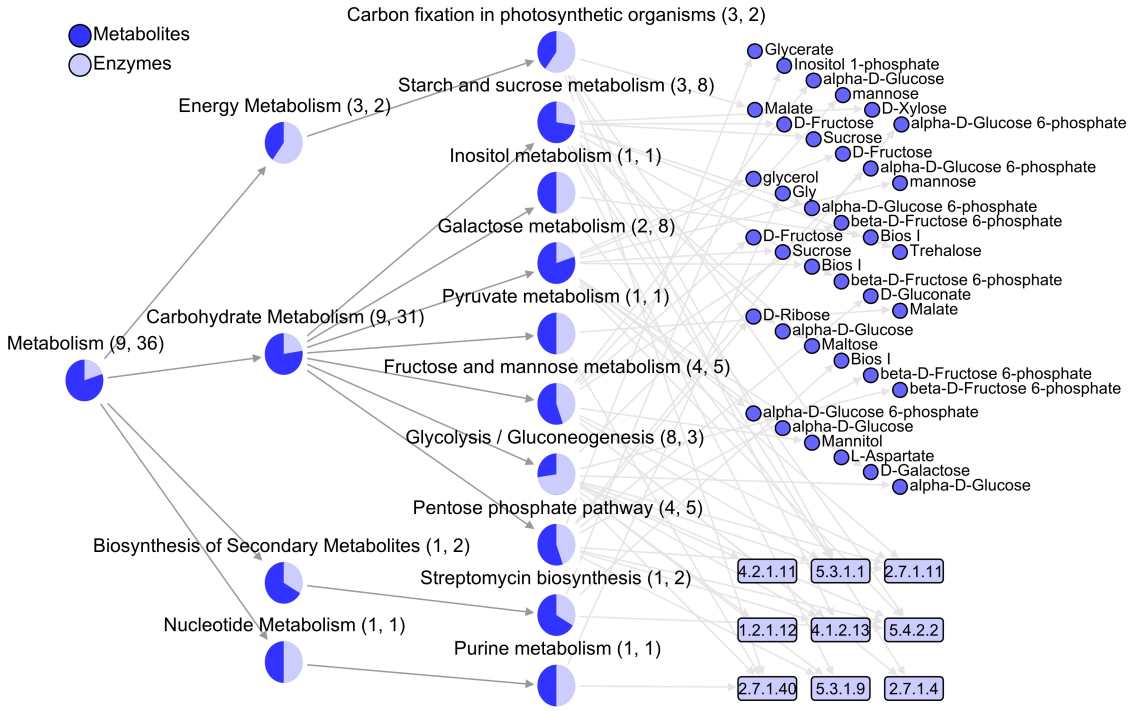

Abbildung 5.4: Datenspezifischer KEGG-BRITE-Mappinggraph für die im KEGG-System enthaltenen Metabolite (dunkelblau) und Enzyme (hellblau) des betrachteten Experimentdatensatzes (aus [124]). Berücksichtigt wurden nur Pathways, welche jeweils mindestens ein im Datensatz vermerktes Metabolit und Enzym enthalten. Kreisdiagramme zeigen die relativen Verhältnisse zur Häufigkeit der möglichen Zuordnung von Metaboliten und Enzymen zum jeweiligen Pathway oder zur Pathwaygruppe. Die Knotenlabel zeigen die Namen der für den Datensatz relevanten Pathways sowie die Anzahl der zugehörigen Enzyme und Metabolite.

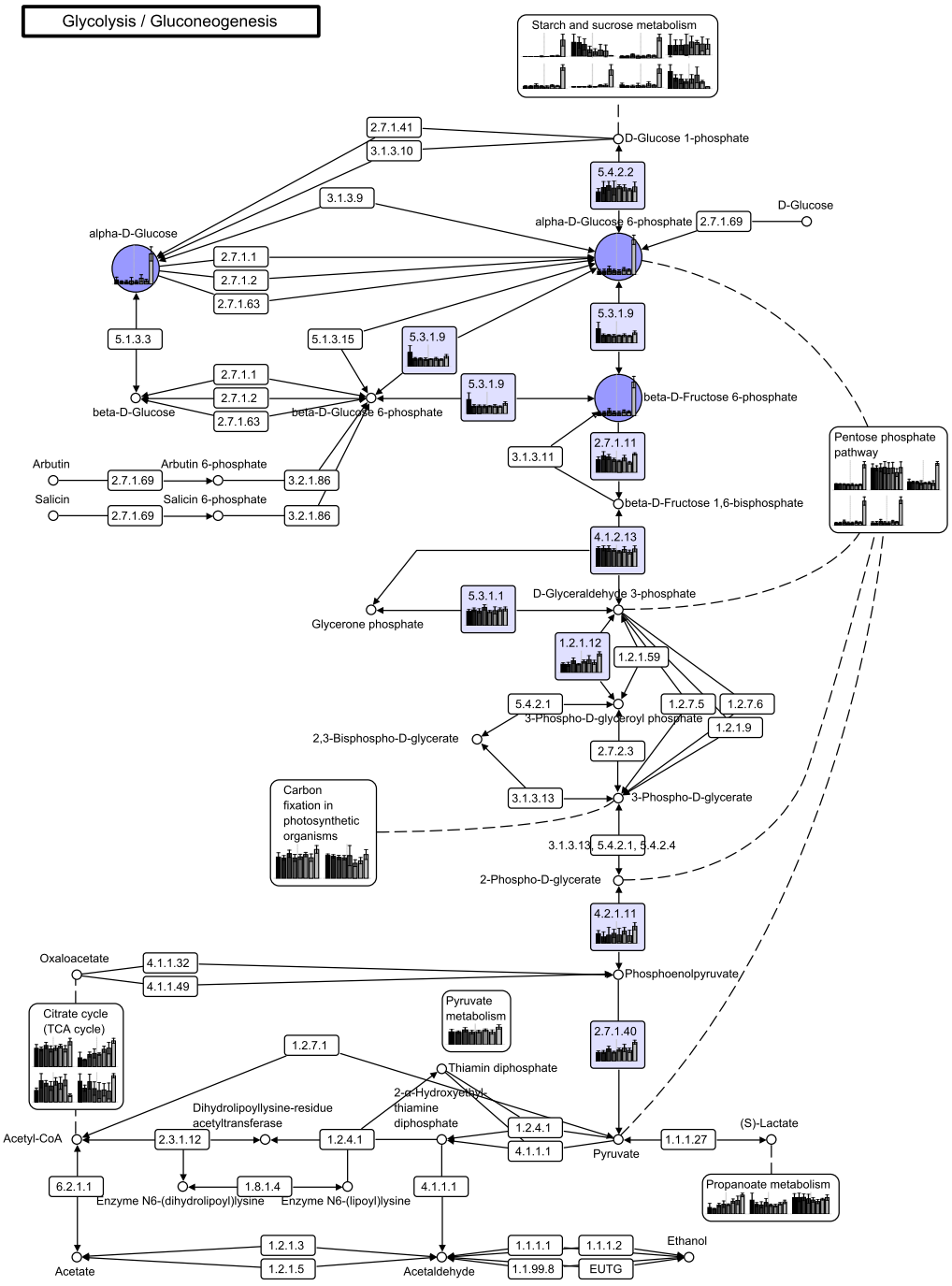

Abbildung 5.5: Mapping von Metabolit- (dunkelblaue Knoten) und Enzymdaten (hellblaue Knoten) auf den KEGG-Glykolyse-Pathway. Metabolitdaten wurden zusätzlich auf *Pathwayverweis*-Knoten gemappt (abgerundete Rechtecke). Jedes Diagramm zeigt Daten verschiedener genetischer Linien: ein Wildtyp, fünf konstitutive Linien und eine induzierbare Linie (in dieser Reihenfolge). Experimentdaten aus [124].

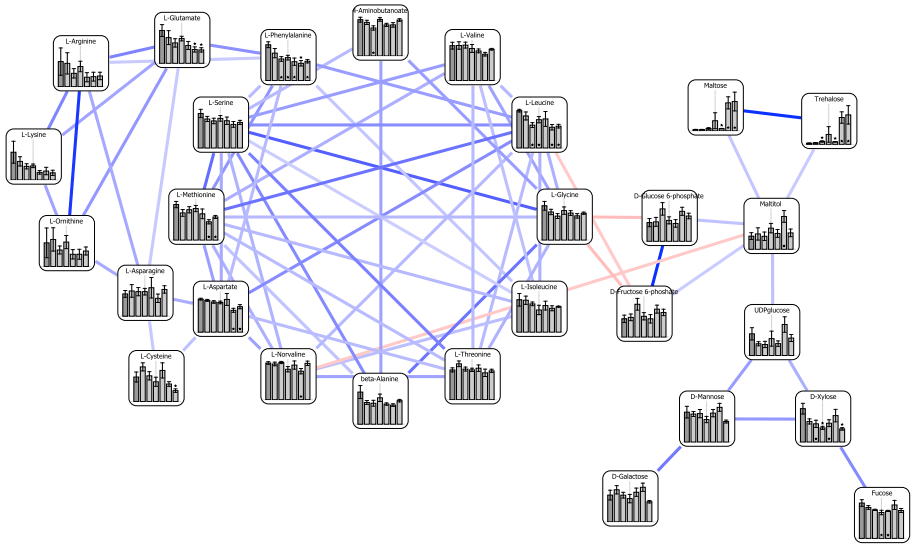

Abbildung 5.6: Visualisierung von Metabolitdaten verschiedener Kartoffelpflanzenlinien. Blaue Kanten zeigen positive, rote Kanten negative signifikante Korrelationswerte (siehe Kapitel 2.2.3, $\alpha \leq 0,1$ %). Die Intensität der Färbung ist abhängig vom ermittelten Pearson-Korrelationsfaktor. Der zentrale Teil des Graphen wurde mit einem Kreislayout bearbeitet, die verbleibenden Knoten mit einem kräftebasierten Layout.

(siehe Kapitel 2.2.3), die Ergebnisvisualisierung ist in Abbildung 5.6 dargestellt. Die stärksten Korrelationen wurden zwischen Glucose-6-Phosphat und Fructose-6-Phosphat sowie zwischen Leucin und Isoleucin beobachtet. Dies entspricht den Ergebnissen einer anderen Studie [29]. Aus Darstellung 5.6 wird ersichtlich, dass die Aminosäuren eine stark korrelierte Gruppe formen, die Zucker und Zuckerderivate hingegen nur locker verbunden sind. Diese Beobachtung stimmt mit den Ergebnissen einer weiteren Studie überein, in der gezeigt wurde, dass Glucose und Mannitol negativ zu einer Gruppe untereinander stark positiv korrelierter Aminosäuren korrelieren [125].

5.4 Visualisierung und Analyse der Stärke- und Proteinanreicherung im Gerstenkorn

Die agrarwirtschaftliche Bedeutung von Getreide basiert auf der Speicherung von Stärke und Proteinen in den Samenkörnern. Das Wachstum der Samenkörner durchläuft dabei verschiedene Phasen, welche sich im Metabolismus widerspiegeln. Die Entwicklung lässt sich dabei in Vorspeicherphase, intermediäre Phase und Speicherphase unterteilen. Während der Vorspeicherphase besteht das Korn hauptsächlich aus dem Fruchtwandmaterial. Die anschließende intermediäre Phase beginnt vier bis fünf Tage nach der Blüte, hier differenzieren sich Teile des sich entwickelnden Korns in verschiedene Gewebetypen. Dabei wächst das Gewicht und der Stärkegehalt nur

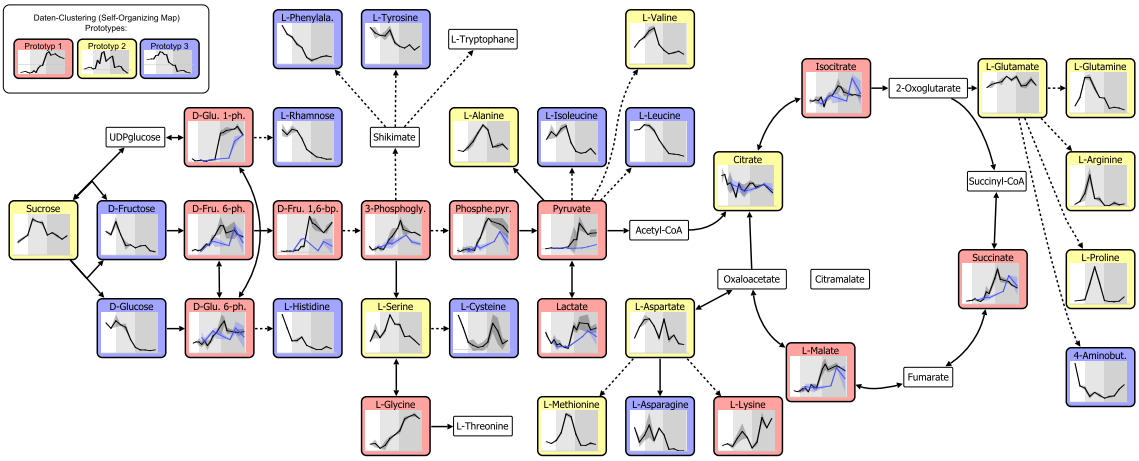

Abbildung 5.7: Metabolitanalyse des Gerstensamens (*Hordeum vulgare*), dargestellt im Kontext eines vereinfachten biochemischen Reaktionsnetzes. Die Knotenfarbe entspricht der Farbe des ähnlichsten Prototypen (links oben). Samen wurden über einen Zeitraum von 20 Tagen am Tag (schwarz) und in der Nacht (blaue Linie) geerntet und die Zusammensetzung analysiert. Der gefärbte Diagrammhintergrund untergliedert den Zeitverlauf in die Vorspeicherphase, intermediäre Phase und die Hauptspeicherphase.

wenig. Das Korn wächst dann weiter und geht zehn bis elf Tage nach der Blüte in die Hauptspeicherphase über, in der dann eine starke Erhöhung des Stärkegehalts sichtbar wird. Trotz umfangreicher Studien über Struktur, Biochemie und Genetik von sich entwickelnden Samen sind die dahinterliegenden regulatorischen Mechanismen zu einem großen Teil unbekannt [126]. Um für diese Prozesse spezifische Muster in der Entwicklung erkennen zu können, werden Zeitserienexperimente und Analysen durchgeführt. In einer Studie [127] wurden über einen Zeitraum von 20 Tagen nach der Blüte jeden zweiten Tag Gerstenkörner tagsüber und teilweise nachts geerntet und die Konzentration von ca. 40 Metaboliten untersucht. Ziel der Analyse war, ein besseres Verständnis für den Einfluss des Metabolismus auf die Stärke- und Proteinanreicherung im Samen zu entwickeln.

Mithilfe der in dieser Arbeit entwickelten Methodik ist es möglich, die erhobenen Daten und die Experimentfaktoren vollständig im Experimentdatensatz abzubilden. Die Untergliederung in Tag und Nacht wird mittels *condition*-Objekten, der Zeitverlauf mithilfe der *sample*-Zeitpunkte gespeichert. Zusätzlich wurde jede Messung doppelt ausgeführt, im Experimentdatensatz sind entsprechend mehrere *measurement*-Objekte mit unterschiedlichen Replikat-IDs vermerkt. Ein vereinfachtes biochemisches Reaktionsnetz, welches ausschließlich Metabolite enthält, dient als Basis für das Datenmapping in VANTED (siehe Abbildung 5.7). Zur Visualisierung der Daten sind Liniendiagramme gut geeignet. Die Daten von Tag und Nacht werden hier durch unterschiedlich gefärbte Linienzüge dargestellt, die Standardabweichung der Replikate ist als „Linienschatten“ unaufdringlich kenntlich gemacht. Die gleichzeitige

vergleichende Darstellung von Tag- und Nachtdaten ist nützlich, um die oft wichtigen Unterschiede leichter analysieren zu können. Die Einfärbung des Diagrammhintergrunds erlaubt die gut erkennbare Darstellung der Entwicklungsphasen und macht die nur bei starker Vergrößerung der Darstellung sinnvoll einsetzbare Achsenbeschriftung der x-Achsen für die Übersichtsdarstellung entbehrlich. Nach dem Datenmapping wurde eine automatisierte Gruppierung der Graphknoten entsprechend dem relativen Verlauf der Metabolitkonzentration über die Zeit durchgeführt. Dazu wurden sechs Neuronen einer Self-Organizing Map [100] trainiert und darauf basierend drei Gruppen gebildet. Diese zeigen einen Anstieg über die Zeit (rote Knoten in der Abbildung), eine Verminderung (blaue Knoten) oder kein signifikantes Metabolitprofil (gelbe Knoten). Es ist aus der Darstellung leicht ersichtlich, dass im Netzwerkkontext nah beieinanderliegende Knoten eher zur gleichen Gruppe gehören. Beispielsweise gehören die Hexosephosphate (Knoten „D-Glucose 6-ph.“, „D-Fructose 6-ph.“) und die weiteren Glykolyse-Intermediäre alle zur selben Gruppe mit über die Zeit ansteigenden Metabolitkonzentrationen. Dies bestätigt Untersuchungen, welche zeigten, dass an der Glykolyse beteiligte Gene ebenfalls mit dem Beginn der Speicherphase induziert werden [128].

6 Zusammenfassung und Ausblick

6.1 Zusammenfassung

Ständig weiterentwickelte Hochdurchsatzanalyseverfahren ermöglichen Forschern in den Lebenswissenschaften einen immer umfassenderen Blick auf die Biochemie der Untersuchungsobjekte. Insbesondere im Forschungsgebiet Systembiologie befinden sich keine Einzelphänomene, sondern das komplexe Wechselspiel der Ebenen Genomics, Proteomics und Metabolomics im Fokus der Untersuchung. Die vorliegende Arbeit stellte zu Beginn relevante Grundlagen zu Biologie, Statistik und Informatik vor. Darauf aufbauend wurde eine Methodik entwickelt, welche es ermöglicht, komplex strukturierte Experimentdatensätze aus den verschiedenen omics-Bereichen in einem einzigen Datenmodell abzubilden. Die Anforderungen von Experimentatoren hinsichtlich relevanter Experimentfaktoren wurden berücksichtigt. Das Konzept des Mappinggraphen erlaubt die flexible Definition unterschiedlicher biologischer Netzwerke und Klassifikationssysteme. Die Zuordnung von relevanten Teilen des Experimentdatensatzes zu Knoten und Kanten des Mappinggraphen erfolgt durch das Datenmapping. Da statische Visualisierungen einen hohen Arbeitsaufwand für Erstellung und Änderung bedürfen, bieten sie dem Anwender nur wenig Hilfestellung bei der Suche nach *a priori* unbekannten Zusammenhängen. Somit ist es erstrebenswert, flexible dynamische Visualisierungssysteme zu entwerfen. Auswahl und Design geeigneter Interaktionsmethoden, zum Beispiel für die Integration von verschiedenen Pathways und andere Navigationsmethoden, sind daher ein wichtiger Bestandteil der vorgestellten Methodik. Im interdisziplinären Feld der Bioinformatik gehört die Bereitstellung von anwendbaren Lösungen zu den wichtigen Zielen der Forschung. Die Implementation als Computerprogramm ermöglicht den praktischen Einsatz der entwickelten Methoden. Da Endanwender oft wichtige Impulse zur zielgerichteten Weiterentwicklung geben, ist die Wahl eines geeigneten Softwareentwicklungsmodells eine wichtige Voraussetzung für eine erfolgreiche Implementation. Das für diese Arbeit verwendete evolutionäre Entwicklungsmodell basiert auf der Idee, dass zu Beginn nur Kernanforderungen umgesetzt und somit frühzeitig Anwender mit in die Software- und Methodenentwicklung einbezogen werden können. Integrierte Daten, also biologische Netzwerke, Klassifikationssysteme und Experimentdaten, können in VANTED, welches die vorgestellte Methodik vollständig umsetzt, interaktiv vielfältig visualisiert und analysiert werden. Dazu sind alle für den Endanwender relevanten Funktionen über eine grafische Benutzeroberfläche leicht zugänglich. Die Kombination von Funktionen zur netzwerkintegrierten Visualisierung und Analyse von Daten unterschiedlicher omics-Bereiche, der Zugriff auf KEGG-Pathways, Gene Ontology und die flexible Visualisierung von nach Zeit, Bedingungen und Replikaten unterteilten Daten, finden sich derzeit in keinem anderen Softwaresystem und machen

VANTED zu einem nützlichen Werkzeug für Forschungsarbeiten in Biologie, Medizin und Bioinformatik. Erste Veröffentlichungen zeigen, dass auch externe Anwender das System erfolgreich für grundlegende statistische Analysen [129] und zur publikationsreifen Visualisierung von Experimentdaten verwenden können [130, 131, 132, 133].

6.2 Ausblick

Methodik und Implementation decken die wichtigsten der zu Beginn und während der Entwicklung erkannten Anforderungen ab. Natürlich sind trotzdem Erweiterungen und Verbesserungen denkbar, welche bisher noch nicht umgesetzt sind. Beispielsweise wäre es sinnvoll, die zu Beginn des Projektes entwickelte DBE-Datenbank [134], welche zur Speicherung und Verwaltung von Experimentdaten am IPK-Gatersleben etabliert wurde, enger mit dem VANTED-System zu verknüpfen. Es würde die Arbeit der Anwender erleichtern, wenn Datensätze nicht nur direkt aus der Datenbank in VANTED geladen, sondern neue Datensätze statt über eine extra Webschnittstelle auch direkt von VANTED aus in die Datenbank übertragen werden könnten. Die Vereinfachung des Datenuploads würde den Datenaustausch am IPK-Gatersleben und die Veröffentlichung der mit VANTED erstellten Visualisierungen erleichtern. Zur Analyse von Genexpressionsdaten werden häufig Statistikmethoden zur Normalisierung benötigt. Bisher muss die entsprechende Filterung und Normalisierung der Daten vor dem Import erfolgen. Es könnte für einige Anwender nützlich sein, ein Plug-in anzubieten, welches die Statistiksoftware R mit VANTED verbindet. Die für R verfügbaren, sehr umfangreichen Statistikbefehle und Bibliotheken, welche dort von einer Kommandozeile aus abrufbar sind, könnten dann für vorbereitete Anwendungsfälle benutzerfreundlich über die grafische VANTED-Oberfläche ausgeführt werden. Da das System als Open-Source-Anwendung implementiert und veröffentlicht wurde, besteht die Möglichkeit zur Umsetzung dieser und anderer Ideen über den Rahmen dieser Arbeit hinaus. Am IPK-Gatersleben und in weiteren Kooperationen wird VANTED bereits in verschiedenen Projekten durch die Entwicklung von Plug-ins erweitert [87].

Literaturverzeichnis

[1] Kitano H: **Systems Biology: A Brief Overview** 2002.

[2] Watson J, Crick F: **Genetical implications of the structure of deoxyribonucleic acid**. *Nature* 1953, **171**(4361):964–967.

[3] Sussman J, Lin D, Jiang J, Manning N, Prilusky J, Ritter O, Abola E: **Protein Data Bank (PDB): database of three-dimensional structural information of biological macromolecules**. *Acta Crystallographica Section D: Biological Crystallography* 1998, **54**(6):1078–1084.

[4] Moll A, Hildebrandt A, Lenhof H, Kohlbacher O: **BALLView: An object-oriented molecular visualization and modeling framework**. *Journal of Computer-Aided Molecular Design* 2005, **19**(11):791–800.

[5] Ashburner M, Ball C, Blake J, Botstein D, Butler H, Cherry J, Davis A, Dolinski K, Dwight S, Eppig J, et al.: **Gene ontology: tool for the unification of biology. The Gene Ontology Consortium.** *Nat Genet* 2000, **25**:25–9.

[6] Kanehisa M, Goto S, Hattori M, Aoki-Kinoshita KF, Itoh M, Kawashima S, Katayama T, Araki M, Hirakawa M: **From genomics to chemical genomics: new developments in KEGG**. *Nucleic Acids Research* 2006, **34**:D354–D357.

[7] Kanehisa M, Araki M, Goto S, Hattori M, Hirakawa M, Itoh M, Katayama T, Kawashima S, Okuda S, Tokimatsu T, et al.: **KEGG for linking genomes to life and the environment**. *Nucleic Acids Research* 2008, **36**(Database issue):D480.

[8] Schmitt H: **Die Entwicklung der Biochips-Technologie: Ein Überblick**. *http://www.bats.ch/bats/forum/01genom/biochips.php* 2001.

[9] Ito T, Chiba T, Ozawa R, Yoshida M, Hattori M, Sakaki Y: **A comprehensive two-hybrid analysis to explore the yeast protein interactome**. *Proceedings of the National Academy of Sciences* 2001, **98**(8):4569–4574.

[10] Rain J, Selig L, De Reuse H, Battaglia V, Reverdy C, Simon S, Lenzen G, Petel F, Wojcik J, Schaechter V, et al.: **The protein-protein interaction map of Helicobacter pylori**. *Nature* 2001, **409**(6817):211–215.

[11] Giot L, Bader J, Brouwer C, Chaudhuri A, Kuang B, Li Y, Hao Y, Ooi C, Godwin B, Vitols E, et al.: **A Protein Interaction Map of Drosophila melanogaster**. *Science* 2003, **302**(5651):1727–1736.

[12] Li S, Armstrong C, Bertin N, Ge H, Milstein S, Boxem M, Vidalain P, Han J, Chesneau A, Hao T, et al.: **A Map of the Interactome Network of the**

Metazoan C. elegans. *Science* 2004, **303**(5657):540–543.

[13] Rual J, Venkatesan K, Hao T, Hirozane-Kishikawa T, Dricot A, Li N, Berriz G, Gibbons F, Dreze M, Ayivi-Guedehoussou N, et al.: **Towards a proteome-scale map of the human protein-protein interaction network**. *Nature* 2005, **437**(7062):1173–1178.

[14] Florkin M, Stotz E: *Enzyme Nomenclature; Recommendations (1964) of the International Union of Biochemistry on the Nomenclature and Classification of Enzymes, Together with Their Units and the Symbols of Enzyme Kinetics.* Elsevier 1965.

[15] Bairoch A: **The ENZYME database in 2000**. *Nucleic Acids Research* 2000, **28**:304–305.

[16] Fromont-Racine M, Rain J, Legrain P: **Toward a functional analysis of the yeast genome through exhaustive two-hybrid screens**. *Nature Genetics* 1997, **16**(3):277–282.

[17] Walhout A, Sordella R, Lu X, Hartley J, Temple G, Brasch M, Thierry-Mieg N, Vidal M: **Protein Interaction Mapping in C. elegans Using Proteins Involved in Vulval Development**. *Science* 2000, **287**(5450):116–122.

[18] Reboul J, Vaglio P, Rual J, Lamesch P, Martinez M, Armstrong C, Li S, Jacotot L, Bertin N, Janky R, et al.: **C. elegans ORFeome version 1.1: experimental verification of the genome annotation and resource for proteome-scale protein expression.** *Nat Genet* 2003, **34**:35–41.

[19] Tyers M, Mann M: **From genomics to proteomics**. *Nature* 2003, **422**:193–197.

[20] Aebersold R, Mann M: **Mass spectrometry-based proteomics**. *Nature* 2003, **422**:198–207.

[21] Gasteiger E, Gattiker A, Hoogland C, Ivanyi I, Appel R, Bairoch A: **ExPASy: the proteomics server for in-depth protein knowledge and analysis**. *Nucleic Acids Research* 2003, **31**(13):3784–3788.

[22] Daviss B: **Growing pains for metabolomics**. *The Scientist (Philadelphia, PA)* 2005, **19**(8):25–28.

[23] Kanehisa M, Goto S: **KEGG: Kyoto Encyclopedia of Genes and Genomes**. *Nucleic Acids Research* 2000, **28**:27–30.

[24] Schomburg I, Chang A, Ebeling C, Gremse M, Heldt C, Huhn G, Schomburg D: **BRENDA, the enzyme database: updates and major new developments**. *Nucleic Acids Research* 2004, **32**(Database Issue):D431.

[25] Pharkya P, Nikolaev E, Maranas C: **Review of the BRENDA Database**. *Metabolic Engineering* 2003, **5**(2):71–73.

[26] Krieger CJ, Zhang P, Müller LA, Wang A, Paley S, Arnaud M, Pick J, Rhee SY, Karp PD: **MetaCyc: a multiorganism database of metabolic pathways and enzymes**. *Nucleic Acids Research* 2004, **32**:438–442.

[27] Vastrik I, D'Eustachio P, Schmidt E, Joshi-Tope G, Gopinath G, Croft D, de Bono B, Gillespie M, Jassal B, Lewis S, et al.: **Reactome: a knowledge base of biologic pathways and processes**. *Genome Biology* 2007, **8**(3):R39.

[28] **BioCarta: Charting Pathways of Life**. *http://biocarta.com/genes/* 2005.

[29] Roessner U, Luedemann A, Brust D, Fiehn O, Linke T, Willmitzer L, Fernie AR: **Metabolic profiling allows comprehensive phenotyping of genetically or environmentally modified plant systems**. *Plant Cell* 2001, **13**:11–29.

[30] Sachs L: *Angewandte Statistik*. Springer, 4th edition 1974.

[31] Donda A, Herde E, Kuhn O, Struck R (Eds): *Allgemeine Statistik*. Verlag Die Wirtschaft 1970.

[32] Bayer O, Hackel H, Pieper V, Tiedge J: *Wahrscheinlichkeitsrechnung und mathematische Statistik*. No. 17 in Mathematik für Ingenieure, Naturwissenschaftler, Ökonomen und sonstige anwendungsorientierte Berufe, Verlag Harri Deutsch 1980.

[33] Snedecor GW, Cochran WG: *Statistical Methods*. The Iowa State University Press, 8th edition 1989.

[34] David H, Hartley H, Pearson E: **The Distribution of the Ratio, in a Single Normal Sample, of Range to Standard Deviation**. *Biometrika* 1954, **41**(3/4):482–493.

[35] Grubbs F: **Procedures for Detecting Outlying Observations in Samples**. *Technometrics* 1969, **11**:1–21.

[36] Stefansky W: **Rejecting Outliers in Factorial Designs**. *Technometrics* 1972, **14**(2):469–479.

[37] Student: **The probable error of a mean**. *Biometrika* 1908, **6**:1–25.

[38] Satterthwaite FE: **An approximate distribution of estimates of variance components**. *Biom. Bull.* 1946, **2**(6):110–114.

[39] Motulsky H: *Intuitive Biostatistics*. Oxford University Press 1995.

[40] Weisstein EW: **Fisher's Exact Test (MathWorld – A Wolfram Web Resource)**. *http://mathworld.wolfram.com/FishersExactTest.html* 2008.

[41] Fisher R: **On the interpretation of χ 2 from contingency tables, and the calculation of P**. *Journal of the Royal Statistical Society* 1922, **85**:87–94.

[42] Lee A, Lewenz M, Pearson K: **On the Correlation of the Mental and Physical Characters in Man. Part II**. *Proceedings of the Royal Society of London* 1902, **71**:106–114.

[43] Spearman C: **"General Intelligence," Objectively Determined and Measured**. *The American Journal of Psychology* 1904, **15**(2):201–292.

[44] Fisher R: **Applications of 'Student's' distribution**. *Metron* 1925, **5**(3):90–104.

[45] Benjamini Y, Hochberg Y: **Controlling the false discovery rate: a practical and powerful approach to multiple testing**. *Journal of the Royal Statistical Society* 1995, **57**:289–300.

[46] Story JD: **A direct approach to false discovery rates**. *Journal of the Royal Statistical Society* 2002, **64**:479–498.

[47] Benjamini Y, Yekutieli D: **The control of the false discovery rate in multiple testing under dependency**. *The Annals of Statistics* 2001, **29**(4):1165–1186.

[48] Turau V, Vogel F: *Algorithmische Graphentheorie*. Oldenbourg Wissenschaftsverlag 2004.

[49] Wiese R, Eiglsperger M, Kaufmann M: **yFiles: Visualization and Automatic Layout of Graphs**. In *Graph Drawing: 9th International Symposium, GD 2001, Vienna, Austria, September 23-26, 2001: Revised Papers*, Springer 2002:453.

[50] **JUNG, the Java Universal Network/Graph Framework**. *http://jung.sourceforge.net* 2007.

[51] Bachmaier C, Brandenburg FJ, Forster M, Raitner M, Holleis P: **Gravisto: Graph Visualization Toolkit**. In *Proceedings of the 12th International Symposium on Graph Drawing*, Volume 3383 of *LNCS*, Springer Verlag 2004:502–503.

[52] Schumann H, Müller W: **Informationsvisualisierung: Methoden und Perspektiven (Information Visualization: Techniques and Perspectives)**. *it–Information Technology* 2004, **46**(3/2004):135–141.

[53] Di Battista G, Eades P, Tammasia R, Tollis IG: *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, New Jersey 1999.

[54] Kamada T: *Visualizing Abstract Objects and Relations*. World Scientific Publishing Co., Inc. River Edge, NJ, USA 1989.

[55] Sugiyama K: *Graph Drawing and Applications for Software and Knowledge Engineers*. World Scientific 2002.

[56] Card SK, Mackinlay JD, Shneiderman B: *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers, Inc. 1999.

[57] Shneiderman B: *The eyes have it: a task by data type taxonomy for informationvisualizations*. IEEE Computer Society Press Los Alamitos, CA, USA 1996.

[58] Furnas G: **Effective view navigation**. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM New York, NY, USA 1997:367–374.

[59] Becker R, Cleveland W: **BRUSHING SCATTERPLOTS**. *Dynamic Graphics for Statistics* 1988.

[60] McDonald J, Stuetzle W, Buja A: **Painting multiple views of complex ob-**

jects. *Proceedings of the European conference on object-oriented programming on Object-oriented programming systems, languages, and applications* 1990, **25**(10):245–257.

[61] Jelen B, Alexander M: *Pivot Table Data Crunching for Microsoft (R) Office Excel (R) 2007 (Business Solutions)*. Que Corp. Indianapolis, IN, USA 2006.

[62] Plaisant C, Carr D, Shneiderman B: **Image-Browser Taxonomy and Guidelines for Designers**. *IEEE Software* 1995, **12**(2):21–32.

[63] Warfield J: **Crossing theory and hierarchy mapping**. *IEEE Transactions on Systems, Man, and Cybernetics* 1977, **7**(7):505–523.

[64] Carpano M: **Automatic Display of Hierarchized Graphs for Computer-Aided Decision Analysis**. *IEEE TRANS. SYS., MAN, AND CYBER.* 1980, **10**(11):705–715.

[65] Sugiyama K, Tagawa S, Toda M: **Methods for Visual Understanding of Hierarchical System Structures**. *IEEE TRANS. SYS. MAN, AND CYBER.* 1981, **11**(2):109–125.

[66] Jünger M, Leipert S: **Level Planar Embedding in linear Time**. *Journal of Graph Algorithms and Applications* 2002, **6**:1.

[67] Sugiyama K, Tagawa S, Toda M: **Methods for visual understanding of hierarchical systems**. *IEEE Trans. Syst. Man Cybern.* 1981, **11**:109–125.

[68] Fruchterman T, Reingold E: **Graph drawing by force-directed placement**. *Software - Practice and Experience* 1991, **21**(11):1129–1164.

[69] Klukas C, Koschützki D, Schreiber F: **Graph Pattern Analysis with PatternGravisto**. *Journal of Graph Algorithms and Applications* 2005, **9**:19–29.

[70] Brandenburg F, Forster M, Pick A, Raitner M, Schreiber F: **Biopath–exploration and visualization of biochemical pathways**. *Graph Drawing Software, Mathematics and Visualization. Springer Verlag* 2003.

[71] Karp PD, Paley S: **Automated Drawing of Metabolic Pathways**. In *Proceedings of the Third International Conference on Bioinformatics and Genome Research*. Edited by Hunter L, Searls D, Shavlik J, AAAI Press 1994:207–215.

[72] Karp PD, Riley M, Paley SM, Pellegrini-Toole A, Krummenacker M: **The EcoCyc and MetaCyc databases**. *Nucleic Acids Res.* 2000, **28**:56–59.

[73] Karp PD, Ouzounis CA, C Moore-Kochlacs ea: **Expansion of the BioCyc collection of pathway/genome databases to 160 genomes**. *Nucleic Acids Res.* 2005, **33**(19):6083–6089.

[74] Becker MY, Rojas I: **A graph layout algorithm for drawing metabolic pathways**. *Bioinformatics* 2001, **17**(5):461–467.

[75] Wegner K, Kummer U: **A new dynamical layout algorithm for complex biochemical reaction networks.** *BMC Bioinformatics* 2005, **6**:212.

[76] Karp PD, Paley SM: **Representation of Metabolic Knowledge: Pathways**. In *Proceedings of the Second International Conference on Intelligent*

Systems for Molecular Biology. Edited by Altman R, Brutlag D, Karp PD, Lathrop R, Searls D, AAAI Press 1993:225–238.

[77] Schreiber F: **High Quality Visualization of Biochemical Pathways in BioPath**. *In Silico Biology* 2002, **2**(2):59–73.

[78] Dogrusoz U, Giral E, Cetintas A, Civril A, Demir E: **A Compound Graph Layout Algorithm for Biological Pathways**. In *Graph Drawing, New York, 2004*. Edited by Pach J, Springer 2004:pp. 442–447.

[79] Demir E, Babur O, Dogrusöz U, Gürsoy A, Nisanci G, Çetin Atalay R, Ozturk M: **PATIKA: an integrated visual environment for collaborative construction and analysis of cellular pathways**. *Bioinformatics* 2002, **18**(7):996–1003.

[80] Colantuoni C, Henry G, Zeger S, Pevsner J: **SNOMAD (Standardization and NOrmalization of MicroArray Data): web-accessible gene expression data analysis.** *Bioinformatics* 2002, **18**(11):1540–1541.

[81] Rolletschek H, Radchuk R, Klukas C, Schreiber F, Wobus U, Borisjuk L: **Evidence of a key role for photosynthetic oxygen release in oil storage in developing soybean seeds**. *The New Phytologist* 2005, **167**:777–786.

[82] Tokimatsu T, Sakurai N, Suzuki H, Ohta H, Nishitani K, Koyama T, Umezawa T, Misawa N, Saito K, Shibatanenell D: **KaPPA-View. A web-based analysis tool for integration of transcript and metabolite data on plant metabolic pathway maps**. *Plant Physiology* 2005, **138**:1289–1300.

[83] Thimm O, Blasing O, Gibon Y, Nagel A, Meyer S, Kruger P, Selbig J, Muller L, Rhee S, Stitt M: **MapMan: a user-driven tool to display genomics data sets onto diagrams of metabolic pathways and other biological processes**. *The Plant Journal* 2004, **37**(6):914–939.

[84] Rost U, Kummer U: **Visualisation of biochemical network simulations with SimWiz**. *IEE Proceedings Systems Biology* 2004, **1**:184–189.

[85] Junker BH, Klukas C, Schreiber F: **VANTED: A system for advanced data analysis and visualization in the context of biological networks**. *BMC Bioinformatics* 2005, **7**:109.

[86] Klukas C, Junker B, Schreiber F: **The VANTED software system for transcriptomics, proteomics and metabolomics analysis**. *Journal of Pesticide Science* 2006, **31**(3):289–292.

[87] Grafahrend-Belau E, Schreiber F, Koschützki D, Junker B: **Flux balance analysis of barley seeds: a computational approach to study systemic properties of central metabolism**. *Plant Physiology* 2009, **149**:585.

[88] Adler P, Reimand J, Janes J, Kolde R, Peterson H, Vilo J: **KEGGanim: pathway animations for high-throughput data**. *Bioinformatics* 2008, **24**(4):588.

[89] Oliver S: **On the MIAMI standards and central repositories, of microarray**. *Comp. Funct. Genomics* 2003, **5**:1.

[90] Taylor C, Paton N, Garwood K, Kirby P, Stead D, Yin Z, Deutsch E, Selway L, Walker J, Riba-Garcia I, et al.: **A systematic approach to modeling, capturing, and disseminating proteomics experimental data**. *Nature Biotechnology* 2003, **21**:247–254.

[91] Jenkins H, Hardy N, Beckmann M, Draper J, Smith AR, Taylor J, Fiehn O, Goodacre R, Bino RJ, Hall R, Kopka J, Lane GA, Lange BM, Liu JR, Mendes P, Nikolau BJ, Oliver SG, Paton NW, Rhee S, Roessner-Tunali U, Saito K, Smedsgaard J, Sumner LW, Wang T, Walsh S, Wurtele ES, Kell DB: **A proposed framework for the description of plant metabolomics experiments and their results**. *Nature Biotechnology* 2004, **22**(12):1601–1606.

[92] Rutkowski T: **Weiterentwicklung der FLAREX-Datenbank – Erstellung der Benutzerschnittstelle**. Studienarbeit, Fachhochschule Harz, Wernigerode 2004.

[93] Goto S, Kawashima S, Okuji Y, Kamiya T, Miyazaki S, Numata Y, Kanehisa M: **KEGG/EXPRESSION: A Database for Browsing and Analysing Microarray Expression Data**. *Genome Informatics* 2000, **11**:222–223.

[94] Reimand J, Tooming L, Peterson H, Adler P, Vilo J: **GraphWeb: mining heterogeneous biological networks for gene modules with functional significance**. *Nucleic Acids Research* 2008, **36**(Web Server issue):W452.

[95] Hermjakob H, Montecchi-Palazzi L, Lewington C, Mudali S, Kerrien S, Orchard S, Vingron M, Roechert B, Roepstorff P, Valencia A, et al.: **IntAct: an open source molecular interaction database**. *Nucleic acids research* 2004, **32**(Database Issue):D452.

[96] Klukas C, Schreiber F: **Dynamic exploration and editing of KEGG pathway diagrams**. *Bioinformatics* 2007, **23**(3):344–350.

[97] Dwyer T, Marriott K, Stuckey PJ: **Fast Node Overlap Removal - Correction**. In Kaufmann and Wagner [135] 2006:446–447.

[98] Grafahrend-Belau E, Weise S, Koschützki D, Scholz U, Junker B, Schreiber F: **MetaCrop: a detailed database of crop plant metabolism**. *Nucleic Acids Research* 2008, **36**(Database issue):D954.

[99] Misue K, Eades P, Lai W, Sugiyama K: **Layout Adjustment and the Mental Map**. *Journal of Visual Languages and Computing* 1995, **6**(2):183–210.

[100] Kohonen T: **The Self-Organizing Map**. In *Proceedings of the IEEE*, Volume 78 1990:1464–1480.

[101] Royce W: **Managing the development of large software systems: concepts and techniques**. In *Proceedings of the 9th international conference on Software Engineering*, IEEE Computer Society Press Los Alamitos, CA, USA, IEEE Computer Society Press Los Alamitos, CA, USA 1987:328–338.

[102] Gamma E, Helm R, Johnson R, Vlissides J: *Design patterns: elements of reusable object-oriented software*. Addison-Wesley Reading, MA 1995.

[103] Le Novere N, Moodie S, Sorokin A, Hucka M, Schreiber F, Demir E, Mi H, Matsuoka Y, Wegner K, Kitano H: **Systems Biology Graphical Notation: Process Diagram Level 1**. *Nature Precedings* 2008.

[104] Klukas C, Spies K, Scholz U, Lange M, Schreiber F: **Study of Tools for Pathway Visualization** 2008. [Nicht öffentliche Vergleichsstudie im Auftrag der BASF/SunGene].

[105] Longabaugh W, Davidson E, Bolouri H: **Visualization, documentation, analysis, and communication of large-scale gene regulatory networks**. *Biochim Biophys Acta* 2008.

[106] Kolpakov F: **BioUML – Framework for visual modeling and simulation biological systems**. In *Int. Conf. on Bioinformatics of Genome Regulation and Structure (BGRS)* 2002.

[107] Funahashi A, Morohashi M, Kitano H, Tanimura N: **CellDesigner: a process diagram editor for gene-regulatory and biochemical networks**. *Biosilico* 2003, **1**(5):159–162.

[108] Shannon P, Markiel A, Ozier O, Baliga N, Wang J, Ramage D, Amin N, Schwikowski B, Ideker T: **Cytoscape: a software environment for integrated models of biomolecular interaction networks**. *Genome Research* 2003, **13**:2498–2504.

[109] Kohler J, Baumbach J, Taubert J, Specht M, Skusa A, Ruegg A, Rawlings C, Verrier P, Philippi S: **Graph-based analysis and visualization of experimental results with ONDEX**. *Bioinformatics* 2006, **22**(11):1383–1390.

[110] **PathBuilder**. *http://www.ibioinformatics.org/research_tools.htm* 2009.

[111] van Iersel M, Kelder T, Pico A, Hanspers K, Coort S, Conklin B, Evelo C: **Presenting and exploring biological pathways with PathVisio**. *BMC Bioinformatics* 2008, **9**:399.

[112] **Protein Lounge Pathway Builder Tool**. *http://www.proteinlounge.com/pathwaybuilder.asp* 2009.

[113] Jirstrand M, Gunnarsson J, Johansson H: **PathwayLab-A customizable modeling and simulation tool**. In *International Conference on Systems Biology (ICSB)* 2004.

[114] Talcott C: **Symbolic modeling of signal transduction in pathway logic**. In *WSC '06: Proceedings of the 38th conference on Winter simulation*, Winter Simulation Conference 2006:1656–1665.

[115] Hu Z, Mellor J, Wu J, DeLisi C: **VisANT: an online visualization and analysis tool for biological interaction data**. *BMC Bioinformatics* 2004, **5**:17.

[116] Holford M, Li N, Nadkarni P, Zhao H: **VitaPad: visualization tools for the analysis of pathway data**. *Bioinformatics* 2005, **21**(8):1596–1602.

[117] Hucka M, Finney A, Sauro H, Bolouri H, Doyle J, Kitano H: **The ERATO Systems Biology Workbench: enabling interaction and exchange**

between software tools for computational biology. In *Pac. Symp. Biocomput*, Volume 1 2002:450–461.

[118] Kelder T, Pico A, Iersel M, Conklin B, Evelo C: **WikiPathways: Pathway Editing for the People**. In *11Th Annual International Conference on Research in Computational Biology, SanFrancisco (April 2007)* 2007.

[119] Rolletschek H, Borisjuk L, Radchuk R, Miranda M, Heim U, Wobus U, Weber H: **Seed-specific expression of a bacterial phosphoenolpyruvate carboxylase in Vicia narbonensis increases protein content and improves carbon economy**. *Plant Biotechnol. J.* 2004, **2**:211–219.

[120] Golombek S, Heim U, Horstmann C, Wobus U, Weber H: **Phosphoenolpyruvate carboxylase in developing seeds on *Vicia faba*. Gene expression and metabolic regulation**. *Planta* 1999, **208**:66–72.

[121] Grafahrend-Belau E, Junker BH, Koschützki D, Klukas C, Weise S, Scholz U, Schreiber F: **Towards Systems Biology of Developing Barley Grains: A Framework for Modeling Metabolism**. In *Proceedings of the 5th International Workshop on Computational Systems Biology, Leipzig 2008*, Volume 41 of *TICPS Series*, Tampere: Tampere International Center for Signal Processing 2008:41–44.

[122] Becker S, Feist A, Mo M, Hannum G, Palsson B, Herrgard M: **Quantitative prediction of cellular metabolism with constraint-based models: the COBRA Toolbox**. *Nature protocols* 2007, **2**(3):727–738.

[123] Geigenberger P: **Regulation of sucrose to starch conversion in growing potato tubers**. *Journal of Experimental Botany* 2003, **54**:457–465.

[124] Junker BH, Wuttke R, Tiessen A, Geigenberger P, Sonnewald U, Willmitzer L, Fernie AR: **Temporally regulated expression of a yeast invertase in potato tubers allows dissection of the complex metabolic phenotype obtained following its constitutive expression**. *Plant Molecular Biology* 2004, **56**:91–110.

[125] Weckwerth W: **Metabolomics in Systems Biology**. *Annu. Rev. Plant. Biol.* 2003, **54**:669–689.

[126] James M, Denyer K, Myers A: **Starch synthesis in the cereal endosperm**. *Current Opinion in Plant Biology* 2003, **6**:215–222.

[127] Rolletschek H, Weschke W, Weber H, Wobus U, Borisjuk L: **Energy state and its control on seed development: starch accumulation is associated with high ATP and steep oxygen gradients within barley grains**. *Journal of Experimental Botany* 2004, **55**:1351–1359.

[128] Sreenivasulu N, Altschmied L, Radchuk V, Gubatz S, Wobus U, Weschke W: **Transcript profiles and deduced changes of metabolic pathways in maternal and filial tissues of developing barley grains**. *Plant Journal* 2004, **37**:539–553.

[129] Abbasi A, Saur A, Hennig P, Tschiersch H, Hajirezaei M, Hofius D, Sonne-

wald U, Voll L: **Tocopherol deficiency in transgenic tobacco (Nicotiana tabacum L.) plants leads to accelerated senescence.** *Plant, Cell & Environment* 2008.

[130] Tognetti V, Zurbriggen M, Morandi E, Fillat M, Valle E, Hajirezaei M, Carrillo N: **Enhanced plant tolerance to iron starvation by functional substitution of chloroplast ferredoxin with a bacterial flavodoxin**. *Proceedings of the National Academy of Sciences* 2007, **104**(27):11495.

[131] Riewe D, Grosman L, Zauber H, Wucke C, Fernie A, Geigenberger P: **Metabolic and developmental adaptations of growing potato tubers in response to specific manipulations of the adenylate energy status**. *Plant Physiology* 2008, **146**(4):1579.

[132] Van Dongen J, Fröhlich A, Ramirez-Aguilar S, Schauer N, Fernie A, Erban A, Kopka J, Clark J, Langer A, Geigenberger P: **Transcript and metabolite profiling of the adaptive response to mild decreases in oxygen concentration in the roots of arabidopsis plants**. *Annals of Botany* 2008.

[133] Ahkami A, Lischewski S, Haensch K, Porfirova S, Hofmann J, Rolletschek H, Melzer M, Franken P, Hause B, Druege U, et al.: **Molecular physiology of adventitious root formation in Petunia hybrida cuttings: involvement of wound response and primary metabolism**. *New Phytologist* 2009, **181**(3):613–625.

[134] Borisjuk L, Hajirezaei MR, Klukas C, Rolletschek H, Schreiber F: **Integrating data from biological experiments into metabolic networks with the DBE information system**. *In Silico Biology* 2004, **5**(2):0011.

[135] Kaufmann M, Wagner D (Eds): *Graph Drawing, 14th International Symposium, GD 2006, Karlsruhe, Germany, September 18-20, 2006. Revised Papers*, Volume 4372 of *Lecture Notes in Computer Science*, Springer 2007.

Erklärungen

Selbständigkeitserklärung (entsp. Promotionsordnung §5, Absatz 2b): Hiermit erkläre ich, dass ich diese eingereichte Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die von mir angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Werken wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Frühere Bewerbungen um einen Doktorgrad (§5, Absatz 2c): Hiermit erkläre ich, dass ich mich bisher um keinen weiteren Doktorgrad beworben habe.

Frühere Promotionsversuche (§5, Absatz 2d): Hiermit erkläre ich, dass ich bisher keine vergeblichen Promotionsversuche unternommen habe.

.....
Unterschrift

Veröffentlichungen

(1) Borisjuk L, Hajirezaei MR, Klukas C, Rolletschek H, Schreiber F: **Integrating data from biological experiments into metabolic networks with the DBE information system.** *In Silico Biology* 2004, **5**, 0011.

(2) Rolletschek H, Radchuk R, Klukas C, Schreiber F, Borisjuk L: **Evidence of a key role for photosynthetic oxygen release in oil storage in developing soybean seeds.** *New Phytologist* 2005, **167**(3): 777–786.

(3) Klukas C, Koschützki D, Schreiber F: **Graph Pattern Analysis with PatternGravisto.** *Journal of Graph Algorithms and Applications* 2005, **9**(1):19–29.

(4) Junker BH, Klukas C, Schreiber F: **VANTED: A system for advanced data analysis and visualization in the context of biological networks.** *BMC Bioinformatics* 2006, **7**:109.

(5) Klukas C, Junker BH, Schreiber F.: **VANTED: Datenauswertung im Netzwerk-Kontext.** *GenomXpress* 2006, **1**:16–18.

(6) Klukas C, Junker BH, Schreiber F.: **The VANTED software system for transcriptomics, proteomics and metabolomics analysis.** *Journal of Pesticide Science* 2006, **31**(3): 289–292.

(7) Weise S, Grosse I, Klukas C, Koschützki D, Scholz U, Schreiber F, Junker BH: **Meta-All: a system for managing metabolic pathway information.** *BMC Bioinformatics* 2006, **7**:465.

(8) Klukas C, Schreiber S, Schwöbbermeyer H: **Coordinated Perspectives and Enhanced Force-Directed Layout for the Analysis of Network Motifs.** Proc. Asia Pacific Symp. Information Visualisation. 2006, CRPIT **60**:39–48.

(9) Klukas C, Schreiber F: **Dynamic exploration and editing of KEGG pathway diagrams.** *Bioinformatics* 2007, **23**: 344–350.

(10) Rolletschek H, Nguyen TH, Häusler RE, Rutten T, Göbel C, Feussner I, Radchuk R, Tewes A, Claus B, Klukas C, Linemann U, Weber H, Wobus U and Borisjuk L: **Antisense inhibition of the plastidial glucose-6-phosphate/phosphate translocator in Vicia seeds shifts cellular differentiation and promotes protein storage.** *The Plant Journal* 2007, **51**(3): 468–484.

(11) Grafahrend-Belau E, Junker BH, Koschützki D, Klukas C, Weise S, Scholz U, Schreiber F: **Towards Systems Biology of Developing Barley Grains: A Framework for Modeling Metabolism.** In *Proceedings of the 5th International Workshop on Computational Systems Biology*, Leipzig, 2008, **41**: 41-–44.

(12) Grafahrend-Belau E, Klukas C, Junker BH, Schreiber F: **FBA-SimVis: Interactive visualisation of constraint-based metabolic models.** *BMC Bioinformatics* 2009 (accepted).

(13) Weise S, Colmsee C, Grafahrend-Belau E, Junker BH, Klukas C, Lange M, Scholz U, Schreiber F: **An Integration and Analysis Pipeline for Systems Biology in Crop Plant Metabolism.** (accepted, DILS 2009).

(14) Grafahrend-Belau E, Junker BH, Klukas C, Koschützki D, Schreiber F and Schwöbbermeyer H: **Topology of plant metabolic networks.** In Björn H. Junker and Jörg Schwender (Eds.), *Plant Metabolic Networks*, Springer, 2008 (in press).

(15) Wheelock CE, Goto S, Yetukuri L, D'Alexandri FL, Klukas C, Schreiber F, Orešič M: **Bioinformatics strategies for the analysis of lipids.** In Donald Armstrong (Editor), *Lipidomics: Methods in Molecular Biology*, Humana Press Inc, Totowa, New Jersey, USA, 2009 (in press).