Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Diplomarbeit Nr. 3562

Skalierbare Visualisierung dynamischer Hypergraphen

Peter Körner

Studiengang:	Informatik
Prüfer/in:	Prof. Dr. Daniel Weiskopf
Betreuer/in:	Dr. Fabian Beck
Beginn am:	16. September 2013
Beendet am:	18. März 2014
CR-Nummer:	H.3.3, H.5.2, I.3.3, I.3.6, I.3.8

Kurzfassung

Für die Visualisierung von Hypergraphen stehen verschiedene Möglichkeiten zur Verfügung, wie zum Beispiel die Teilmengendarstellung und *Node-Link-Diagramme*; diese skalieren allerdings teilweise schlecht. Sollen dynamische Hypergraphen visualisiert werden, entsteht die zusätzliche Schwierigkeit, die Veränderung der Hypergraphen darzustellen. Auch hier gibt es mehrere Möglichkeiten – beispielsweise Animation und statische Visualisierung – mit ihren eigenen Vor- und Nachteilen.

In der vorliegenden Arbeit werden ein Ansatz basierend auf dem *Quad-Schema* zur Hypergraphvisualisierung und Eigenschaften der Matrixdarstellung von Graphen mit der statischen Visualisierung der Zeit (*time-to-space mapping*) kombiniert, um eine skalierbare Visualisierung zu erzeugen. Hierbei werden die Hyperkanten des dynamischen Hypergraphen als Spalten eines Gitters dargestellt, die Knoten als Zeilen. Die Zugehörigkeit eines Knotens zu einer Hyperkante wird durch ein Rechteck an der entsprechenden Gitterposition visualisiert. Über die Darstellung dieser Rechtecke können zusätzlich Gewichte und Kategorien der Hyperkanten und Knoten, sowie Gewichte der mit einer Hyperkante inzidenten Knoten, visualisiert werden. Weiterhin stehen mehrere Methoden zur Änderung der Anordnung der Hyperkanten und Knoten zur Verfügung.

Ein Visualisierungswerkzeug ermöglicht darüber hinaus die Interaktion mit der Visualisierung, um beispielsweise Elemente auszuwählen und Details zu diesen anzuzeigen. Mithilfe einer Fallstudie wird die Nützlichkeit der Visualisierung und der Interaktionsmöglichkeiten demonstriert.

Inhaltsverzeichnis

1. Einleitung

Die Informationsvisualisierung befasst sich unter anderem mit der Visualisierung von Graphen – Strukturen, die sich zur Modellierung einer Vielzahl von Sachverhalten eignen. Mit (statischen) Graphen lässt sich beispielsweise eine hierarchische Firmenstruktur ebenso modellieren wie die Abstammungsbeziehungen der Tier- und Pflanzenarten, ein Straßennetz oder die Zusammenhänge zwischen Software-Artefakten. Zur Berücksichtigung einer zeitlichen Dimension können statische Graphen zu dynamischen Graphen erweitert werden. Mit einem solchen Graphen könnte beispielsweise eine sich im Lauf der Jahre ändernde Firmenstruktur abgebildet werden.

Eine andere Erweiterungsmöglichkeit besteht darin, mehr als zwei Knoten pro Kante des Graphen zu erlauben. Die auf diese Art erweiterten Graphen werden als *Hypergraphen* bezeichnet. Mit diesen Graphen ist es möglich, einige Relationen, die bereits mit „gewöhnlichen“ Graphen modelliert werden können, auf eine natürlichere Art zu modellieren. Die Zusammenarbeit mehrerer Autoren bei der Publikation wissenschaftlicher Arbeiten kann z.B. als Graph mit maximal zwei Knoten pro Kante modelliert werden (wobei Knoten den Autoren entsprechen und gewichtete Kanten der gemeinsamen Autorschaft an Publikationen); ohne eine entsprechende Markierung der Kanten ist dann allerdings nicht genau feststellbar, welche Autoren zusammen an einer Publikation beteiligt waren. Passender und intuitiver ist die Modellierung als Hypergraph, wobei ein Autor einem Knoten entspricht und eine Publikation bzw. die Zusammenarbeit an einer Publikation einer Hyperkante. Weiterhin kann auf diesen Knoten eine Gewichtung eingeführt werden, sodass beispielsweise die Stärke der Beteiligung eines Autors an einer Publikation repräsentiert werden kann („fuzzy cluster“). Ein weiteres Anwendungsbeispiel ist die Modellierung von Transaktionen bzw. Commits in Versionskontrollsystemen bei der Softwareentwicklung. Ein Commit entspricht hierbei einer Hyperkante; die durch den Commit veränderten Dateien den Knoten der Hyperkante. Durch Gewichte dieser Knoten könnte hierbei die Anzahl der Änderungen an der jeweiligen Datei (z.B. in geänderten Zeilen, Diff-Chunks etc.) repräsentiert werden.

Werden statische Hypergraphen verwendet, sind alle Hyperkanten in diesem einen Graphen „zusammengefasst“. Durch die Erweiterung auf dynamische Hypergraphen (analog zur Erweiterung der gewöhnlichen statischen Graphen auf dynamische Graphen) ist es möglich, die zeitliche Entwicklung von Daten zu modellieren. Publikationen können beispielsweise in zeitliche Intervalle gruppiert werden, ebenso Commits (z.B. in die Zeitintervalle zwischen Veröffentlichungen eines Softwareprojekts).

Für die Visualisierung gewöhnlicher statischer bzw. dynamischer Graphen steht in der Informationsvisualisierung eine Vielzahl von Methoden zur Verfügung. Für statische Graphen sind dies unter anderem *Node-Link-Diagramme* und die Matrix-Darstellung. Die Visualisierung dynamischer Graphen bringt die Schwierigkeit mit sich, die zeitliche Dimension darzustellen. Auch hierfür gibt es unterschiedliche Ansätze, beispielsweise die Abbildung der Zeit auf die Zeit (*time-to-time mapping*)

oder in den Raum (*time-to-space mapping*). Bei der Wahl eines dieser Ansätze müssen Kompromisse eingegangen werden, beispielsweise eignet sich die Abbildung der Zeit auf die Zeit schlecht, um verschiedene Zeitschritte des Graphen zu vergleichen [BBV+12] und es wird unter Umständen die *mental map* [PHG07] des Betrachters beeinträchtigt. Die Abbildung der Zeit auf den Raum hat demgegenüber einen größeren Platzbedarf für die Visualisierung zur Folge.

Die Visualisierung statischer Hypergraphen kann unter anderem mit teilmengenbasierten oder kantenbasierten Methoden [KVKS09] geschehen; diese sind jedoch eher für kleine bzw. dünne Hypergraphen geeignet – bei dichten Hypergraphen kann starkes *overplotting* auftreten, da viele Linienkreuzungen entstehen können.

In der vorliegenden Arbeit wird ein Ansatz für die visuell skalierbare Visualisierung dynamischer Hypergraphen entwickelt, welcher auf dem *Quad-Schema* [KP94] und Eigenschaften der matrixbasierten Visualisierung gewöhnlicher Graphen basiert und zur Abbildung der zeitlichen Dimension den Raum nutzt. Dies verhindert *overplotting* und sorgt für den Erhalt oder zumindest eine weniger starke Beeinträchtigung der *mental map* des Betrachters beim Analysieren der Visualisierung.

Weiterhin können mit dem vorgestellten Ansatz sowohl Knoten- und Hyperkantengewichte als auch Gewichte der Knoten von Hyperkanten visualisiert werden. Außerdem ist die Visualisierungstechnik darauf ausgelegt, Strukturen in den visualisierten Datensätzen sichtbar zu machen, um die Lesbarkeit und Interpretierbarkeit der Visualisierung zu verbessern.

Die Visualisierungstechnik wird als interaktives Visualisierungswerkzeug implementiert; mit dessen Hilfe wird sie zum Abschluss anhand verschiedener Datensätze demonstriert und untersucht.

Gliederung

Die Arbeit ist folgendermaßen gegliedert:

Kapitel 2 – Verwandte Arbeiten stellt Arbeiten vor, die mit dieser Arbeit in Zusammenhang stehen

Kapitel 3 – Visualisierungstechnik beschreibt die Visualisierungstechnik

Kapitel 4 – Implementierung enthält Implementierungsdetails und beschreibt weitere Funktionen der implementierten Anwendung

Kapitel 5 – Fallstudie demonstriert und untersucht die Visualisierungstechnik anhand von Beispieldatensätzen

Kapitel 6 – Zusammenfassung und Ausblick fasst die Arbeit zusammen und geht auf Erweiterungs- und Verbesserungsmöglichkeiten ein.

2. Verwandte Arbeiten

Die in dieser Arbeit entwickelte Visualisierungstechnik ist an verschiedene Verfahren der Informationsvisualisierung angelehnt. Diese und weitere Verfahren werden im Folgenden vorgestellt.

Ein klassischer Ansatz für die Visualisierung gewöhnlicher **statischer Graphen** ist die Darstellung als *Node-Link-Diagramm* (*Knoten-Kanten-Diagramm*). Hierbei werden die Knoten des Graphen meist als Kreise dargestellt, die Kanten als Verbindungslinien zwischen diesen. Es können sowohl Kantengewichte (z.B. als Kantenlabels oder über die Kantendicke bzw. -farbe) als auch Knotengewichte (beispielsweise als Labels innerhalb der Kreise oder durch deren Größe bzw. Farbe) dargestellt werden. Auch die Richtung der Kanten bei gerichteten Graphen kann visualisiert werden, beispielsweise durch die Verwendung von Pfeilen statt einfachen Linien zum Verbinden der Kreise.

Abbildung 2.1 zeigt links ein *Node-Link-Diagramm* für einen ungerichteten Graphen mit fünf Knoten und vier gewichteten Kanten.

Die Anordnung der Kreise und Verbindungslinien sollte ästhetische Kriterien berücksichtigen [HMM00], wie z.B. eine gleichmäßige Verteilung der Knoten und Kanten, gleiche Kantenlängen und vor allem eine geringe Anzahl an Kantenkreuzungen. Bekannte Algorithmen für die Layoutberechnung für *Node-Link-Diagramme* sind die Algorithmen von Fruchterman/Reingold [FR91] und Kamada/Kawai [KK89]. Bei dichten Graphen kann die Darstellung als *Node-Link-Diagramm* zu starkem *overplotting* (Übereinanderzeichnen von Kanten/Knoten) und somit zu *visual clutter* [RLMJ05] führen.

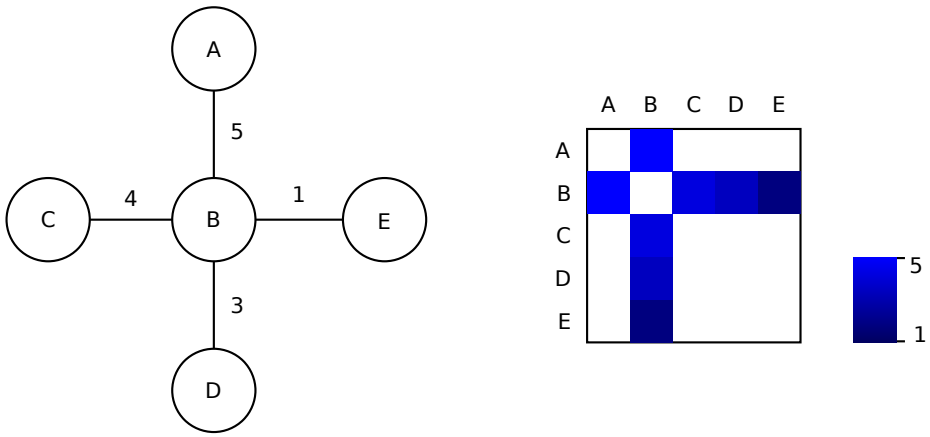


Abbildung 2.1.: Beispiel für die Visualisierung eines statischen Graphen; links: *Node-Link-Diagramm*; rechts: Matrix-Darstellung

Eine weitere Möglichkeit ist die Darstellung eines statischen Graphen in Matrixform (siehe Abbildung 2.1 rechts für den bereits beschriebenen Beispielgraphen). Hierbei werden die Knoten des Graphen am linken und oberen Rand eines Rechtecks angeordnet; ist ein Knoten X mit einem Knoten Y adjazent, wird die Zelle in der Zeile des Knotens X und Spalte des Knotens Y markiert, beispielsweise durch Einfärben. Diese Darstellung kann sowohl gerichtete als auch ungerichtete Graphen visualisieren (in diesem Fall ist die Matrix symmetrisch bzw. es würde ausreichen, nur die Hälfte der Matrix einschließlich ihrer Diagonale zu zeichnen). Die Gewichtung von Kanten kann beispielsweise über die Zellenfarbe angegeben werden, die Gewichtung von Knoten über die Knotendarstellung am Rand der Matrix (z.B. auch über die Farbe).

Diese Darstellung hat den Vorteil, dass kein *overplotting* auftreten kann; außerdem ist sie auch für große Graphen geeignet, da die Darstellungen von Knoten und Kanten (bei der Darstellung auf einem Bildschirm) ggf. bis auf „Pixelgröße“ verkleinert werden können. Nachteile sind jedoch unter anderem die schwierige Pfadverfolgung [GFC04] und ggf. verschwendeter Platz bei der Darstellung ungerichteter Graphen mit komplett gezeichneter Matrix. Außerdem muss beachtet werden, dass die Knotenanordnung einen großen Einfluss auf die Sichtbarkeit charakteristischer Strukturen in der Matrix hat [WF08] (und somit auf die Sichtbarkeit der Strukturen, wie z.B. Cliquen, des dargestellten Graphen).

Soll nicht nur ein statischer, sondern ein **dynamischer Graph** visualisiert werden, steht für die Abbildung der zeitlichen Dimension beispielsweise die Zeit (*time-to-time mapping*) oder der Raum (*time-to-space mapping*) zur Verfügung. Die Abbildung auf die Zeit verwendet Animation, um die einzelnen Zeitschritte des dynamischen Graphen nacheinander darzustellen. Hierbei tritt unter anderem das Problem auf, dass die *mental map* [PHG07] des Betrachters beeinträchtigt wird, wenn sich beispielsweise bei der Animation eines *Node-Link-Diagramms* die Position von Knoten (bzw. deren Repräsentation als Kreise) im Lauf der Animation ändert. Dieses Problem muss bei der Berechnung des Knotenlayouts (bzw. dessen Transformationen zwischen den Zeitschritten) berücksichtigt werden. Außerdem kann es für den Betrachter schwierig sein, die Struktur des Graphen für mehrere Zeitschritte zu vergleichen, da es hierfür unter anderem notwendig ist, frühere Zustände des Graphen im Gedächtnis zu behalten [BBV+12].

Diese Aufgabe kann unter Umständen leichter gelöst werden, wenn stattdessen die Abbildung der Zeit in den Raum verwendet wird. In diesem Fall werden die Zeitschritte des Graphen separat auf der Zeichenfläche abgebildet und sind somit (bei genügend großer Zeichenfläche) gleichzeitig sichtbar. Ein Nachteil dieser Darstellung gegenüber der Verwendung von Animation ist allerdings der höhere Platzbedarf der Visualisierung. Beispiele für auf diesem Ansatz basierende Visualisierungen sind *Parallel Edge Splatting* [BVB+11] und [SWS10].

Abbildung 2.2 zeigt ein Beispiel für die Visualisierung eines dynamischen Graphen mittels *Node-Link-Diagramm* und *time-to-space mapping*.

Die Visualisierung eines **statischen Hypergraphen** kann wie die Visualisierung eines statischen gewöhnlichen Graphen auf unterschiedliche Weise erfolgen. Zwei grundlegende Methoden sind teilmengenbasiert bzw. kantenbasiert [KVKS09]: Für die teilmengenbasierte Darstellung wird ein Hypergraph als ein Mengensystem aufgefasst (die Hyperkanten entsprechen Mengen von Knoten, siehe Kapitel 3); die auf der Zeichenfläche beispielsweise durch Kreise dargestellten Knoten werden für jede Hyperkante so umrandet, dass genau die Knoten der Hyperkante innerhalb der Umrandung liegen.

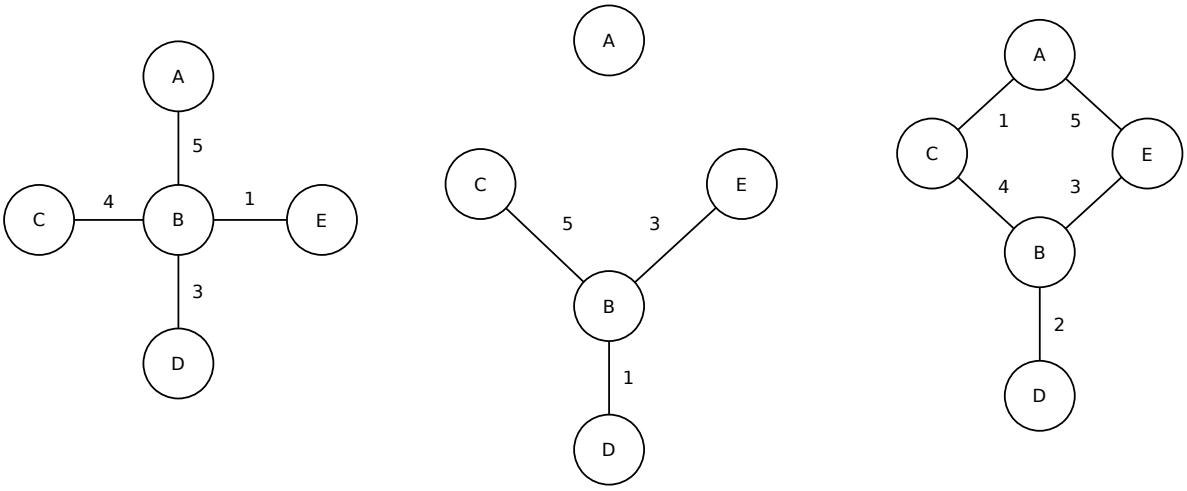


Abbildung 2.2.: Beispiel für die Visualisierung eines dynamischen Graphen (*Node-Link-Diagramm*, *time-to-space mapping*, drei Zeitschritte)

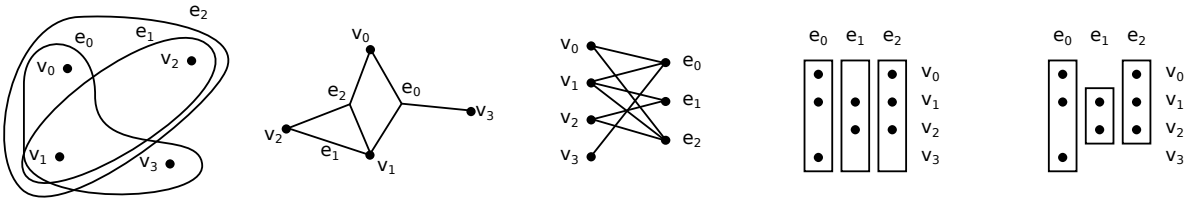


Abbildung 2.3.: Beispiel für die Visualisierung eines statischen Hypergraphen; 1. Position: teilmengenbasierte Methode; 2. Position: kantenbasierte Methode; 3. Position: Darstellung als bipartiter Graph; 4. Position: *Quad-Schema* (1. Variante); 5. Position: *Quad-Schema* (2. Variante)

Die kantenbasierte Methode verhält sich analog zu den bereits erwähnten *Node-Link-Diagrammen*: Knoten werden als Kreise gezeichnet und verbunden – hier jedoch nicht unbedingt durch einfache Linien, sondern, je nach Knotenzugehörigkeit zu den Hyperkanten, durch sich verzweigende Linien [KVKS09] bzw. Kurven [Mäk90]. Diese beiden Methoden sind in Abbildung 2.3 (1. und 2. Position) dargestellt. Die kantenbasierte Methode kann zudem so abgewandelt werden, dass die Knoten auf einer Linie angeordnet werden und die Teilkanten alle in Punkten auf einer zweiten, zur ersten Linie parallel verlaufenden, Linie zusammenlaufen [Mäk90]. Weiterhin kann ein Hypergraph als bipartiter Graph gezeichnet werden [KVKS09], wobei dessen Knotenmengen jeweils den Knoten des Hypergraphen und den Hyperkanten des Hypergraphen entsprechen; die Kanten des bipartiten Graphen repräsentieren die Inzidenz von Knoten und Hyperkanten des Hypergraphen. Die hieraus resultierende Darstellung (siehe Abbildung 2.3, 3. Position) ähnelt der oben beschriebenen Abwandlung der kantenbasierten Methode. Ein wichtiger Nachteil der beschriebenen Methoden ist das *overplotting* bei dichten Hypergraphen – dies wird teilweise schon anhand der Beispiele deutlich.

Eine Methode, die *overplotting* vermeidet, ist das *Quad-Schema* [KP94]. Hierbei werden die Knoten des Hypergraphen am Rand des Visualisierungsbereichs vertikal untereinander angeordnet; die Hyperkanten werden als Umrandungsrechtecke gezeichnet, die an der vertikalen Position jedes Knotens, mit dem die Hyperkante inzident ist, eine Markierung erhalten. Die Umrandungsrechtecke der Hyperkanten können wahlweise über die ganze Höhe der Knotenanordnung verlaufen oder nur über den Bereich, der für jede Hyperkante durch den „obersten“ und „untersten“ Knoten der Hyperkante festgelegt wird. Beide Varianten sind in Abbildung 2.3 dargestellt (4. und 5. Position).

Eine weitere Möglichkeit der Hypergraph-Visualisierung besteht darin, die Knoten des Hypergraphen kreisförmig und die Hyperkanten als Kreisabschnitte konzentrisch um diesen Kreis herum anzuordnen [KJ13]. Bei dieser Methode geht allerdings viel Platz verloren, wenn die Zeichenfläche rechteckig ist und es können nur wenige Knoten und Hyperkanten dargestellt werden.

3. Visualisierungstechnik

Die im Folgenden vorgestellte Visualisierungstechnik kombiniert das *Quad-Schema* zur Visualisierung statischer Hypergraphen mit Eigenschaften der Matrixdarstellung statischer Graphen und verwendet den Ansatz der Abbildung der Zeit in den Raum zur Visualisierung der zeitlichen Dimension des darzustellenden dynamischen Hypergraphen.

Ein Zeitschritt des visualisierten dynamischen Hypergraphen wird hierbei auf einen rechteckigen Bereich der Zeichenfläche abgebildet; zur Visualisierung mehrerer Zeitschritte werden die entsprechenden Bereiche horizontal nebeneinander (bzw. untereinander) angeordnet (*time-to-space mapping*).

In der Visualisierung werden die Hyperkanten wie beim *Quad-Schema* spaltenförmig angeordnet (sie werden allerdings nicht mit einem Rahmen umrandet), während die Knoten untereinander – zeilenförmig – angeordnet werden; die Zugehörigkeit eines Knotens zu einer Hyperkante wird ebenfalls im Schnittpunkt der entsprechenden Zeile und Spalte visualisiert. Die genaue Darstellung dieser Gitterzelle am Schnittpunkt codiert, wie bei der Matrixdarstellung gewöhnlicher Graphen, ein Gewicht – hier jedoch das Gewicht eines *Endknotens* (d.h. eines Knotens einer Hyperkante) statt des Gewichts einer Kante. Außerdem wird in Anlehnung an die Matrixdarstellung zwischen den Zeilen bzw. Spalten kein Leerraum eingefügt.

Die Abbildung einer Hyperkante auf eine Spalte kann alternativ auch als „Auseinanderrollen“ der Kreisabschnitte in [KJ13] interpretiert werden. Die dort verwendete kreisförmige Anordnung der Knoten wird hierbei in die beschriebene zeilenförmige Anordnung transformiert.

Durch die gitterförmige Anordnung und die Vermeidung größerer Zwischenräume zwischen Visualisierungselementen wird der für die Visualisierung zur Verfügung stehende Platz gut ausgenutzt. Zusammen mit der gewählten Darstellung der Knoten, Hyperkanten und Endknoten sorgt dies für eine gute Skalierbarkeit der Visualisierung. Bei ausreichend großer Zeichenfläche ist so die Visualisierung dynamischer Hypergraphen mit mehreren hundert Knoten und Hyperkanten möglich.

Im Folgenden wird zuerst das Datenmodell vorgestellt, das der Visualisierung zugrundeliegt; danach wird der Aufbau der Visualisierung beschrieben.

3.1. Datenmodell

Das grundlegende Datenmodell für die Visualisierung wurde sehr allgemein gehalten; es basiert auf dem Konzept der *(ungerichteten) Hypergraphen*. Ein solcher Graph ist definiert als ein Tupel $G = (V, E)$ bestehend aus Knotenmenge V und Hyperkantenmenge $E \subseteq \mathcal{P}(V)$.

Ein *dynamischer Hypergraph* Γ mit N Zeitschritten kann somit definiert werden als Folge von N Hypergraphen G_i, die eine gemeinsame Knotenmenge V besitzen: $\Gamma = (G_1, G_2, \ldots, G_N)$, mit $G_i = (V, E_i)$.

Durch die Definition mehrerer Funktionen wird dieser Graph zu einem *gewichteten dynamischen Hypergraphen* erweitert: Durch

$$w_v \colon V \to \mathbb{R}^+$$

wird den Knoten ein Gewicht zugeordnet, durch die Funktionen

$$w_{e,i} \colon E_i \to \mathbb{R}^+,\ i \in I$$

den Hyperkanten des i-ten Hypergraphen und durch

$$w_{n,i} \colon E_i \times V \to_p \mathbb{R}^+,\ i \in I$$

den Knoten der Hyperkanten des i-ten Hypergraphen. I bezeichnet hierbei und im Folgenden die Menge der Graph-Indizes und es gilt $I = \{1, \ldots, N\}$; 0 sei nicht in $\mathbb{R}^+$ enthalten.

Knoten-Kanten-Paaren, für die gilt, dass der jeweilige Knoten nicht mit der jeweiligen Kante inzident ist, wird durch die partiellen Funktionen $w_{n,i}$ kein Wert zugeordnet (anstatt diese Tatsache durch das Gewicht 0 auszudrücken).

Außerdem werden durch Funktionen *Labels* zugeordnet: Durch

$$l_v \colon V \to \Sigma^*$$

werden den Knoten Labels zugeordnet, durch die Funktionen

$$l_{e,i} \colon E_i \to \Sigma^*$$

den Hyperkanten des i-ten Hypergraphen und durch

$$l_g \colon I \to \Sigma^*$$

den Hypergraphen. Σ bezeichnet das Alphabet, das den Labels zugrundeliegt; leere Labels sind möglich.

Für die Sortierung von Knoten, Hyperkanten und Labels werden zudem lineare Ordnungen benötigt:

$$\leq_v \subseteq V \times V$$

auf den Knoten und

$$\leq_{e,i} \subseteq E_i \times E_i$$

auf den Hyperkanten des i-ten Hypergraphen, sowie

$$\leq_{lv} \subseteq \Sigma^* \times \Sigma^*$$

auf den Labels der Knoten und

$$\leq_{lv,i} \subseteq \Sigma^* \times \Sigma^*$$

auf den Labels der Hyperkanten des i-ten Hypergraphen.

Der Spezialfall eines *gewichteten statischen Hypergraphen* ergibt sich für den Fall $N = 1$.

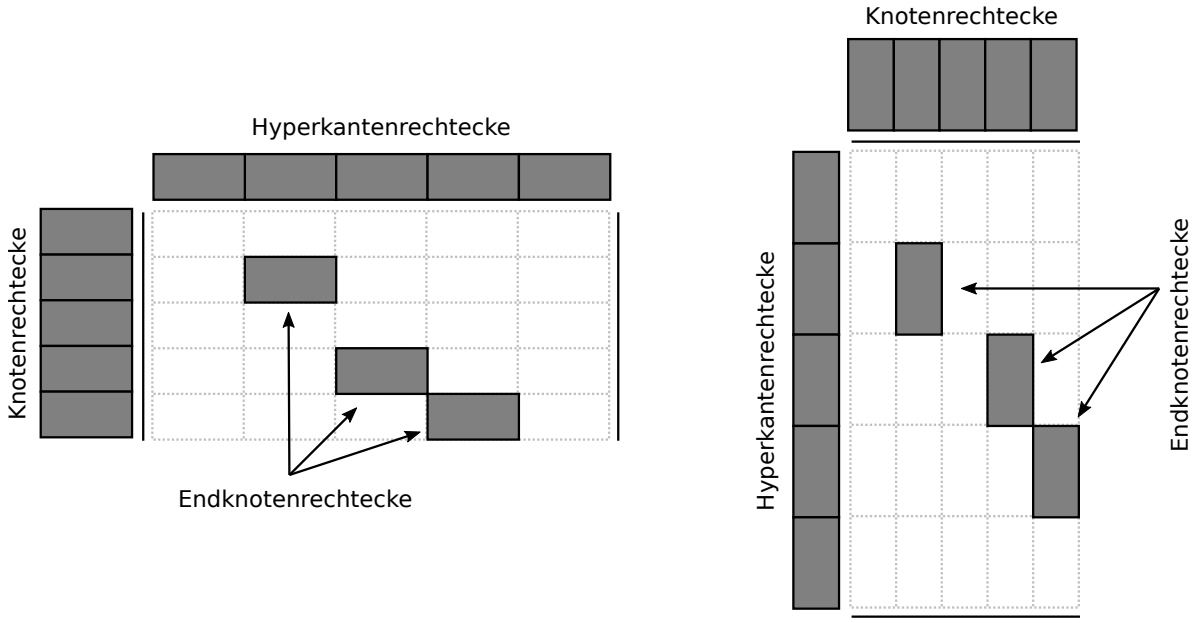


Abbildung 3.1.: Schema der normalen (links) und transponierten (rechts) Visualisierung eines statischen Hypergraphen (gestrichelte Hilfslinien sind nicht Teil der Visualisierung)

3.2. Visualisierung eines statischen Hypergraphen

Zunächst wird die Visualisierung eines statischen Hypergraphen betrachtet. Der zu visualisierende dynamische Hypergraph besteht (wie im vorherigen Abschnitt beschrieben) aus einer Folge statischer Hypergraphen – die Visualisierung des dynamischen Hypergraphen wird später ebenfalls aus den Visualisierungen der statischen Hypergraphen aufgebaut.

Ein statischer Hypergraph wird auf einen rechteckigen Bereich der Zeichenfläche abgebildet. Hierbei werden die Elemente v_k der Knotenmenge V vertikal angeordnet; für jeden Knoten wird hierbei ein Rechteck (*Knotenrechteck*) gezeichnet, welches das Gewicht des Knotens visualisiert (siehe Abbildung 3.1). Die genaue Darstellung wird im folgenden Abschnitt (3.2.1) beschrieben. Die Elemente e_m der Hyperkantenmenge E werden horizontal angeordnet; hierbei wird für jede Hyperkante ein Rechteck (*Hyperkantenrechteck*) gezeichnet, welches das Gewicht der Hyperkante visualisiert. Eine Zeile dieser gitterförmigen Anordnung, die mit einem Knotenrechteck beginnt, entspricht somit einem Knoten des Hypergraphen; eine Spalte, die mit einem Hyperkantenrechteck beginnt, einer Hyperkante. Die Knoten-/Hyperkantenrechtecke können als „Stellvertreter“ des jeweiligen Knotens bzw. der jeweiligen Hyperkante interpretiert werden – dies wird unter anderem für die Interaktion mit der Visualisierung (Kapitel 4) verwendet werden.

Die Zugehörigkeit eines Knotens v_k zu einer Hyperkante e_m (ein Endknoten) wird nun visualisiert, indem in der Gitterzelle am Schnittpunkt der entsprechenden Zeile und Kante ein Rechteck gezeichnet wird. Dieses Rechteck visualisiert zudem das Gewicht dieses Endknotens (siehe Abschnitt 3.2.1).

Am linken und rechten Rand des Endknoten-Gitters wird außerdem vom oberen bis zum unteren Rand des Gitters jeweils eine Begrenzungslinie gezeichnet.

Zur besseren Nutzung der zur Verfügung stehenden Zeichenfläche kann die Visualisierung „transponiert“ werden, d.h. es kann die Zuordnung von Knoten zu Zeilen und Hyperkanten zu Spalten vertauscht werden – Knoten werden dann Spalten zugeordnet, Hyperkanten entsprechend Zeilen (siehe Abbildung 3.1). Diese Vertauschung bietet sich beispielsweise an, wenn der zu visualisierende Hypergraph mehr Knoten als Hyperkanten besitzt und die Breite der Zeichenfläche größer ist als ihre Höhe.

3.2.1. Abbildung der Gewichte

Die Knoten-, Hyperkanten- und Endknoten-Gewichte werden durch die Darstellung der entsprechenden Knoten-, Hyperkanten- und Endknotenrechtecke visualisiert. Diese kann zudem durch Visualisierungsparameter beeinflusst werden.

Zunächst müssen sowohl Knoten- als auch Hyperkanten- und Endknotengewichte normiert werden. Das normierte Gewicht eines Knotens v wird hierbei als

$$\tilde{w}_v(v) = \frac{w_v(v)}{\max(\{w_v(v') : v' \in V\})}$$

berechnet, das normierte Gewicht einer Hyperkante e als

$$\tilde{w}_{e,1}(e) = \frac{w_{e,1}(e)}{\max(\{w_{e,1}(e') : e' \in E_1\})}$$

und das normierte Gewicht eines Endknotens (von Hyperkante e und Knoten v) als

$$\tilde{w}_{n,1}(e,\, v) = \frac{w_{n,1}(e,\, v)}{\max(\{w_{n,1}(e',\, v') : e' \in E_1 \wedge v' \in V \wedge w_{n,1}(e',\, v') \text{ definiert}\})}\,.$$

Das Gewicht eines Knotens wird also mit dem Maximalgewicht aller Knoten normiert, das Gewicht einer Hyperkante mit dem Maximalgewicht aller Hyperkanten und das Gewicht eines Endknotens mit dem Maximalgewicht aller Endknoten.

Im nachfolgenden Schritt wird das normierte Gewicht in Abhängigkeit eines *weight mapping*-Parameters logarithmisch, linear oder exponentiell transformiert. Diese Transformation wird für ein normiertes, d.h. im Intervall $[0, 1]$ liegendes, Gewicht $\tilde{w}$ berechnet als

$$\hat{w} = \log_{10}(\tilde{w} \cdot 9 + 1)$$

für die logarithmische Transformation,

$$\hat{w} = \tilde{w}$$

für die lineare Transformation und

$$\hat{w} = \frac{10^{\tilde{w}} - 1}{9}$$

für die exponentielle Transformation. Durch die Wahl der Basis 10 für die Logarithmierung bzw. die Exponentiation und entsprechende Skalierung und Verschiebung wird hierbei sichergestellt, dass das Ergebnis der Transformation wieder im Intervall $[0, 1]$ liegt. Bei Verwendung der logarithmischen Transformation werden schwächere Gewichte im Vergleich mit stärkeren Gewichten besser erkennbar; bei exponentieller Transformation stärkere Gewichte im Vergleich mit schwächeren Gewichten.

Die normierten, transformierten Gewichte können nun in die Visualisierung abgebildet werden.

Zur Abbildung der Knotengewichte werden die Knotenrechtecke horizontal im Verhältnis $\frac{a_h}{b_h} = \frac{\hat{w}}{1-\hat{w}}$ zweigeteilt (dargestellt oben links in Abbildung 3.2). Der rechte Teil wird mit der dunkleren Farbe des dem Knoten zugeordneten Farbpaares ausgefüllt, der linke optional mit der helleren Farbe (die verwendeten Farben werden in Abschnitt 3.2.3 genauer beschrieben) – im Folgenden wird das Ausfüllen mit der helleren Farbe (auch bei Hyperkanten- und Endknotenrechtecken) auch als „Zeichnen des Hintergrundrechtecks“ bezeichnet. Analog hierzu werden Hyperkantenrechtecke zur Visualisierung eines Gewichts $\hat{w}$ vertikal im Verhältnis $\frac{a_v}{b_v} = \frac{\hat{w}}{1-\hat{w}}$ aufgeteilt (oben mittig in Abbildung 3.2), der untere Bereich des Rechtecks mit der dunkleren Farbe, der obere optional mit der helleren Farbe des Farbpaares ausgefüllt, das der Hyperkante zugeordnet ist. Die Optionen zum Ausfüllen der linken bzw. oberen Rechtecke der Knoten-/Hyperkantenrechtecke sind hierbei gekoppelt, d.h. entweder wird bei beiden Arten von Rechtecken das zweite Teilrechteck ausgefüllt oder bei keiner.

Außerdem kann zum Zeichnen der Knoten- bzw. Kantenrechtecke ein Farbverlauf statt einfachen Farben verwendet werden; bei Knotenrechtecken verläuft dieser von oben (leicht hellere Farbe) nach unten (leicht dunklere Farbe), bei Hyperkantenrechtecken von links (leicht hellere Farbe) nach rechts (leicht dunklere Farbe). Dieser Farbverlauf hilft, übereinander bzw. nebeneinander platzierte Elemente besser voneinander unterscheiden zu können.

Für die Abbildung der Endknotengewichte stehen sowohl die bereits beschriebenen Aufteilungsvarianten als auch weitere zur Verfügung. Ein normiertes, transformiertes Gewicht $\hat{w}$ kann

- auf die horizontale Aufteilung (mit $\frac{a_h}{b_h} = \frac{\hat{w}}{1-\hat{w}}$)
- zentriert auf die horizontale Aufteilung (mit $\frac{a_{h1}}{b_{h1}+b_{h2}} = \frac{\hat{w}}{1-\hat{w}}$)
- auf die vertikale Aufteilung (mit $\frac{a_v}{b_v} = \frac{\hat{w}}{1-\hat{w}}$)
- zentriert auf die vertikale Aufteilung (mit $\frac{a_{v1}}{b_{v1}+b_{v2}} = \frac{\hat{w}}{1-\hat{w}}$)
- zentriert auf ein inneres Rechteck (mit $\frac{a_{i1} \cdot b_{i1}}{a_{i2} \cdot b_{i2}} = \frac{\hat{w}}{1} = \hat{w}$)

abgebildet werden (links oben/unten, mittig oben/unten, rechts in Abbildung 3.2).

Auch in diesen Fällen wird der untere/linke/rechte (in der Abbildung graue) Bereich mit der dunklen Farbe des den Endknoten zugeordneten Farbpaares ausgefüllt; der restliche Bereich des Endknotenrechtecks kann (unabhängig von den Knoten-/Hyperkantenrechtecken) mit der hellen Farbe gefüllt werden.

Die Abbildung der Endknotengewichte auf die horizontale Aufteilung bietet sich an, wenn die Breite der Endknoten größer ist als ihre Höhe; die Abbildung auf die vertikale Aufteilung, wenn ihre Höhe größer ist als die Breite. Bei Verwendung der zentrierten Varianten ergibt sich unter Umständen

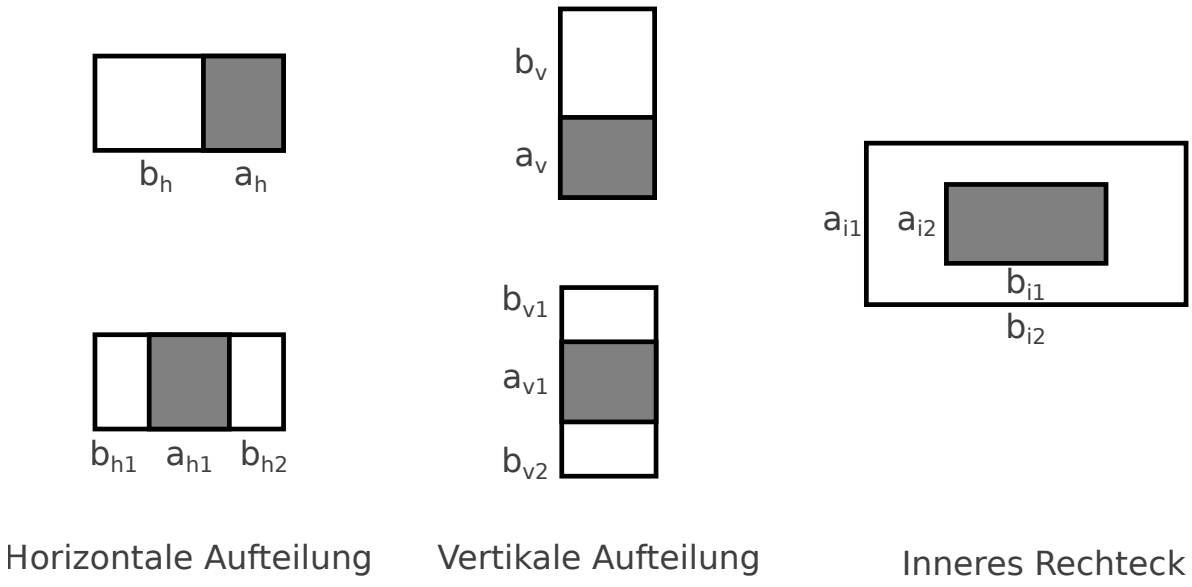


Abbildung 3.2.: Abbildung der Endknotengewichte auf Knoten-, Hyperkanten und Endknotenrechtecke

eine ansprechendere Visualisierung; ebenso bei der Abbildung der Gewichte auf innere Rechtecke. In diesem Fall könnte es jedoch schwieriger sein, Gewichte abzulesen, da sich in Abhängigkeit der Gewichte sowohl Breite als auch Höhe des inneren Rechtecks verändern.

Bei der transponierten Darstellung der Visualisierung werden alle Höhen und Breiten entsprechend vertauscht.

3.2.2. Knoten- und Hyperkantenanordnung

In Kapitel 2 wurde bereits darauf hingewiesen, dass die Anordnung der Zeilen und Spalten bei einer matrix- bzw. gitterförmigen Visualisierung entscheidend ist, um Strukturen in den visualisierten Daten erkennen zu können. Aus diesem Grund stehen für die entwickelte Visualisierungstechnik mehrere Möglichkeiten zur Umordnung der Knoten- und Hyperkantenanordnung zur Verfügung.

Die Anordnung der Knoten und Hyperkanten kann hierbei unabhängig voneinander aus verschiedenen Varianten gewählt werden.

Die erste Möglichkeit ist hierbei die „Original"-Anordnung. Hierbei werden die Knoten von oben nach unten entsprechend der Ordnung $\leq_v$ angeordnet; die Hyperkanten analog hierzu von links nach rechts entsprechend der Ordnung $\leq_{e,1}$.

Die zweite Möglichkeit ist die Anordnung basierend auf einer Sortierung. Diese Sortierung kann sowohl bei den Knoten als auch den Hyperkanten basierend auf deren Labels oder Gewichten erfolgen – für die Sortierung anhand der Labels werden die im Datenmodell eingeführten Ordnungen benötigt. Die Hyperkanten können außerdem nach der Anzahl ihrer Endknoten sortiert werden, die Knoten nach

der Anzahl der Endknoten, die sich auf den jeweiligen Knoten beziehen. Alle Sortierungsvarianten können sowohl auf- als auch absteigend angewandt werden.

Eine dritte Möglichkeit ist die Anordnung der Knoten bzw. Hyperkanten basierend auf einer durch hierarchisches Clustering erzeugten binären Clusteringhierarchie. Auf den Knoten bzw. Hyperkanten wird hierfür im ersten Schritt mithilfe eines hierarchischen Clusteringverfahrens eine solche Hierarchie aufgebaut.

Als Clusteringmethoden stehen zur Verfügung [clu]:

- *pairwise single-linkage*
- *pairwise maximum-linkage*
- *pairwise average-linkage*
- *pairwise centroid-linkage*

Als Distanzmaße jeweils [1]:

- *Pearson correlation coefficient*
- *absolute value of the Pearson correlation coefficient*
- *uncentered Pearson correlation*
- *absolute uncentered Pearson correlation*
- *Spearman's rank correlation*
- *Kendall's τ*
- *Euclidean distance*
- *city-block distance*

Für die Bestimmung einer Knotenanordnung wird im zweiten Schritt die auf den Knoten aufgebaute Clusteringhierarchie im Tiefensuchverfahren durchlaufen und aus den gefundenen Blattknoten der Hierarchie Schritt für Schritt die Knotenanordnung bestimmt.

Die Bestimmung der Hyperkantenreihenfolge läuft zunächst analog hierzu ab; nach dem Berechnen der Clusteringhierarchie (jetzt auf den Hyperkanten) wird diese jedoch – unter Ausnutzung eines Freiheitsgrades der Hierarchie – umsortiert. Hierzu wird für jeden Knoten der Hierarchie ein *Schwerpunkt* berechnet; der Schwerpunkt eines Blattknotens (d.h. einer Hyperkante e) berechnet sich als

$$b_e(e) = \frac{\sum_{v \in \{v' : w_{n,1}(e,v') \text{ definiert}\}} pos(v)}{|\{v' : w_{n,1}(e, v') \text{ definiert}\}|},$$

[1] Die letzten beiden Distanzmaße führten zu unerwartet schlechten Ergebnissen – es ist möglich, dass sie für die getesteten Anwendungsfälle weniger gut geeignet sind oder dass die Implementierung des Visualisierungswerkzeugs hier einen Fehler besitzt.

wobei $pos(v)$ die ab 1 indizierte vertikale Position des Knotens v im Endknoten-Gitter der Visualisierung angibt. Durch die bei 1 beginnende Indizierung erhalten Hyperkanten ohne Endknoten einen niedrigeren Schwerpunkt (0) als Hyperkanten mit einem einzigen Endknoten, der sich auf den obersten Knoten bezieht (Schwerpunkt 1). Der Schwerpunkt eines inneren Knotens der Hierarchie berechnet sich als das arithmetische Mittel der Schwerpunkte des linken und rechten Kindknotens.

Daraufhin werden für jeden inneren Knoten der Hierarchie die jeweiligen Kindknoten so angeordnet, dass gilt $b_l \leq b_r$, wobei b_l den Schwerpunkt des linken Kindknotens bezeichnet und b_r den des rechten Kindknotens.

Abbildung 3.3 zeigt für einen (sehr kleinen) Beispieldatensatz oben das Visualisierungsschema mit Original-Hyperkantenanordnung, in der zweiten Zeile mit Anordnung über die Clusteringhierarchie ohne Umsortierung sowie die dazugehörige Hierarchie, in der dritten Zeile die Anordnung *mit* Umsortierung der Clusteringhierarchie (und wieder die dazugehörige Hierarchie). Die berechneten Schwerpunkte wurden unter den Knoten der Hierarchie als Beschriftung hinzugefügt. Es ist zu erkennen, dass im Umsortierungsschritt die Position der Blattknoten für die Hyperkanten e_1 und e_2 vertauscht wird, da der Schwerpunkt von e_2 kleiner ist als der von e_1. Dieses Beispiel soll nur der Veranschaulichung dienen; die Auswirkung auf die Visualisierung wird vor allem bei großen Datensätzen und im Vergleich über mehrere Zeitschritte sichtbar.

In einem dritten Schritt wird dann ebenfalls die Hierarchie im Tiefensuchverfahren durchlaufen und aus den gefundenen Blattknoten der Hierarchie die Hyperkantenanordnung bestimmt.

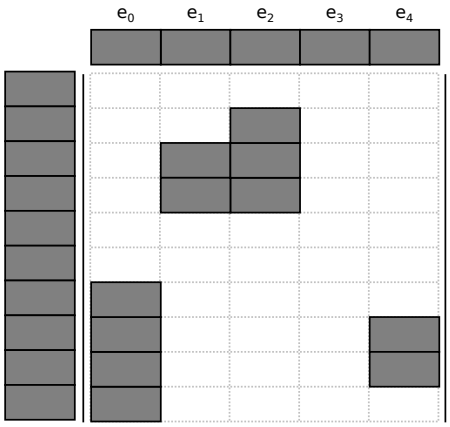
Durch die Umsortierung der Clusteringhierarchie soll erreicht werden, dass in der später erstellten Visualisierung des dynamischen Hypergraphen ähnliche Hyperkanten in unterschiedlichen Zeitschritten an ähnlichen Positionen platziert werden, um die *mental map* des Betrachters zu erhalten.

Es ist zu beachten, dass aufgrund der Umsortierung der Hyperkanten-Clusteringhierarchie die Anordnung der Hyperkanten durch die Knotenanordnung beeinflusst wird.

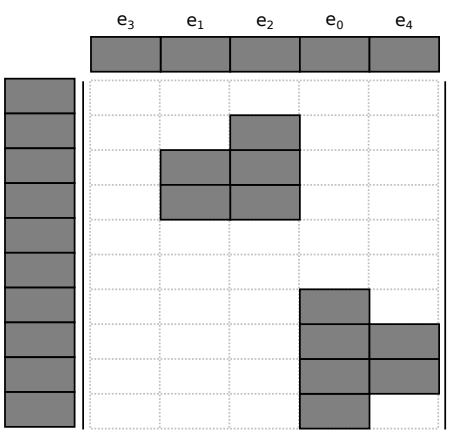
Als am besten geeignet erwiesen sich (zumindest für die getesteten teils realen, teils synthetischen Datensätze) die Clusteringmethoden *pairwise single-linkage* und *pairwise centroid-linkage* in Kombination mit den Distanzmaßen *uncentered Pearson correlation*, *Spearman's rank correlation*, und *Kendall's* τ.

3.2.3. Kategorisierung und Farbpalette

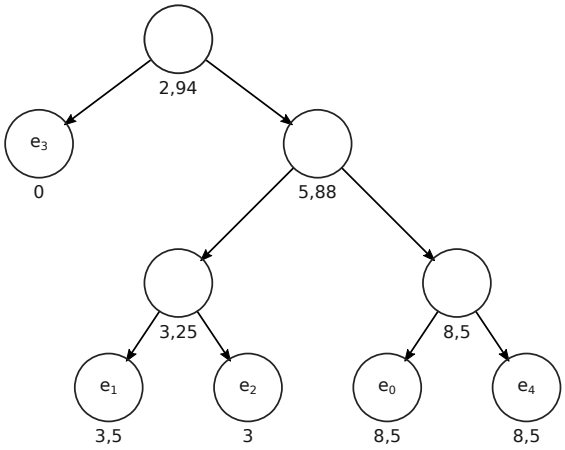
Da die Farbe der Knoten-, Hyperkanten- und Endknotenrechtecke noch nicht für die Codierung von Information verwendet wurde, kann sie zur Visualisierung der Kategorie von Knoten bzw. Hyperkanten verwendet werden. Es steht eine Farbpalette mit acht Farbpaaren (jeweils eine dunklere und eine hellere Variante einer Farbe) zur Verfügung, von denen fünf Farbpaare für die Visualisierung von Kategorien verwendet werden, ein Farbpaar für die Codierung der Zugehörigkeit zu keiner Kategorie (bei bestehender Kategorisierung) und zwei Farbpaare für die Interaktion benötigt werden (siehe Abschnitt 4.4). Falls keine Kategorisierung vorliegt, wird für alle Knoten-, Hyperkanten und Endknotenrechtecke die Farbe der ersten Kategorie verwendet.



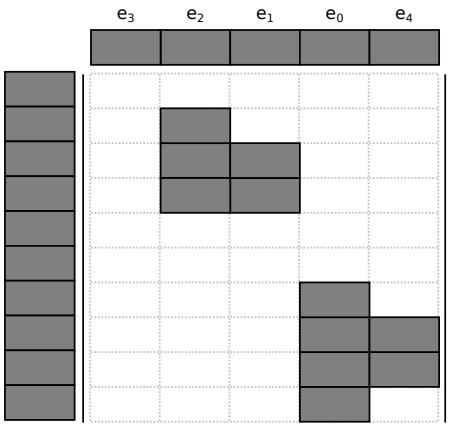
Originalanordnung



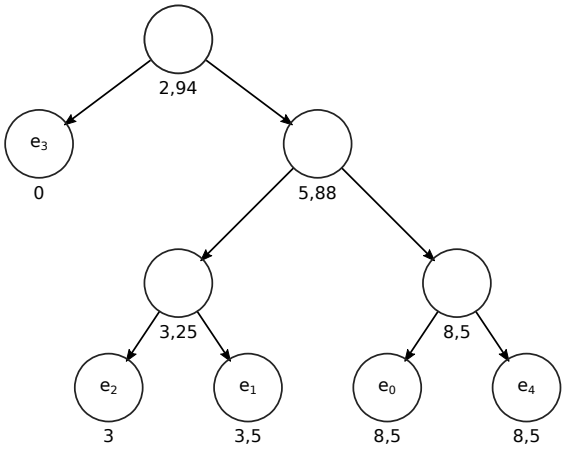
Anordnung über Clustering ohne Umsortierung der Hierarchie



Clusteringhierarchie



Anordnung über Clustering mit Umsortierung der Hierarchie



Umsortierte Clusteringhierarchie

Abbildung 3.3.: Beispiel für die Auswirkung der Umsortierung der Clusteringhierarchie auf die Hyperkantenanordnung

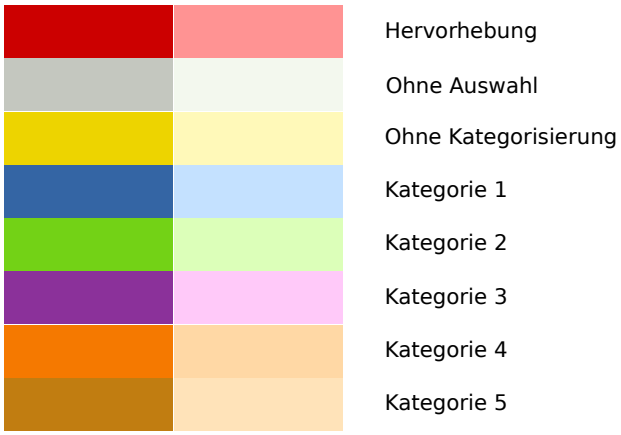


Abbildung 3.4.: Farbpalette bestehend aus acht Farbpaaren

Die Farbpalette (Abbildung 3.4) ist angelehnt an die Farben des *Tango Icon Theme* [tan], um eine ästhetisch ansprechende Visualisierung zu erhalten. Die Farben wurden, zur besseren Unterscheidbarkeit, leicht angepasst. Weitere Farbpaletten (z.B. qualitative Farbpaletten von ColorBrewer [col]) wurden getestet, führten allerdings nicht zu guten Ergebnissen oder enthielten eine zu geringe Anzahl an leicht unterscheidbaren Farben. [2]

3.3. Visualisierung eines dynamischen Hypergraphen

Nun kann die Visualisierung leicht für dynamische Hypergraphen angepasst werden. Statt einen statischen Hypergraphen auf einen rechteckigen Bereich der Zeichenfläche abzubilden, werden die statischen Hypergraphen der Zeitschritte des dynamischen Hypergraphen auf jeweils einen rechteckigen Bereich abgebildet; diese Bereiche werden nebeneinander (bzw. bei transponierter Visualisierung untereinander) angeordnet. Die Reihenfolge ist hierbei durch die Position der statischen Hypergraphen in Γ gegeben (vgl. Abschnitt 3.1, Datenmodell).

Die Knotenrechtecke werden hierbei nur einmal gezeichnet (am linken Rand der Visualisierung), Begrenzungslinien an den Rändern der Endknotengitter der einzelnen statischen Hypergraphen werden zusammengefasst, falls sie direkt benachbart sind.

Für die Normierung der Kantengewichte (siehe Abschnitt 3.2.1) wird jetzt das Maximum der Kantengewichte aller statischen Hypergraphen verwendet; für die Normierung der Endknotengewichte entsprechend das Maximalgewicht der Endknoten aller statischen Hypergraphen des dynamischen Hypergraphen.

[2] Hinweis: Beim Erstellen der Visualisierung wurden nicht-farbkalibrierte Bildschirme verwendet; der Farbeindruck kann je nach Ausgabegerät (teilweise stark) variieren.

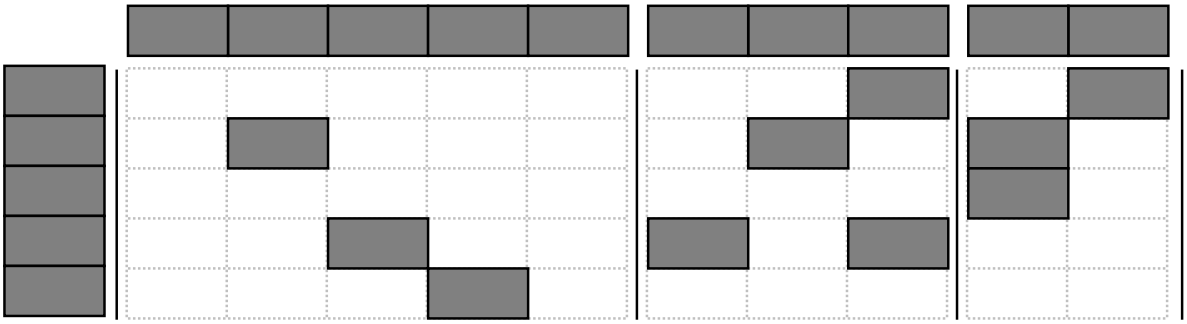


Abbildung 3.5.: Schema der Visualisierung eines dynamischen Hypergraphen (gestrichelte Hilfslinien sind nicht Teil der Visualisierung)

Die Sortierung der Knoten nach Anzahl der Endknoten berücksichtigt analog hierzu für jeden Knoten dessen zugeordnete Endknoten aller statischen Hypergraphen; ebenso das Clustering für Knoten. Für die „Original"-Anordnung der Hyperkanten werden jetzt die Ordnungen $\leq_{e,\,i}$ benötigt.

Die Anordnung der Hyperkanten wird für alle statischen Hypergraphen separat bestimmt und beeinflusst nur die Hyperkanten des jeweiligen statischen Hypergraphen; für die Anordnung mittels Clustering werden entsprechend für N statische Hypergraphen N Clusteringhierarchien (eine für jeden statischen Hypergraphen) berechnet und umsortiert.

Die restliche Erstellung der Visualisierung bleibt unverändert.

4. Implementierung

Die Implementierung als interaktives Visualisierungswerkzeug erfolgte in C++ und Qt; für das Clustering wird die „C Clustering Library“ [clu] eingesetzt.

Die graphische Benutzeroberfläche ermöglicht über Menüs und in *dock widgets* das Erstellen und Verändern der Visualisierung und die Interaktion mit dieser. Die Verwendung der von Qt bereitgestellten *dock widgets* bietet unter anderem den Vorteil, dass Teile der graphischen Oberfläche vom Hauptfenster des Programms abgelöst werden können. Dies ist vor allem bei der Verwendung mehrerer Bildschirme nützlich – das Hauptfenster mit der Visualisierung kann dann z.B. auf dem größten Bildschirm angezeigt werden, sodass für die Visualisierung möglichst viel Platz zur Verfügung steht, während einige oder alle *dock widgets* auf einem zweiten Bildschirm angezeigt werden.

Abbildung 4.1 zeigt die graphische Benutzeroberfläche (es wurde einer der Datensätze geladen, die für die Fallstudie verwendet wurden) – oben links im Programmfenster ist die Visualisierung zu erkennen, rechts befinden sich *dock widgets* mit Einstellungsmöglichkeiten, unten links eines, das in einer Tabelle die momentan ausgewählten Endknoten auflistet und unten rechts ein *dock widget* mit Informationen zum Element, auf dem sich momentan der Cursor befindet.

4.1. Dateiimport und Eingabeformat

Zum Erstellen einer Visualisierung wird über das „Datei“-Menü eine Datei im *GraphML*-Format [gra] importiert. Dieses Format ist ein XML-basiertes Format zur Speicherung von statischen (Hyper-)Graphen. Die Knoten eines Hypergraphen (*graph*-Element) werden durch *node*-Elemente repräsentiert, die Hyperkanten durch *edge*-Elemente, die *endpoint*-Elemente beinhalten, die Knoten angeben, die mit der jeweiligen Hyperkante inzident sind.

Um statt eines statischen einen dynamischen Hypergraphen zu speichern, wurde die Möglichkeit genutzt, in einer GraphML-Datei mehrere (Hyper-)Graphen zu speichern: Der dynamische Hypergraph wird, entsprechend dem Datenmodell (siehe Abschnitt 3.1), als eine Folge statischer Hypergraphen (*graph*-Elemente) gespeichert. Hierbei wird die Liste der Knoten des dynamischen Hypergraphen in jedes *graph*-Element dupliziert; auf diese Weise sollten Programme, die statische Hypergraphen aus GraphML-Dateien einlesen können, zumindest den ersten (oder sogar einen der anderen) statischen Hypergraphen laden können. Die Speicherung von Knoten-, Hyperkanten und Endknoten-Gewichten sowie von Graph-, Knoten- und Hyperkanten-Labels wurde ebenfalls standardkonform gelöst, indem das GraphML-Format mithilfe eines XML Schemas (siehe Anhang) so erweitert wurde, dass die entsprechenden Elemente ein *weight*- bzw. *label*-Attribut besitzen können. Ein Beispiel für eine solche Datei, die einen dynamischen Hypergraphen mit zwei Zeitschritten, vier Knoten, drei Hyperkanten und fünf Endknoten enthält, ist in Listing 1 zu sehen.

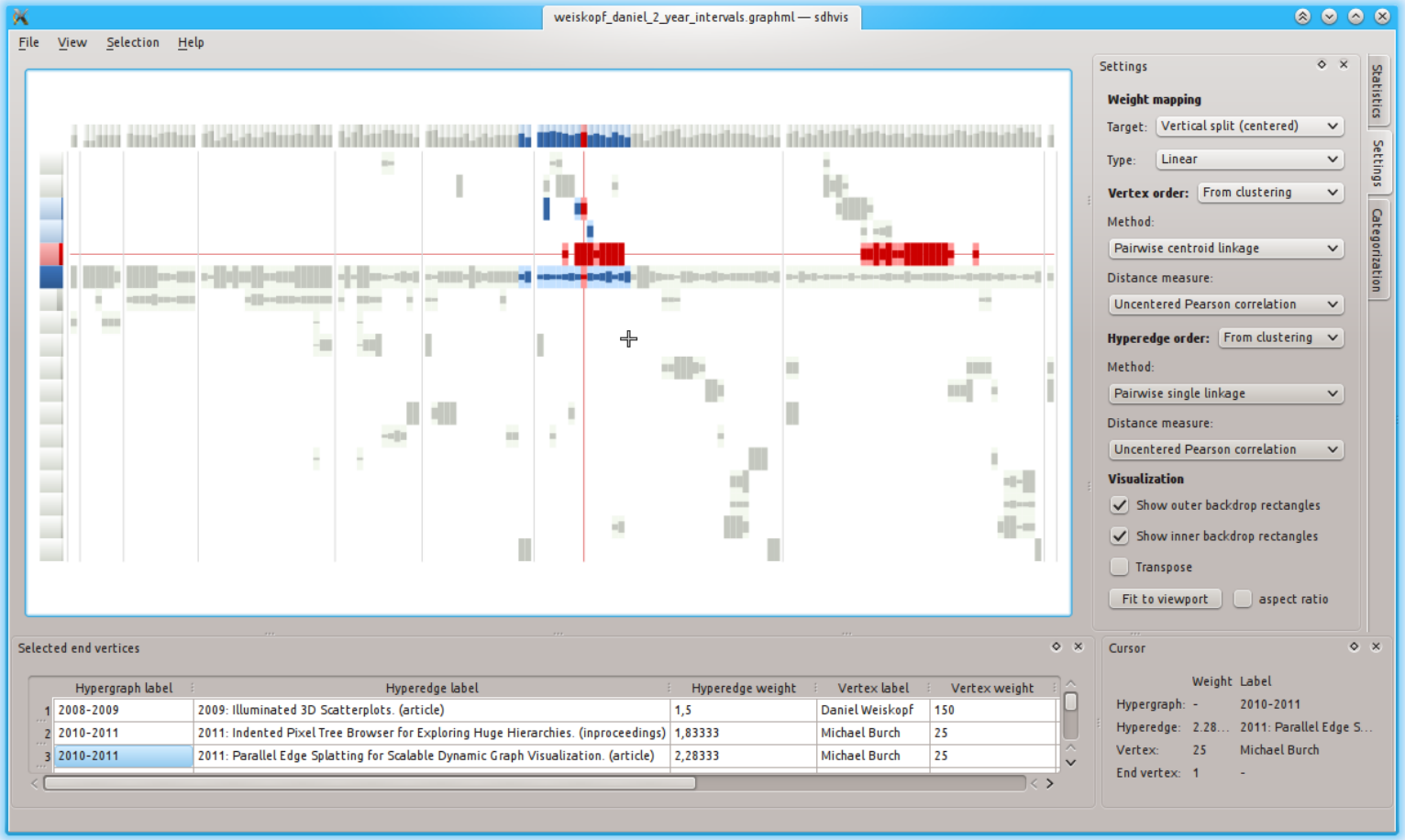


Abbildung 4.1.: Benutzeroberfläche (unter Kubuntu 13.10)

```
<?xml version="1.0" ?>
<graphml xmlns="http://graphml.graphdrawing.org/xmlns"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns graphml+weights+labels.xsd">

        <graph edgedefault="undirected" id="g0" label="hypergraph:0">
                <node id="n0" label="node:0" weight="0.0"/>
                <node id="n1" label="node:1" weight="77.0"/>
                <node id="n2" label="node:2" weight="157.0"/>
                <node id="n3" label="node:3" weight="238.0"/>
                <hyperedge label="hyperedge:0" weight="0.0">
                        <endpoint node="n0" weight="255.0"/>
                        <endpoint node="n3" weight="255.0"/>
                </hyperedge>
                <hyperedge label="hyperedge:1" weight="69.0">
                        <endpoint node="n1" weight="255.0"/>
                </hyperedge>
        </graph>

        <graph edgedefault="undirected" id="g1" label="hypergraph:1">
                <node id="n0" label="node:0" weight="0.0"/>
                <node id="n1" label="node:1" weight="77.0"/>
                <node id="n2" label="node:2" weight="157.0"/>
                <node id="n3" label="node:3" weight="238.0"/>
                <hyperedge label="hyperedge:2" weight="202.0">
                        <endpoint node="n0" weight="255.0"/>
                        <endpoint node="n2" weight="255.0"/>
                </hyperedge>
        </graph>

</graphml>
```

Listing 1: Beispiel für einen im GraphML-Format gespeicherten dynamischen Hypergraphen

Besitzt ein Element kein explizites Gewicht, wird beim Dateiimport das Gewicht 1,0 angenommen; für fehlende explizite Labels eine leere Zeichenkette.

Die im Datenmodell geforderte linearen Ordnungen auf den Knoten bzw. den Hyperkanten ergeben sich aus der Einlese-Reihenfolge der entsprechenden Elemente aus der Datei (daher auch die Bezeichnung „Original“-Anordnung).

Bei der Implementierung wurde auf leichte Erweiterbarkeit geachtet; die Unterstützung für ein zusätzliches Eingabeformat erfordert nur die Anpassung zweier Methoden zur Ergänzung der Dateierweiterung(en) für das Format sowie die Implementierung einer Methode, die den Inhalt einer solchen Datei einliest und den dynamischen Hypergraph in der internen Datenstruktur aufbaut.

Um unwichtige Elemente des Datensatzes von der Visualisierung auszuschließen, kann nach dem Importieren gefiltert werden. Hierzu wird ein Dialog angezeigt, in dem der Nutzer auswählen kann, ob entweder Knoten oder Hyperkanten gefiltert werden sollen und welcher Schwellwert der Filterung zugrunde liegen soll. Wenn Knoten gefiltert werden sollen, gibt dieser Schwellwert an, wie viele Hyperkanten mit einem Knoten mindestens inzident sein müssen (d.h. „wie viele Endknoten in Bezug auf den Knoten existieren müssen“), dass dieser Knoten nicht ausgefiltert wird; bei der Filterung von

Hyperkanten gibt er an, wie viele Endknoten eine Hyperkante mindestens besitzen muss, um nicht ausgefiltert zu werden.

Weiterhin kann vom Benutzer festgelegt werden, ob nach dem Filtern von Kanten die Hyperkanten ohne Endknoten bzw. nach dem Filtern von Hyperkanten die Knoten, die mit keiner Hyperkante inzident sind, entfernt werden sollen.

Um die geöffnete Datei neu einzulesen (und die Filtereinstellungen erneut anzuzeigen), kann diese über einen zusätzlichen Eintrag im „Datei“-Menü neu geladen werden.

In einem *dock widget* stehen grundlegende Informationen zum geladenen Datensatz zur Verfügung, wie z.B. die Anzahl der (statischen) Hypergraphen, der Hyperkanten, Knoten und Endknoten, sowie einige weitere Werte.

4.2. Erstellen der Visualisierung

Nachdem eine Datei importiert wurde, wird der eingelesene dynamische Hypergraph visualisiert. Hierfür wird Qts *Graphics View Framework* verwendet. Die Visualisierung wird – wie in Kapitel 3 beschrieben – auf einer *QGraphicsScene* erstellt und mithilfe einer *QGraphicsView* in der graphischen Benutzeroberfläche dargestellt. Die Zellen des Endknotengitters (und somit auch die Endknotenrechtecke) sind hierbei auf der Zeichenfläche (*graphics scene*) jeweils 3×3 Längeneinheiten groß; ebenso zu Beginn die Knoten- und Hyperkantenrechtecke.

Nach dem Aufbau der Visualisierung wird diese in den zur Verfügung stehenden Anzeigebereich eingepasst, wobei in x- und y-Richtung ggf. eine Streckung/Stauchung auftritt (dies betrifft alle Visualisierungselemente bis auf Begrenzungslinien zwischen den statischen Hypergraphen und die zusätzlich eingezeichneten Cursor-Linien (siehe Abschnitt 4.4, Interaktion), die durch eine spezielle Einstellung hiervon ausgenommen sind).

Während des Einpassens wird außerdem die Größe der Knoten- bzw. Hyperkantenrechtecke so angepasst, dass diese im Anzeigebereich eine Breite bzw. Höhe von ca. 20 Pixeln besitzen.

Das Einpassen kann vom Benutzer über eine Schaltfläche in der Benutzeroberfläche auf Wunsch auch später nochmals durchgeführt werden.

Weiterhin kann eingestellt werden, ob beim Einpassen das Seitenverhältnis der Elemente der *graphics scene* beibehalten (d.h. in der x- und y-Richtung gleich skaliert) werden soll. Das Aktivieren dieser Option hat dementsprechend zur Folge, dass Endknotenrechtecke als Quadrate gezeichnet werden. Diese Einstellung ergibt zusammen mit der Abbildung der Endknotengewichte auf innere Rechtecke (siehe Abschnitt 3.2.1) und ausreichender Größe des Anzeigebereichs eine ästhetisch besonders ansprechende Visualisierung; sie eignet sich vor allem dann, wenn das Verhältnis der Anzahl der Hyperkanten zur Anzahl der Knoten in etwa dem Verhältnis von Breite zu Höhe (bzw. Höhe zu Breite im transponierten Fall) des Anzeigebereichs entspricht.

Wie in Kapitel 3 bereits beschrieben, kann die Darstellung der Knoten- und Hyperkantenrechtecke einen Farbverlauf beinhalten; in Abhängigkeit der Anzeigehöhe der Knotenrechtecke bzw. Anzeigebreite der Hyperkantenrechtecke wird dieser aktiviert bzw. deaktiviert.

4.3. Visualisierungsparameter

Über die Benutzeroberfläche können die weiteren Visualisierungsparameter angepasst werden. Die wichtigsten sind hierbei die Art der Abbildung der Endknotengewichte, die Art der Transformation aller Gewichte (siehe Abschnitt 3.2.1) und die Art der Knoten- und Hyperkantenanordnungen (jeweils „Original"-Anordnung, Anordnung basierend auf Sortierung nach Label/Gewicht/Anzahl der Endknoten und Anordnung basierend auf hierarchischem Clustering (mit den in Abschnitt 3.2.2 bereits erwähnten Methoden und Distanzmaßen)).

Bei der Änderung der Anordnung der Knoten muss wie bereits erwähnt beachtet werden, dass die Anordnung der Hyperkanten ebenfalls neu berechnet werden muss; dies wird durch die Implementierung sichergestellt.

Falls eine Kategorisierung der Knoten bzw. Hyperkanten erfolgen soll, muss ein regulärer Ausdruck angegeben werden, der genau eine *capturing group* enthält. Mithilfe dieses regulären Ausdrucks wird aus den Labels der Knoten bzw. Hyperkanten jeweils die Kategorie des Knoten/der Hyperkante extrahiert. Passt der reguläre Ausdruck auf ein Label, wird der Inhalt der *capturing group* dem Knoten bzw. der Hyperkante als Kategorie zugewiesen, ansonsten eine leere Zeichenkette. Die extrahierten Kategorien werden daraufhin nach der Häufigkeit ihres Vorkommens sortiert; die vier häufigsten werden für die Kategorisierung direkt übernommen, die restlichen Kategorien werden zu einer „Rest"-Kategorie zusammengefasst. In der Benutzeroberfläche stehen drei Vorgaben zur Auswahl:

- Kategorisierung nach Teilzeichenkette in der letzten Klammer des Labels
- Kategorisierung nach Teilzeichenkette vor dem ersten Doppelpunkt im Label
- Kategorisierung nach (einfacher) Dateierweiterung

Alternativ kann auch ein benutzerdefinierter regulärer Ausdruck angegeben werden.

Den (maximal fünf) Kategorien, und somit den Knoten und Endknoten bzw. den Hyperkanten und Endknoten, werden daraufhin die fünf Kategorie-Farbpaare der aktuellen Farbpalette zugewiesen; den im Fall der Kategorisierung von Knoten nicht kategorisierten Hyperkanten wird außerdem das Farbpaar mit der Bedeutung „nicht kategorisiert" zugewiesen, bei der Kategorisierung von Hyperkanten entsprechend den Knoten.

In der graphischen Oberfläche wird eine Liste der Kategorien mit den zugeordneten Farben und der Häufigkeit des Vorkommens angezeigt; außerdem kann die Kategorisierung durch eine Schaltfläche wieder gelöscht werden.

Für die Kategorisierung steht eine vordefinierte Farbpalette zur Verfügung; auch hier kann allerdings leicht eine Erweiterung des Programms stattfinden und es können sowohl Paletten aus vordefinierten Farben als auch anhand der maximalen Anzahl von Kategorien berechnete Paletten ergänzt werden (dieser Wert wird allerdings durch eine Konstante im Quellcode festgelegt, d.h. die Farbpaletten können nicht anhand der Anzahl der tatsächlich gefundenen Kategorien berechnet werden).

Weiterhin kann in der Benutzeroberfläche eingestellt werden, ob – wie in Abschnitt 3.2.1 beschrieben – der zweite Teil der Knoten- und Hyperkantenrechtecke bzw. der Endknotenrechtecke eingefärbt

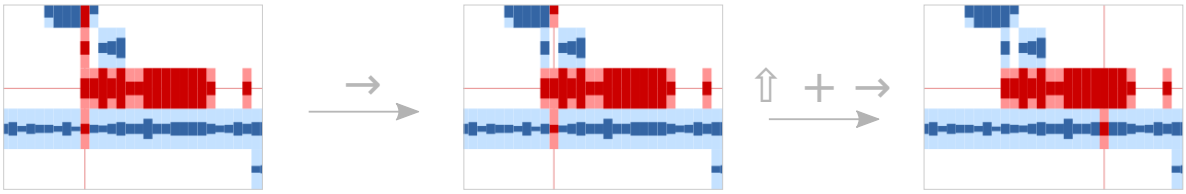


Abbildung 4.2.: Cursorbewegung um eine und zehn Positionen

werden soll und es kann zwischen der nicht transponierten und transponierten Variante der Visualisierung umgeschaltet werden. Für letztere wird die Visualisierung um die linke obere Ecke des Endknotengitters (den Ursprung des *scene*-Koordinatensystems) um 90° im Uhrzeigersinn rotiert und danach an der y-Achse gespiegelt.

4.4. Interaktion

Die statische Visualisierung wird durch verschiedene Interaktionsmöglichkeiten erweitert. Zunächst kann der Nutzer in der Standardansicht der Visualisierung eine Übersicht über den Datensatz gewinnen. Soll ein bestimmtes Element näher betrachtet werden, kann durch gedrückt halten der Umschalttaste und einen Linksklick auf ein Knoten-, Hyperkanten- oder Endknotenrechteck der *Cursor* auf dieses Element gesetzt werden. Daraufhin werden Informationen zum Element in einem *dock widget* angezeigt (siehe Abbildung 4.1 rechts unten). Durch Drücken der Pfeiltasten kann der Cursor horizontal (Pfeiltaste links/rechts) oder vertikal (Pfeiltaste hoch/runter) bewegt werden, sofern in der jeweiligen Richtung ein weiteres Element (Knoten-, Hyperkanten- oder Endknotenrechteck) existiert. Durch gleichzeitiges gedrückt halten der Umschalttaste kann in Zehnerschritten weitergesprungen werden. Die Visualisierungselemente der Spalte und Zeile, in der sich der Cursor befindet, werden mithilfe des Hervorhebungs-Farbpaares hervorgehoben; zusätzlich werden im Hintergrund der Visualisierung Führungslinien angezeigt. Soll der Cursor wieder gelöscht werden, kann bei gedrückter Umschalttaste in einen Bereich der Visualisierung ohne Knoten-, Hyperkanten- und Endknotenrechteck linksgeklickt werden. Abbildung 4.2 zeigt die Cursorbewegung an einem Beispiel: Der Cursor wird zuerst mit „Pfeil rechts“ um ein Element nach rechts bewegt, dann durch zusätzliche Verwendung der Umschalttaste um zehn Elemente.

Sind mehrere Elemente von Interesse, kann eine *Auswahl* erstellt werden. Dies kann auf mehrere Arten geschehen. Falls der Cursor sich auf dem auszuwählenden Element befindet, kann über das „Selection“-Menü oder über die Tastenkürzel Strg + S bzw. Strg + X eine neue Auswahl angelegt werden (hierbei wird ggf. eine bestehende Auswahl gelöscht) bzw. die bestehende Auswahl erweitert werden. Befindet sich der Cursor hierbei auf einem Knotenrechteck, werden alle Endknoten die sich auf diesem Knoten beziehen, in die Auswahl aufgenommen bzw. ergänzt; befindet er sich auf einem Hyperkantenrechteck entsprechend alle Endknoten der Hyperkante. Befindet er sich auf einem Endknoten, wird nur dieser für die (Ergänzung der) Auswahl berücksichtigt. Weiterhin kann die Auswahl durch einfaches Linksklicken auf ein Visualisierungselement ersetzt werden und durch Linksklicken auf ein Visualisierungselement bei gedrückt gehaltener Steuerungstaste ergänzt werden.

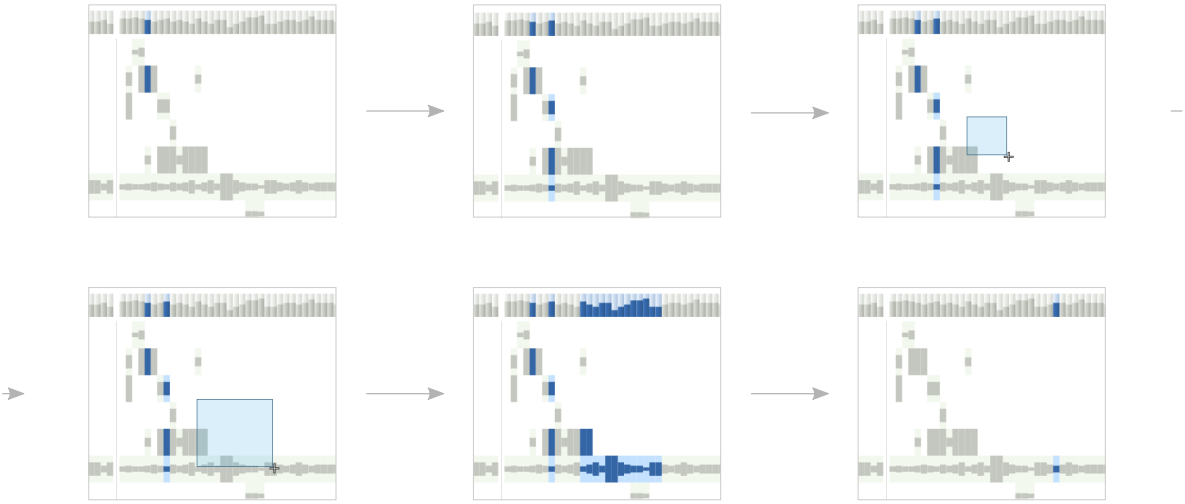


Abbildung 4.3.: Festlegen, Erweitern und Ersetzen der Auswahl

Hierbei bewirkt eine Auswahl eines Knotenrechtecks bzw. Hyperkantenrechtecke ebenfalls die Auswahl der ganzen Zeile bzw. Spalte.

Schließlich gibt es die Möglichkeit, durch Rechtsklicken mit der Maus und Aufziehen eines Auswahlrahmens mehrere Elemente für die Änderung der Auswahl zu berücksichtigen. Auch hier wird in Abhängigkeit der Steuerungstaste die Auswahl ersetzt oder erweitert, wobei jedes Element, das sich ganz oder teilweise im Auswahlrahmen befindet, gesondert betrachtet wird und ggf. für die Auswahl einer ganzen Zeile/Spalte sorgt. Abbildung 4.3 zeigt das Festlegen einer Auswahl durch Klicken auf ein Endknotenrechteck, Erweitern der Auswahl zuerst durch Klick auf ein Hyperkantenrechteck, danach durch Aufziehen eines Auswahlrechtecks und abschließendes Ersetzen der Auswahl durch Klick auf ein Endknotenrechteck.

Existiert eine Auswahl, wird allen nicht ausgewählten Elementen das Farbpaar für nicht ausgewählte Elemente zugewiesen; mit der vorgegebenen Farbpalette werden diese so „ausgegraut“. Zeilen- und Spaltenköpfe von Zeilen bzw. Spalten, die ausgewählte Elemente enthalten, werden mit ihren Originalfarben gezeichnet. So kann schnell ein Überblick gewonnen werden, in welchen Zeilen bzw. Spalten eine Auswahl existiert.

Für die detaillierte Untersuchung werden die Informationen der ausgewählten Endknoten zeilenweise in einer Tabelle angezeigt (links unten in Abbildung 4.1). Dies umfasst Hypergraphlabel, Hyperkantenlabel und -gewicht, Knotenlabel und -gewicht sowie Endknotengewicht). Diese Tabelle ist nach Spalten sortierbar; außerdem sind die Werte der Zellen kopierbar um z.B. ein langes Label direkt in ein anderes Programm übertragen zu können.

Wird auf eine Zelle in der Tabelle geklickt, wird der Cursor auf das entsprechende Visualisierungselement gesetzt; ebenso wird beim Bewegen des Cursors die dem Element unter dem Cursor entsprechende Tabellenzeile markiert, falls sich der Cursor auf einem ausgewählten Element befindet (ansonsten wird eine eventuell bestehende Markierung in der Tabelle gelöscht). Abbildung 4.4 zeigt die Auswirkung einer Cursorbewegung nach rechts auf die Tabelle.

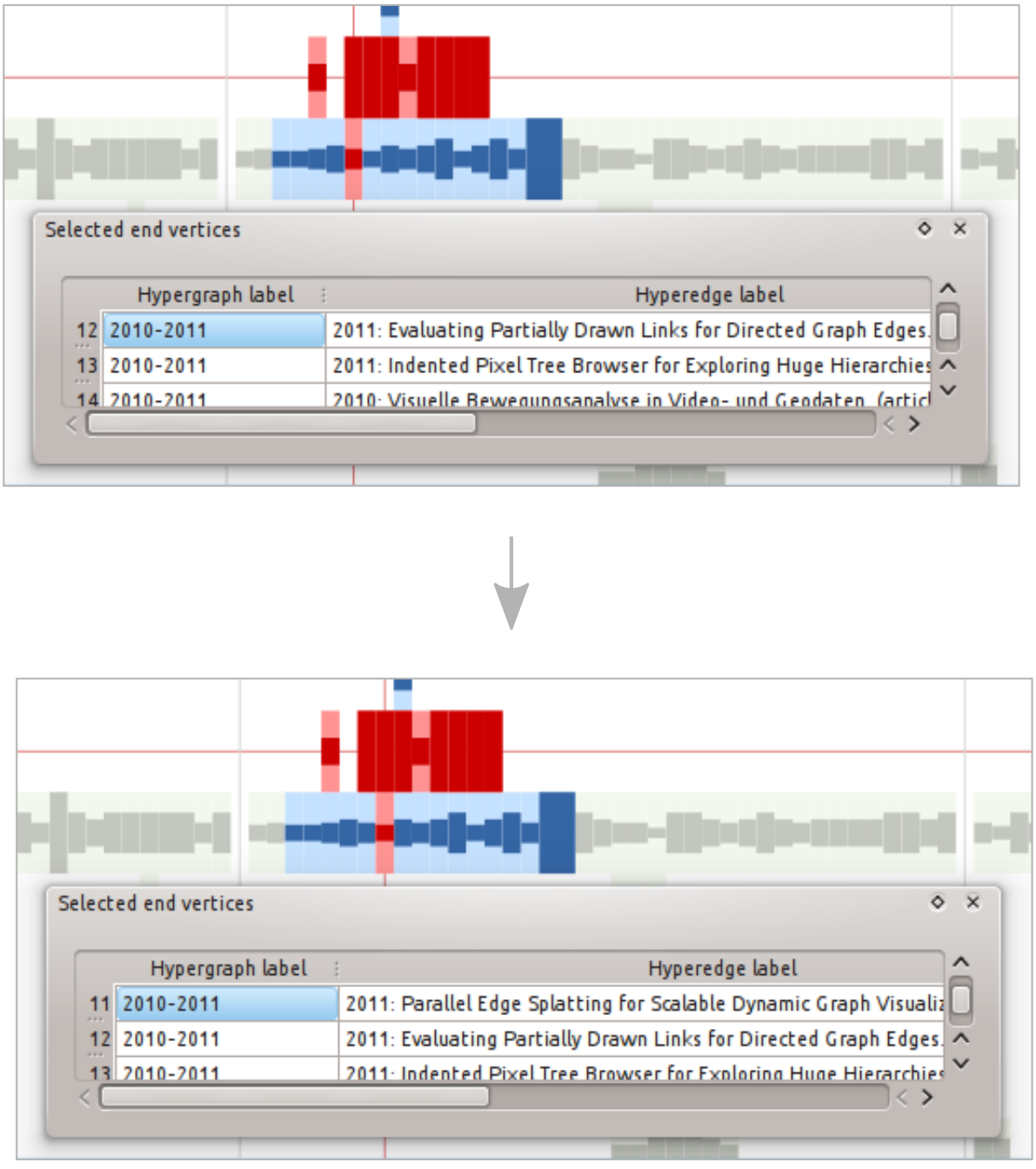


Abbildung 4.4.: Synchronisation von Cursor und markierter Tabellenzeile

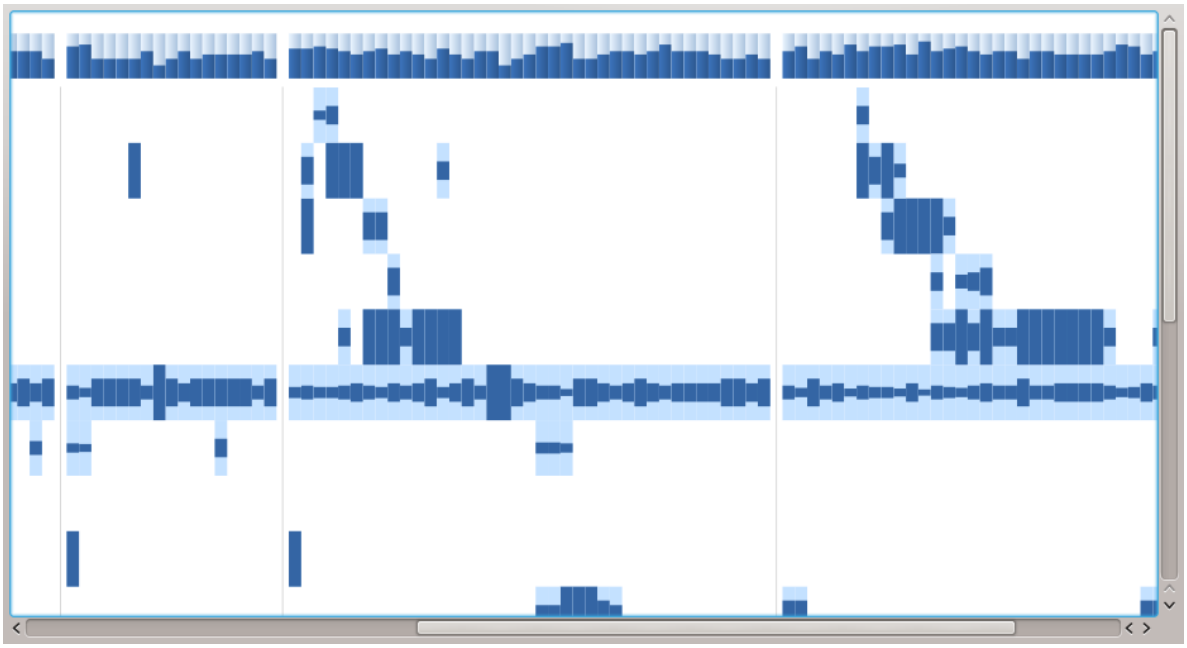


Abbildung 4.5.: Geometrisches Zoomen/Verschieben

Durch die Zeilennummerierung in der Tabelle kann schnell bestimmt werden, wie viele Elemente ausgewählt sind.

Wird die Art der Knoten- bzw. Hyperkantenanordnung geändert, bleibt die Auswahl trotzdem bestehen und auch der Cursor befindet sich nach der Umordnung auf dem Element, auf dem er sich davor befand. Auf diese Weise kann z.B. untersucht werden, wie die Änderung der Art der Anordnung die Position einzelner bzw. mehrerer Elemente beeinflusst.

Soll ein Teil der Visualisierung vergrößert dargestellt werden, kann geometrisches Zoomen [Spe07] angewandt werden. Dieses ist über das „View“-Menü bzw. die Tastenkürzel Strg + + zum Vergrößern bzw. Strg + - zum Verkleinern verfügbar. Hierbei wird durch Scrollbalken angezeigt, welcher Ausschnitt der Visualisierung gerade sichtbar ist; das Verschieben dieses Ausschnitts ist über das Mausrad (vertikal und horizontal) bzw. über die Scrollbalken möglich. Außerdem wird der sichtbare Bereich automatisch verschoben, wenn der Cursor auf ein nicht sichtbares Element bewegt wird.

Eine weitere Möglichkeit, Details zu Elementen der Visualisierung zu erhalten, ist die Nutzung ihrer *tool tips* (siehe Abbildung 4.6). Sowohl Knoten- als auch Hyperkanten- und Endknotenrechtecke besitzen diese; je nach Element enthalten sie unterschiedlich viele Informationen (für Knoten z.B. Gewicht und Label, für Endknoten dagegen das Label des (statischen) Hypergraphen, in dem sich der Endknoten befindet, Label und Gewicht der Hyperkante, zu der der Endknoten gehört, Label und Gewicht des Knotens, auf den sich der Endknoten bezieht, sowie das Gewicht des Endknotens). Durch Nutzung der *tool tips* ist es beispielsweise möglich, genaue Informationen zu einem Element zu erhalten, ohne die Position des Cursors oder die aktuelle Auswahl zu verändern.

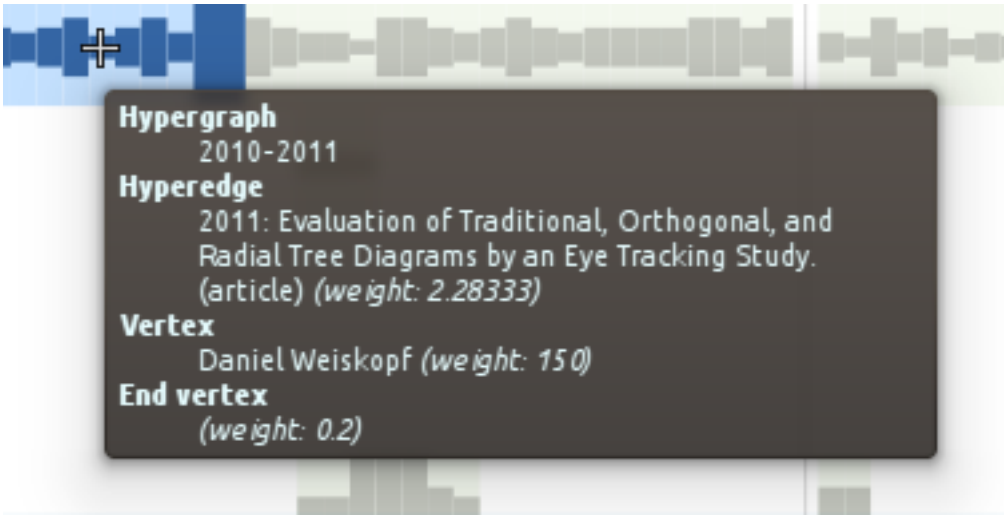


Abbildung 4.6.: *Tool tip* eines Endknotens

4.5. Dateiexport

Soll die Visualisierung extern weiterverwendet oder für die spätere Verwendung gespeichert werden, kann sie im aktuellen Zustand (einschließlich Auswahl und Cursor) in eine Datei im PNG-Format exportiert werden. Der Benutzer kann in einem Dialog auswählen, ob die komplette Visualisierung oder nur der sichtbare Bereich exportiert werden soll. Außerdem kann festgelegt werden, ob Daten zur Visualisierung in die Metadaten der PNG-Datei geschrieben werden sollen. Diese Daten beinhalten die Parameter, die für die Visualisierung verwendet wurden (wie z.B. die Art der Knoten- und Hyperkantenanordnung), Daten zur Programmversion, sowie den Namen der Datei, aus der der Datensatz importiert wurde und deren MD5-Hash [md5]. Auf diese Weise kann später nachvollzogen werden, wie – und aus welchen Daten – die Visualisierung erstellt wurde [1] . Der gespeicherte Hashwert hilft unter anderem, die richtige Datei zu bestimmen, falls mehreren Versionen des Datensatzes in gleich benannten Dateien gespeichert wurden. Hierfür muss für alle Dateien, die als Quelldateien in Frage kommen, ebenfalls der Hashwert berechnet werden; durch Vergleich mit dem gespeicherten Wert kann dann (falls vorhanden) die für das Erstellen der Visualisierung verwendete Datei bestimmt werden.

Außerdem können dieselben Daten ebenfalls in eine Textdatei gespeichert werden. Diese kann im Vergleich mit den Metadaten der PNG-Datei leichter betrachtet werden, allerdings kann sie auch leichter „verloren gehen“.

[1] Es ist nicht unbedingt möglich, mithilfe der gespeicherten Quelldateiinformationen und Visualisierungsparameter das Visualisierungsergebnis exakt zu reproduzieren. Durch geänderte Kompilierungsoptionen (wie z.B. (de)aktivierte Optimierung) können sich z.B. die Ergebnisse von Fließkommazahlvergleichen ändern [fpc], was sich beispielsweise auf die Anordnung der Knoten und Hyperkanten auswirken kann.

Listing 2 zeigt ein Beispiel für die eingebetteten bzw. in die Textdatei geschriebenen Daten (die Formatierung der Daten ist an YAML [yam] angelehnt).

```
ApplicationName: sdhvis
ApplicationVersion: 0.9.5
SourceFileName: weiskopf_daniel_2_year_intervals.graphml
SourceFileHashes: {MD5: 852e72c217258ab871567fa7b8f1fb23}
FilterTarget: ftVertices
FilterThreshold: 5
RemoveEmptyNonFilteredElements: false
WeightMappingTarget: wmtVerticalSplitCentered
WeightTransformation: wtLinear
VertexOrderType: otFromClustering
VertexClusteringMethod: cmPairwiseCentroidLinkageClustering
VertexClusteringDistanceMeasure: dmUncenteredPearsonCorrelation
HyperedgeOrderType: otFromClustering
HyperedgeClusteringMethod: cmPairwiseSingleLinkageClustering
HyperedgeClusteringDistanceMeasure: dmUncenteredPearsonCorrelation
Transposed: false
DrawInnerBackdropRectangles: true
DrawOuterBackdropRectangles: true
MaximumNumberOfCategories: 5
CategorizationTarget: ctHyperedges
CategorizationRegEx: "\\(([^)]+)\\)$"
ColorPalette: Tango-like
ExportArea: eaCompleteVisualization
```

Listing 2: Beispiel für Daten zur Visualisierung (in PNG-Metadaten/Textdatei)

5. Fallstudie

Die Visualisierungstechnik soll anhand mehrerer Beispieldatensätze demonstriert und untersucht werden.

Die Datensätze wurden auf Basis von DBLP-Daten erzeugt. DBLP [dbl] enthält Metadaten, wie z.B. Autoren, Titel und Publikationsjahr, einer großen Anzahl von Publikationen aus dem Bereich der Informatik.

Es wurden über die Autorensuche auf *http://dblp.uni-trier.de/* Publikationsdaten für sieben Professoren des Instituts für Visualisierung und Interaktive Systeme (VIS) der Universität Stuttgart gesucht und von der jeweiligen Autorenseite als XML-Dateien heruntergeladen. Im darauffolgenden Datenvorverarbeitungsschritt wurden mithilfe eines Python-3-Skriptes aus unterschiedlichen Kombinationen dieser XML-Dateien mehrere dynamische Hypergraphen erzeugt; hierbei wurden Publikationen folgenden Typs berücksichtigt: „article“, „inproceedings“, „book“, „incollection“, „phdthesis“ und „mastersthesis“. Die erzeugten dynamischen Hypergraphen wurden daraufhin im bereits beschriebenen GraphML-Format gespeichert und mit xmlstarlet [xml] anhand des in Kapitel 4 vorgestellten XML Schemas erfolgreich validiert.

Eine Hyperkante eines solchen Hypergraphen entspricht hierbei einer Publikation; ein Knoten einem Autor. Da in der vorliegenden Arbeit überwiegend dynamische Hypergraphen betrachtet werden, wurden außerdem die Publikationen anhand ihres Veröffentlichungsjahres in Zeitintervalle gleicher Länge (beispielsweise 2 Jahre pro Intervall) gruppiert. Diese Gruppen entsprechen jeweils einem Zeitschritt des zu erstellenden dynamischen Hypergraphen.

Die Gewichtsfunktionen wurden folgendermaßen gewählt: die Knotengewichte (d.h. $w_v(v)$) geben für alle Knoten die Anzahl der (im Datensatz enthaltenen) Publikationen an, an denen der dem Knoten entsprechende Autor beteiligt ist; das Gewicht eines Endknotens v einer Hyperkante e beträgt $\frac{1}{p(v)}$. Hierbei entspricht v einem Mitautor der Publikation, die durch e repräsentiert wird und $p(v)$ gibt die ab 1 indizierte Position des Autors in der Autorenliste der entsprechenden Publikation an. Das Gewicht einer Hyperkante berechnet sich als die Summe der Gewichte ihrer Endknoten.

Für die nachfolgenden Beispiele wurden „weniger wichtige“ Autoren (hier festgelegt als Autoren mit weniger als fünf Publikationen in Bezug auf den visualisierten Datensatz) herausgefiltert. Die Gewichte der Hyperkanten beziehen sich jedoch auf die Gesamtzahl der beteiligten Autoren vor dem Filtern.

Wie im Folgenden erkennbar wird, lassen sich mithilfe der vorgestellten Visualisierungstechnik aus den erzeugten Datensätzen Informationen sowohl in Bezug auf einzelne Autoren oder Publikationen als auch auf die Zusammenarbeit mehrerer Autoren an Publikationen gewinnen.

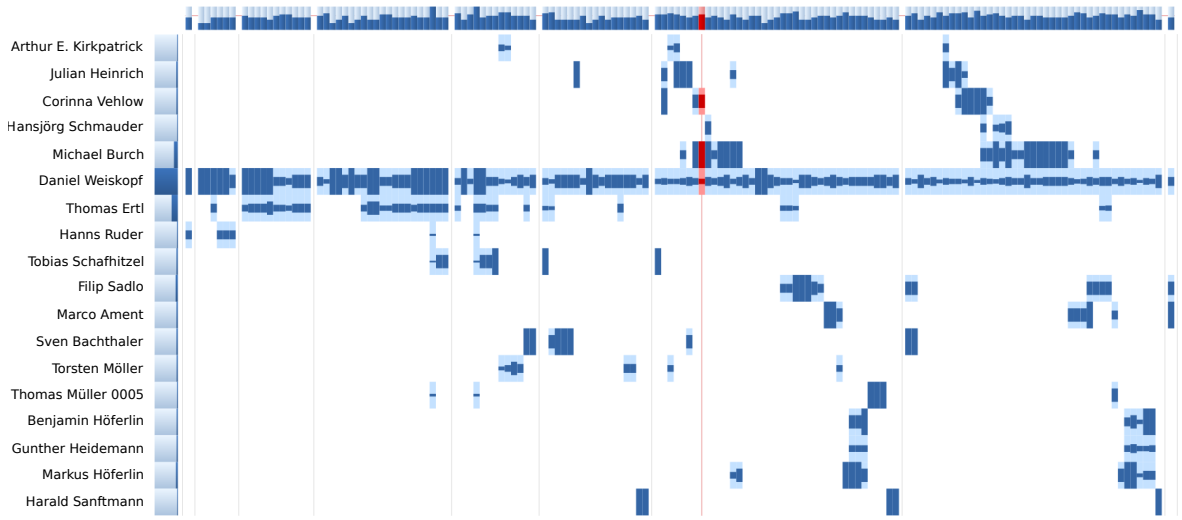


Abbildung 5.1.: Publikationen mit Beteiligung Professor Weiskopfs (Knoten nachträglich beschriftet); Cursor auf Paper „Parallel Edge Splatting for Scalable Dynamic Graph Visualization“

Der erste Datensatz enthält Daten zu Publikationen, an welchen Professor Weiskopf beteiligt ist. Der (gefilterte) Datensatz besteht aus einem dynamischen Hypergraphen mit neun „Zeitschritten“ (Zwei-Jahres-Intervalle, beginnend mit 1998–1999, endend mit 2014–2015), 150 Hyperkanten (d.h. Publikationen), 18 Knoten (d.h. Autoren) und 338 Endknoten (d.h. Mitautorschaften eines Autors an einer Publikation).

Das Gewicht der Endknoten wurde linear auf die Rechteckhöhe (zentriert) abgebildet (Hintergrundrechtecke wurden gezeichnet); für die Sortierung der Knoten wurde hierarchisches Clustering mit der Methode *pairwise centroid linkage* und dem Distanzmaß *uncentered Pearson correlation* verwendet, für die Sortierung der Hyperkanten ebenfalls hierarchisches Clustering, allerdings mit der Methode *pairwise single linkage* und dem Distanzmaß *uncentered Pearson correlation*. Die Knoten wurden nachträglich beschriftet.

Zuerst soll das Paper „Parallel Edge Splatting for Scalable Dynamic Graph Visualization“ untersucht werden. Abbildung 5.1 zeigt den visualisierten Datensatz; der Cursor wurde auf der Hyperkante platziert, die dem zu untersuchenden Paper entspricht.

Beim Vergleich der am oberen Visualisierungsrand dargestellten Gewichte aller Publikationen ist erkennbar, dass an diesem Paper zwar mehr Autoren beteiligt sind als an der Publikation mit der geringsten Anzahl an Autoren, aber auch weniger als an der Publikation mit der größten Anzahl an Autoren.

Die (nicht herausgefilterten) beteiligten Autoren sind in der Spalte des Cursors rot markiert erkennbar. Mithilfe des Cursors können folgende beteiligte Autoren und deren Position in der Autorenliste des Papers bestimmt werden: Corinna Vehlow (Position 2, da das Gewicht des entsprechenden Endknotens 0,5 ($= \frac{1}{2}$) beträgt), Michael Burch (Position 1, aus Gewicht 1), Daniel Weiskopf (Position 5, aus Gewicht 0,2). Ein Beispiel für die angezeigten Cursor-Informationen ist in Abbildung 5.2 sichtbar. Dies sind

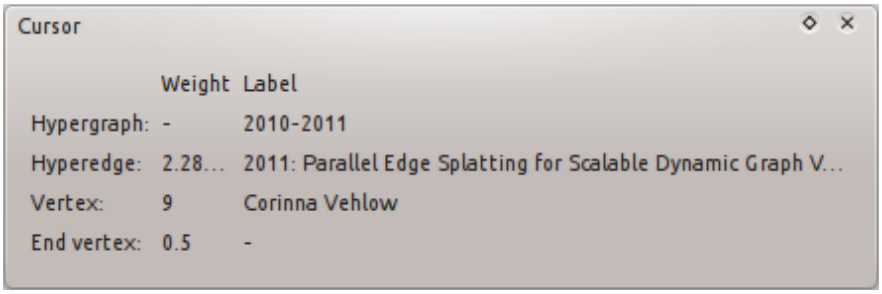


Abbildung 5.2.: Informationen zum Element unter dem Cursor (Screenshot der graphischen Benutzeroberfläche)

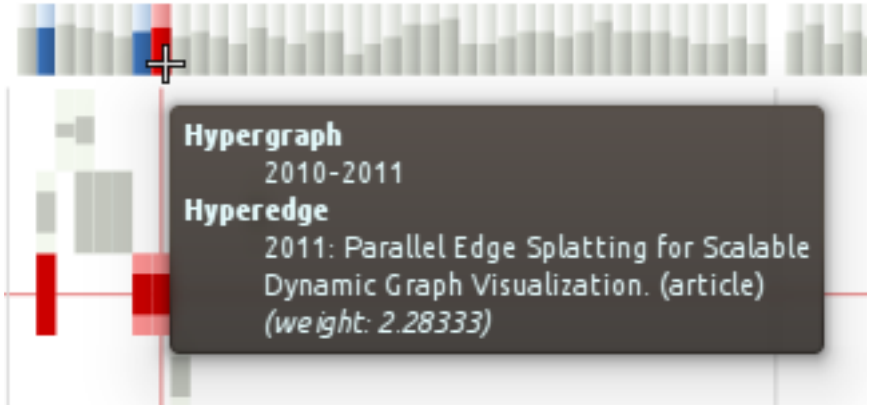


Abbildung 5.3.: Informationen zum Element unter dem Mauszeiger (Screenshot der graphischen Benutzeroberfläche)

nicht unbedingt alle beteiligten Autoren, da durch Filtern oder einen unvollständigen Datensatz einige Autoren in der Visualisierung nicht vorhanden sein könnten. Tatsächlich deuten sowohl die Positionen der bereits genannten Autoren (durch fehlende Positionen), als auch das Gewicht der Hyperkante (größer als $\sum_{i=1}^{3} \frac{1}{i}$) darauf hin, dass zusätzliche Mitautoren existieren. Dieses Gewicht kann beispielsweise aus dem *tool tip* der Hyperkante abgelesen werden, siehe Abbildung 5.3). Ein Vergleich mit der (hier nicht dargestellten) ungefilterten Visualisierung, die 174 Autoren berücksichtigt, liefert Fabian Beck (Position 3) und Stephan Diehl (Position 4) als restliche Autoren. Eine Nachprüfung anhand der Originalpublikation zeigt, dass sowohl die Autoren als auch deren Position in der Autorenliste korrekt sind.

Nun sollen die im Datensatz erfassten Publikationen bestimmt werden, an denen Corinna Vehlow beteiligt ist.

Abbildung 5.4 zeigt den visualisierten Datensatz, wobei der Knoten ausgewählt wurde, der Corinna Vehlow repräsentiert.

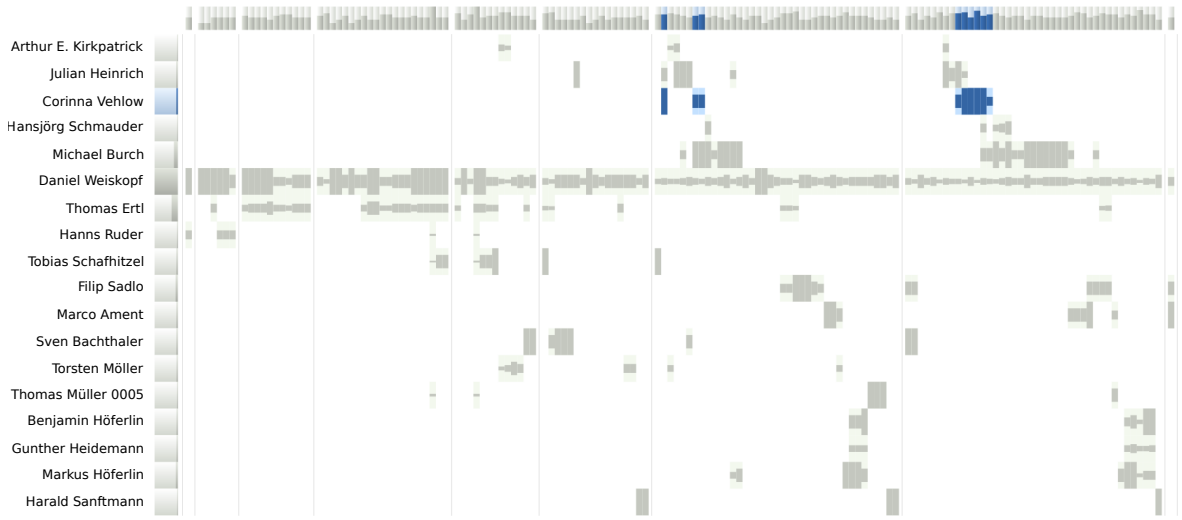


Abbildung 5.4.: Publikationen mit Beteiligung Professor Weiskopfs (Knoten nachträglich beschriftet); Auswahl des Knotens für Corinna Vehlow

Selected end vertices

	Hypergraph label	Hyperedge label	Hyperedge weight	Vertex label	Vertex weight	End vertex weight
1	2010-2011	2011: iHAT: Interactive hierarchical aggregation table. (inproceedings)	2,28333	Corinna Vehlow	9	1
2	2012-2013	2012: Uncertainty-aware visual analysis of biochemical reaction networks. (inproceedings)	2,59286	Corinna Vehlow	9	1
3	2012-2013	2013: Visualizing Fuzzy Overlapping Communities in Networks. (article)	1,83333	Corinna Vehlow	9	1
4	2012-2013	2013: iVUN: interactive Visualization of Uncertain biochemical reaction Networks. (article)	2,82897	Corinna Vehlow	9	1
5	2012-2013	2013: Radial Layered Matrix Visualization of Dynamic Graphs. (inproceedings)	2,08333	Corinna Vehlow	9	1
6	2010-2011	2011: Evaluating Partially Drawn Links for Directed Graph Edges. (inproceedings)	2,08333	Corinna Vehlow	9	0,5
7	2012-2013	2012: iHAT: interactive Hierarchical Aggregation Table for Genetic Association Data. (article)	2,45	Corinna Vehlow	9	0,5
8	2010-2011	2011: Parallel Edge Splatting for Scalable Dynamic Graph Visualization. (article)	2,28333	Corinna Vehlow	9	0,5
9	2012-2013	2012: Rapid Serial Visual Presentation in dynamic graph visualization. (inproceedings)	2,28333	Corinna Vehlow	9	0,333333

Abbildung 5.5.: Informationen zu den ausgewählten Endknoten, sortiert nach Endknoten-Gewicht (Screenshot der graphischen Benutzeroberfläche)

Auch hier können wieder die am Visualisierungsrand dargestellten Gewichte verglichen werden – diesmal am linken Visualisierungsrand. Beim Vergleich des Gewichts des Knotens, der die Autorin repräsentiert ist erkennbar, dass sowohl Autoren mit weniger Publikationen (in Bezug auf den betrachteten Datensatz) als auch Autoren mit mehr Publikationen visualisiert werden.

Aus der Visualisierung können jetzt die Publikationen bestimmt werden, an denen sowohl Professor Weiskopf als auch Corinna Vehlow beteiligt sind (die Beteiligung von Daniel Weiskopf ist durch die Beschaffenheit des Datensatzes gegeben). Zur Bestimmung der Mitarbeit von Corinna Vehlow könnte wie im vorherigen Beispiel der Cursor verwendet werden – hier soll aber von der Möglichkeit der Auswahl von (End)Knoten Gebrauch gemacht werden. Wie bereits erwähnt, ist in Abbildung 5.4 der Knoten ausgewählt, der Corinna Vehlow repräsentiert. In der Tabelle der ausgewählten Endknoten (siehe Abbildung 5.5) sind dann genau die Endknoten enthalten, die zu diesem Knoten gehören.

Aus der Tabelle zu entnehmen ist die Beteiligung an folgenden neun Publikationen:

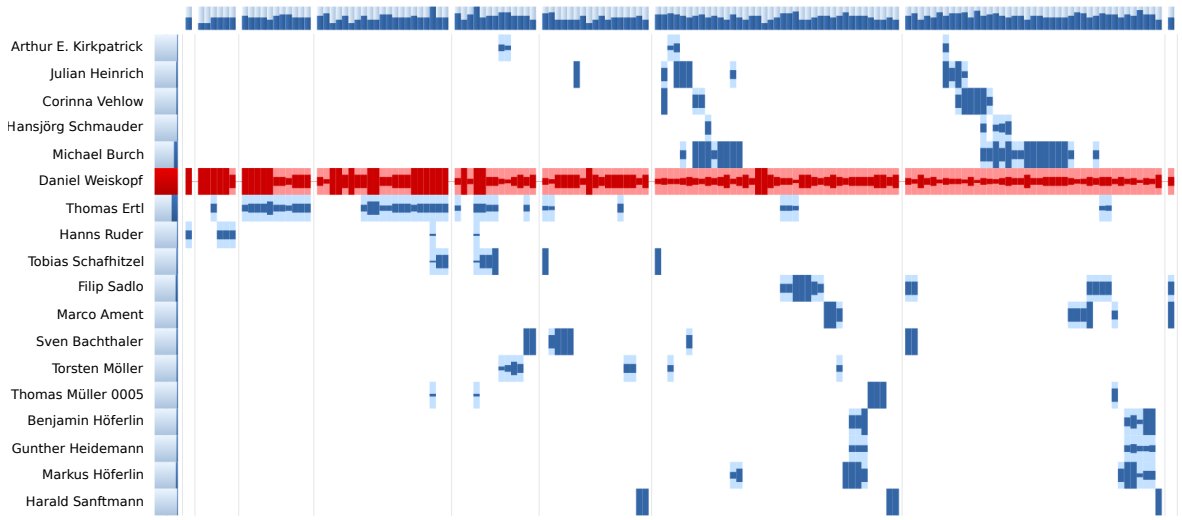


Abbildung 5.6.: Publikationen mit Beteiligung Professor Weiskopfs (Knoten nachträglich beschriftet); Cursor auf Autor „Daniel Weiskopf“

- iHAT: Interactive hierarchical aggregation table.
- Uncertainty-aware visual analysis of biochemical reaction networks.
- Visualizing Fuzzy Overlapping Communities in Networks.
- iVUN: interactive Visualization of Uncertain biochemical reaction Networks.
- Radial Layered Matrix Visualization of Dynamic Graphs.
- Evaluating Partially Drawn Links for Directed Graph Edges.
- Parallel Edge Splatting for Scalable Dynamic Graph Visualization.
- iHAT: interactive Hierarchical Aggregation Table for Genetic Association Data.
- Rapid Serial Visual Presentation in dynamic graph visualization.

Sowohl der Visualisierung als auch der Tabelle ist zu entnehmen, dass Corinna Vehlow bei fünf dieser Publikationen Erstautorin ist, bei drei Publikationen an zweiter Stelle in der Autorenliste steht, bei einer Publikation an dritter Stelle. An einer dieser Publikationen sind (herleitbar aus dem Gewicht der entsprechenden Hyperkante) drei Autoren beteiligt, an zwei Publikationen vier Autoren, an drei Publikationen fünf Autoren, an einer Publikation sechs Autoren, an einer Publikation sieben Autoren und an einer Publikation neun Autoren.

Als nächstes soll der „Publikationsverlauf“ Professor Weiskopfs untersucht werden.

Abbildung 5.6 zeigt den betrachteten Datensatz; der Cursor wurde auf dem Knoten positioniert, der Professor Weiskopf repräsentiert.

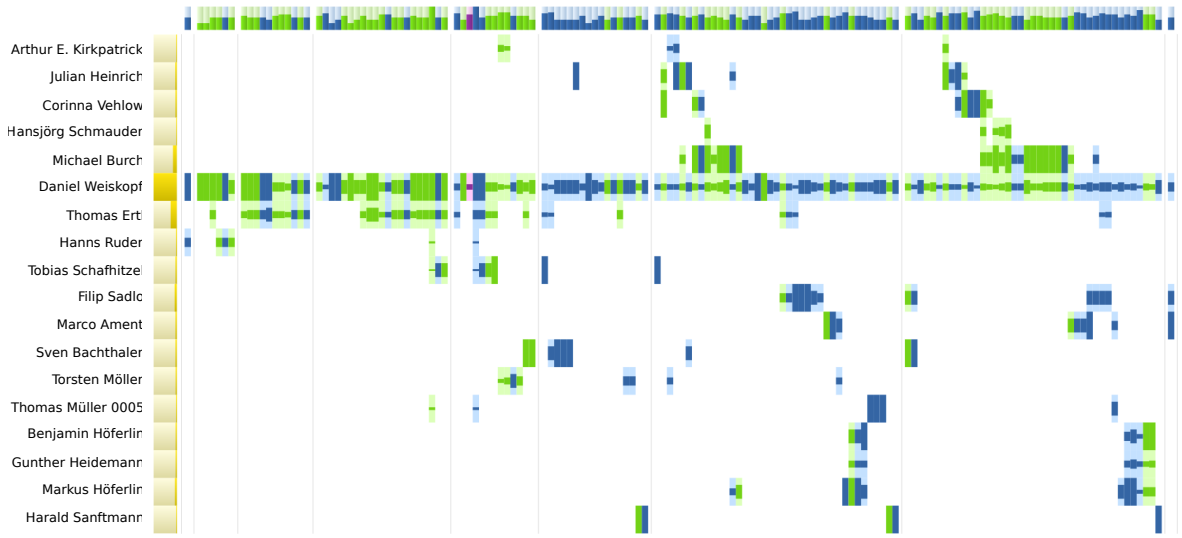


Abbildung 5.7.: Publikationen mit Beteiligung Professor Weiskopfs (Knoten nachträglich beschriftet); Kategorisierung nach Publikationstyp

In der Visualisierung ist zu erkennen, dass die Anzahl der Publikationen in den ersten vier Zeitintervallen (Intervall 1998–1999 bis Intervall 2004–2005) zunimmt, im Zeitintervall 2006–2007 dann fast auf das Niveau von Intervall 2002–2003 abfällt und daraufhin wieder ansteigt (das letzte Intervall enthält die noch nicht (vollständig) abgeschlossenen Jahre 2014/2015).

Weiterhin wird sichtbar, dass Professor Weiskopf zu Beginn häufig Erstautor der betrachteten Publikationen ist, während er später eher an der zweiten oder einer der darauffolgenden Positionen der Autorenliste steht. Dies deutet darauf hin, dass er im Lauf der Zeit die Betreuung von Arbeiten übernahm.

Außerdem ist zu erkennen, dass er in den ersten drei Zeitintervallen nur mit wenigen weiteren Autoren (zwei Autoren in der gefilterten Visualisierung) publizierte, ab dem vierten Zeitintervall (2004–2005) mit einer größeren Anzahl an Autoren. Zum Vergleich: Die (hier nicht dargestellte) ungefilterte Visualisierung zeigt fünf Mitautoren für die ersten zwei Zeitintervalle und 13 Mitautoren für das dritte Zeitintervall. Häufigste Mitautoren zu Beginn waren Hanns Ruder, dann Thomas Ertl (dies wird durch die ungefilterte Visualisierung bestätigt); ab 2010/2011 publizierte Daniel Weiskopf (als nicht-Erstautor) oft zusammen mit Michael Burch, was auf die Rolle Professor Weiskopfs als (Mit-)Betreuer der Arbeiten von Michael Burch hindeuten könnte. Außerdem ist zu vermuten, dass Michael Burch im Zeitraum 2010–2011 an die Universität wechselte, an der Professor Weiskopf in diesem Zeitraum tätig war.

Zuletzt soll die nach Publikationstyp kategorisierte Visualisierung (siehe Abbildung 5.7) betrachtet werden; sie zeigt, dass der Datensatz 79 Publikationen des Typs „article“ enthält (blau), 70 Publikationen des Typs „inproceedings“ (grün) und eine Publikation des Typs „book“ (lila). Abbildung 5.8 enthält die in der graphischen Benutzeroberfläche angezeigte Legende.

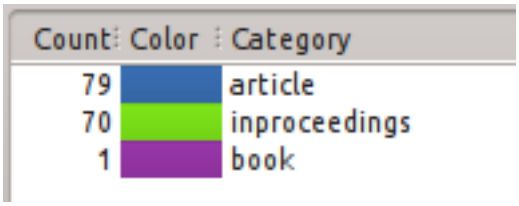


Abbildung 5.8.: Liste der Kategorien mit Anzahl der Hyperkanten und Farbe der Kategorie (Screenshot der graphischen Benutzeroberfläche)

Der zweite Datensatz (Abbildung 5.9) zeigt die Publikationen aller sieben Professoren. Der (gefilterte) Datensatz besteht aus einem dynamischen Hypergraphen mit zehn Zeitschritten (Fünf-Jahres-Intervalle, beginnend mit 1960–1964, endend mit 2010–2014, ohne das Intervall 1975–1979), 814 Hyperkanten (Publikationen), 125 Knoten (Autoren) und 2176 Endknoten (Mitautorschaften eines Autors an einer Publikation).

Die Abbildung der Gewichte und die Hyperkanten- und Knotensortierung wurden gleich gewählt wie für den ersten Datensatz; Beschriftungen und Bereichsmarkierungen wurden nachträglich hinzugefügt.

Die Visualisierung deutet darauf hin, dass das Publikationsvolumen im Verlauf der Jahre zugenommen hat – Professor Gunzenhäuser publizierte in den Jahren 1963–2010 relativ selten (17 Publikationen im Datensatz), während beispielsweise Thomas Ertl im Zeitraum 1989–2013 an mehr als 300 Publikationen beteiligt ist. Durch die Gewichtung der Endknoten wird andererseits erkennbar, dass Professor Ertl nur bei einem relativ kleinen Teil dieser Publikationen Erstautor ist, während dies bei Professor Gunzenhäuser auf fast die Hälfte seiner Publikationen zutrifft.

Außerdem ist zu erkennen, dass die Professoren an einer (teilweise sehr viel) größeren Anzahl an Publikationen beteiligt sind als die meisten anderen Autoren.

Weiterhin sind in der Visualisierung Cluster zu erkennen (beispielsweise in den in Abbildung 5.9 mit *A* und *B* bezeichneten, markierten Bereichen), die darauf hindeuten, dass Autorengruppen existieren, deren Mitglieder hauptsächlich mit jeweils einem der Professoren bzw. anderen Mitgliedern der Gruppe publizieren. Eine stichprobenhafte Überprüfung bestätigt dies.

Erkennbare Muster sind zudem horizontal parallel verlaufende Streifen, sichtbar z.B. in den Bereichen *C*, *D* und *E*. Diese treten auf, wenn eine Gruppe von Autoren an mehreren Publikationen beteiligt ist (in diesem Fall Professor Ertl und Professor Weiskopf, die an mehr als 50 Publikationen zusammen beteiligt sind). Ausgehend von Publikationen, ist die Beteiligung einer Gruppe auch als vertikal parallel verlaufende Streifen sichtbar (wie z.B. in Bereich *F*). Je nach Hyperkanten- bzw. Knotenanordnung sind diese Muster teilweise unterbrochen. Weiterhin ist erkennbar, dass durch die in Kapitel 3 beschriebene Umsortierung der Clusteringhierarchien wie gewünscht ähnliche Cluster in verschiedenen Zeitintervallen an ähnlichen Positionen platziert werden (erkennbar z.B. an den Bereichen *G* und *H*).

Eine Version der Visualisierung ohne Markierungen befindet sich im Anhang in Abbildung A.1.

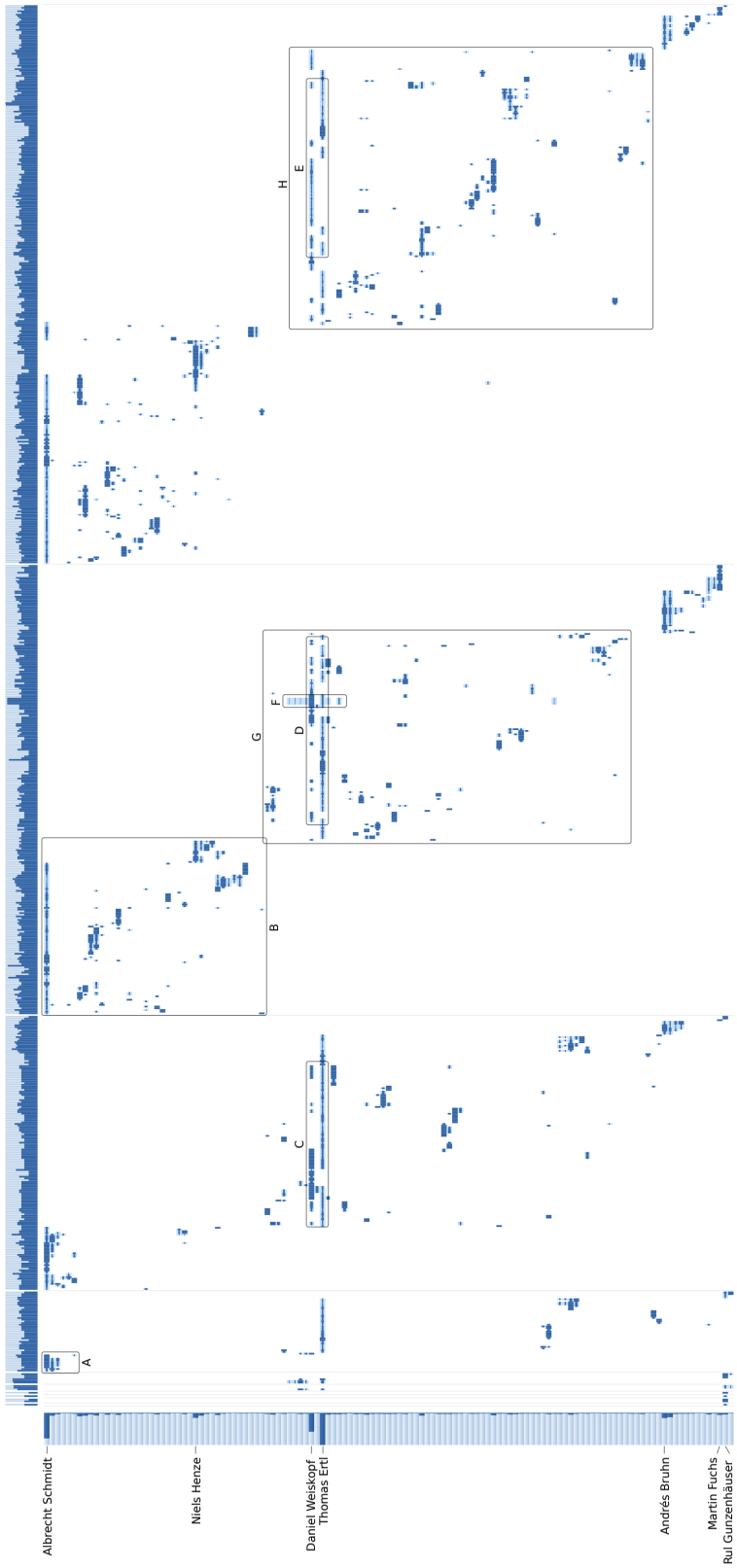


Abbildung 5.9.: Publikationen aller sieben VIS-Professoren (Beschriftungen und Bereichsmarkierungen nachträglich hinzugefügt)

6. Zusammenfassung und Ausblick

In dieser Arbeit wurde ein Ansatz für die skalierbare Visualisierung dynamischer Hypergraphen entwickelt. Dieser basiert auf dem *Quad-Schema* zur Visualisierung statischer Hypergraphen und besitzt außerdem Eigenschaften der Matrixdarstellung gewöhnlicher Graphen. Für die Abbildung der zeitlichen Dimension der dynamischen Hypergraphen wurde die Abbildung der Zeit auf den Raum (*time-to-space mapping*) gewählt, um Probleme der animierten Darstellung einer Visualisierung zu vermeiden.

Der entwickelte Ansatz unterstützt die Visualisierung von Knoten-, Hyperkanten- und Endknotengewichten und ist durch eine Reihe von Möglichkeiten zur Änderung der Anordnung der Knoten und Hyperkanten darauf ausgelegt, Strukturen in den visualisierten Daten sichtbar zu machen.

Die Visualisierungstechnik wurde als interaktives Visualisierungswerkzeug implementiert und bietet durch Interaktionstechniken zusätzliche Möglichkeiten zur Gewinnung von Einsichten in den visualisierten Datensatz.

Anhand einer Fallstudie wurde die Nützlichkeit der Visualisierung und der Interaktionstechniken demonstriert.

Ausblick

Der vorgestellte Visualisierungsansatz könnte für Datensätze verbessert werden, die in allen oder zumindest mehreren Zeitschritten Hyperkanten mit derselben Bedeutung besitzen. Dies trifft z.B. auf einen Datensatz zu, der Regisseure auf Hyperkanten und Schauspieler auf Knoten abbildet, wobei die Zugehörigkeit eines Knotens zu einer Hyperkante die Zusammenarbeit von Schauspieler und Regisseur ausdrückt. Die Schaffenszeit der betrachteten Regisseure kann dann in Teilintervalle aufgeteilt werden, die jeweils einem Schritt des dynamischen Hypergraphen entsprechen. Hierbei kann es vorkommen, dass mehrere Zeitschritte eine Hyperkante enthalten, die denselben Regisseur repräsentiert. Mit der vorgestellten Methode werden diese Hyperkanten unabhängig voneinander horizontal angeordnet – dies widerspricht dem Wunsch, die *mental map* des Betrachters zu erhalten. Eine Möglichkeit der Verbesserung wäre nun, nicht Hyperkanten eines Hypergraphen in einem rechteckigen Bereich zu gruppieren, sondern Hyperkanten, die dieselbe Bedeutung besitzen – im Beispiel also alle Hyperkanten eines Regisseurs. Die Zugehörigkeit der Hyperkanten zu Zeitpunkten des dynamischen Hypergraphen könnte hierbei durch die Reihenfolge der Hyperkanten in der Gruppierung dargestellt werden. Auf diese Weise könnte die zeitliche Entwicklung der Zusammenarbeit zwischen Regisseur und Schauspielern untersucht werden, ohne die jeweilige Hyperkante in den verschiedenen Hypergraphen suchen zu müssen (was im vorgestellten Ansatz trotz der Umsortierung der Clusterhierarchie nötig ist). Eine weitere interessante Anwendung dieser alternativen Anordnung wäre die Analyse der

gespielten Titel (Knoten) mehrerer Radiosender (Hyperkanten) über einen in Abschnitte (z.B. 24 Stunden) aufgeteilten Zeitraum.

Für beide Datensätze wäre außerdem eine Möglichkeit der Aggregation (mehrerer Zeitschritte des dynamischen Hypergraphen zu einem Zeitschritt (gespielte Titel über eine Woche statt einen Tag, …), mehrerer Hyperkanten zu einer Hyperkante (Sender einer Sendergruppe, …), mehrerer Knoten zu einem Knoten (Gruppen von Schauspielern, …) nützlich, um auf einfache Weise einen Überblick über die Daten zu gewinnen.

Diese Aggregation könnte beispielsweise über die für das Clustering aufgebauten Hierarchien geschehen; diese Hierarchien könnten hierfür beispielsweise am (linken/oberen oder rechten/unteren) Rand der Hauptvisualisierung dargestellt werden. Außerdem wäre ein Umsortieren dieser Hierarchien durch den Nutzer denkbar. Auch zusätzliche Arten der Hyperkanten- und Knotensortierung könnten zur Verfügung gestellt werden, z.B. durch weitere Clusteringalgorithmen oder andere Sortierkriterien.

Für einige Datensätze könnte die Art der Kategorisierung erweitert werden, um beispielsweise nach der Anzahl der Endknoten einer Hyperkante oder der Anzahl der Hyperkanten, die mit einem Knoten inzident sind, kategorisieren zu können. Auch eine Kategorisierung nach Gewichten bzw. Wertebereichen von Gewichten wäre möglich. Die zum Visualisieren der Kategorien verwendeten Farben könnten außerdem an den Eigenschaften der menschlichen Wahrnehmung ausgerichtet werden.

Um die Visualisierung von Interaktionstechniken unabhängiger zu machen, wäre zudem die Darstellung von Labels möglich. Hierbei tritt unter Umständen das Problem auf, dass ein großer Bereich der Visualisierungsfläche von langen Labels (beispielsweise Publikationstiteln) belegt wird und somit nicht für die Hauptvisualisierung zur Verfügung steht.

Für die Verwendung *mit* Interaktion könnte das erstellte Programm um semantisches Zoomen oder eine Lupenfunktion (Vergrößern eines Visualisierungsausschnitts am Mauszeiger) erweitert werden. Außerdem könnte eine Suchfunktion für die Suche nach Hyperkanten und Knoten implementiert werden, möglicherweise zusammen mit einer Auflistung der existierenden Hyperkanten/Knoten.

Schließlich könnte das Filtern von Hyperkanten/Knoten erweitert werden, um nicht nur Minimalwerte der Anzahl von Endknoten bzw. inzidenten Knoten zu berücksichtigen, sondern andere Filterkriterien anwenden zu können. Spätestens dann wäre allerdings das Filtern während des Betrachtens eines Datensatzes (also nicht schon beim Import) wünschenswert.

A. Anhang

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://graphml.graphdrawing.org/xmlns"
           xmlns="http://graphml.graphdrawing.org/xmlns"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">

    <xs:redefine schemaLocation="http://graphml.graphdrawing.org/xmlns/1.1/graphml.xsd">

        <xs:attributeGroup name="graph.extra.attrib">
            <xs:attributeGroup ref="graph.extra.attrib"/>
            <xs:attribute name="label" type="xs:string" use="optional"/>
        </xs:attributeGroup>

        <xs:attributeGroup name="node.extra.attrib">
            <xs:attributeGroup ref="node.extra.attrib"/>
            <xs:attribute name="label" type="xs:string" use="optional"/>
            <xs:attribute name="weight" type="xs:double" use="optional"/>
        </xs:attributeGroup>

        <xs:attributeGroup name="hyperedge.extra.attrib">
            <xs:attributeGroup ref="hyperedge.extra.attrib"/>
            <xs:attribute name="label" type="xs:string" use="optional"/>
            <xs:attribute name="weight" type="xs:double" use="optional"/>
        </xs:attributeGroup>

        <xs:attributeGroup name="endpoint.extra.attrib">
            <xs:attributeGroup ref="endpoint.extra.attrib"/>
            <xs:attribute name="weight" type="xs:double" use="optional"/>
        </xs:attributeGroup>

    </xs:redefine>

</xs:schema>
```

Listing 3: XML Schema des verwendeten GraphML-basierten Eingabe-Dateiformats

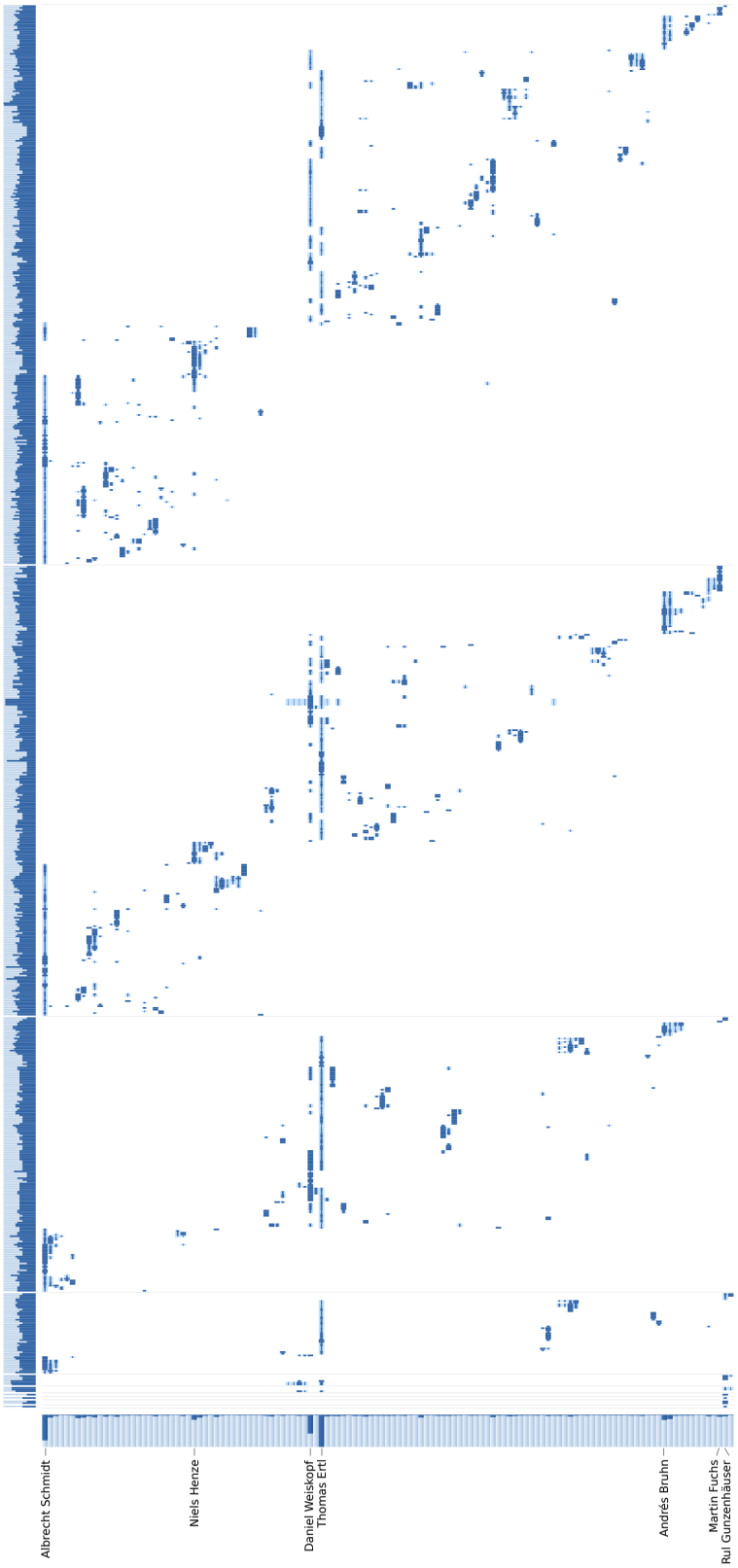


Abbildung A.1.: Publikationen aller sieben VIS-Professoren (Beschriftungen nachträglich hinzugefügt)

Literaturverzeichnis

[BBV+12] F. Beck, M. Burch, C. Vehlow, S. Diehl, D. Weiskopf. Rapid Serial Visual Presentation in dynamic graph visualization. In *Visual Languages and Human-Centric Computing (VL/HCC), 2012 IEEE Symposium on*, S. 185–192. 2012. doi:10.1109/VLHCC.2012.6344514. (Zitiert auf den Seiten 5 und 7)

[BVB+11] M. Burch, C. Vehlow, F. Beck, S. Diehl, D. Weiskopf. Parallel Edge Splatting for Scalable Dynamic Graph Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12:2344–2353, 2011. (Zitiert auf Seite 7)

[clu] The C Clustering Library. http://bonsai.hgc.jp/~mdehoon/software/cluster/cluster.pdf. (Zitiert auf den Seiten 16 und 21)

[col] ColorBrewer: Color Advice for Maps. http://colorbrewer2.org/. (Zitiert auf Seite 19)

[dbl] dblp: DBLP Computer Science Bibliography. http://dblp.uni-trier.de/. (Zitiert auf Seite 32)

[fpc] Bug 323 - optimized code gives strange floating point results. http://gcc.gnu.org/bugzilla/show_bug.cgi?id=323. (Zitiert auf Seite 30)

[FR91] T. M. Fruchterman, E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and experience*, 21(11):1129–1164, 1991. (Zitiert auf Seite 6)

[GFC04] M. Ghoniem, J. Fekete, P. Castagliola. A comparison of the readability of graphs using node-link and matrix-based representations. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, S. 17–24. IEEE, 2004. (Zitiert auf Seite 7)

[gra] The GraphML File Format. http://graphml.graphdrawing.org/. (Zitiert auf Seite 21)

[HMM00] I. Herman, G. Melançon, M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *Visualization and Computer Graphics, IEEE Transactions on*, 6(1):24–43, 2000. (Zitiert auf Seite 6)

[KJ13] A. Kerren, I. Jusufi. A Novel Radial Visualization Approach for Undirected Hypergraphs. 2013. (Zitiert auf den Seiten 9 und 10)

[KK89] T. Kamada, S. Kawai. An algorithm for drawing general undirected graphs. *Information processing letters*, 31(1):7–15, 1989. (Zitiert auf Seite 6)

[KP94] M. Kritz, K. Perlin. A new scheme for drawing hypergraphs. *International journal of computer mathematics*, 50(3-4):131–134, 1994. (Zitiert auf den Seiten 5 und 9)

[KVKS09] M. Kaufmann, M. Van Kreveld, B. Speckmann. Subdivision drawings of hypergraphs. In *Graph Drawing*, S. 396–407. Springer, 2009. (Zitiert auf den Seiten 5, 7 und 8)

[Mäk90] E. Mäkinen. How to draw a hypergraph. *International Journal of Computer Mathematics*, 34(3-4):177–185, 1990. (Zitiert auf Seite 8)

[md5] RFC 1321 - The MD5 Message-Digest Algorithm. http://tools.ietf.org/html/rfc1321. (Zitiert auf Seite 30)

[PHG07] H. C. Purchase, E. Hoggan, C. Görg. How important is the "mental map"?–an empirical investigation of a dynamic graph layout algorithm. In *Graph drawing*, S. 184–195. Springer, 2007. (Zitiert auf den Seiten 5 und 7)

[RLMJ05] R. Rosenholtz, Y. Li, J. Mansfield, Z. Jin. Feature Congestion: A Measure of Display Clutter. 2005. (Zitiert auf Seite 6)

[Spe07] R. Spence. *Information Visualization: Design for Interaction.* Pearson/Prentice Hall, 2007. (Zitiert auf Seite 29)

[SWS10] K. Stein, R. Wegener, C. Schlieder. Pixel-Oriented Visualization of Change in Social Networks. *ASONAM*, 10:233–240, 2010. (Zitiert auf Seite 7)

[tan] Tango Icon Theme Guidelines. http://tango.freedesktop.org/Tango_Icon_Theme_Guidelines. (Zitiert auf Seite 19)

[WF08] L. Wilkinson, M. Friendly. The History of the Cluster Heat Map. 2008. (Zitiert auf Seite 7)

[xml] XMLStarlet Command Line XML Toolkit. http://xmlstar.sourceforge.net/. (Zitiert auf Seite 32)

[yam] The Official YAML Web Site. http://yaml.org/. (Zitiert auf Seite 31)

Alle URLs wurden zuletzt am 17. 03. 2014 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift