Ein neues Empfehlungssystem mit FDSM-basierter einseitiger Projektion und Link Community Clustering

DIPLOMARBEIT

von Ying Gu

Universität Heidelberg
Fakultät für Mathematik
Februar 2012

Betreuer: Prof. Gerhard Reinelt
Dr. Katharina Zweig

Danksagung

An dieser Stelle möchte ich mich bei Professor Gerhard Reinelt und Dr. Katharina Anna Zweig für die interessante und praxisorientierte Aufgabenstellung bedanken.

Für die äußerst gute Betreuung und dafür, dass sie stets für mich erreichbar war und mir die Freiheit gegeben hat, die Arbeit nach eigenen Vorstellungen zu entwickeln, und für ihre Geduld möchte ich mich ganz besonders bei Frau Dr. Zweig bedanken.

Ebenfalls möchte ich mich bei Emőke-Agnes Horvát für die vielen nützlichen Hinweise während der Anfertigung meiner Diplomarbeit bedanken.

Mein Dank gilt auch meinem Liebsten für seine Bereitschaft die Arbeit Korrektur zu lesen und für die vielen sprachlichen Verbesserungsvorschläge.

Nicht zuletzt möchte ich mich auch bei meinen Eltern bedanken, denn ohne ihre Unterstützung wäre dieses Studium niemals möglich gewesen.

Inhaltsverzeichnis

1 Einführung

1.1 KDD und Data Mining

Angesichts ständig fallender Preise von Massenspeichersystemen sowie der zunehmenden Digitalisierung von Verkaufsprozessen (Barcode-Lesegeräte in Supermärkten, Online-Shopping) ist die Sammlung von Kunden- und Verkaufsdaten für Unternehmen immer einfacher geworden. Diese Anhäufung von Datenbeständen wird im Indealfall dazu genutzt, jedem Kunden (Konsumenten) ein dediziertes Angebot an Produktinformationen zu unterbreiten und damit einen besseren Service zu erbringen. Das Ziel von **Knowledge Discovery from Databases (KDD)** ist es, aus einer Unmenge an Rohinformationen ein gewisses Muster zu extrahieren, um Zusammenhänge zwischen einzelnen Objekten herzustellen oder gar Vorhersagen machen zu können über Objekte, die noch nicht miteinander in Verbindung stehen. Fayyad, Piatetsky-Shapiro und Smyth haben in [Fayyad:1996] den Begriff KDD definiert als ein Prozeß der (semi-)automatischen Extraktion von Wissen aus Datenbanken, das *gültig, bisher unbekannt und potentiell nützlich* (für eine gegebene Anwendung) ist.

Der KDD Prozeß besteht aus folgenden Schritten:

$$\boxed{\textit{Rohdaten} \rightarrow \textit{Preprocessing} \rightarrow \textit{Data Mining} \rightarrow \textit{Postprocessing} \rightarrow \textit{Information}}$$

Die Erfassung der **Rohdaten** und die anschließende *manuelle* Vorselektierung (**Preprocessing**) der Daten nach gewissen anwendungsspezifischen Kriterien ist eine langwierige Aufgabe, die in der Regel auf Beiträge von vielen "Zuarbeitern" angewiesen ist. Die so aufbereiteten Daten bzw. Datenbank dienen als Ausgangspunkt der eigentlichen Analyse: Das **Data Mining** konzentriert sich auf die Anwendung *effizienter Algorithmen*, die die in einer Datenbank enthaltenen gültigen Muster findet. Es geht um die Verarbeitung sehr großer Datenbestände, die sich manuell nicht mehr durchführen lässt.
Beim **Postprocessing** wird das Ergebnis dieser Analyse oft weiter gefiltert, visualisiert und interpretiert.

1.2 Aufgabenbeschreibung

Wir werden ein allgemeines Empfehlungssystem für die Vermarktung von Produkten entwickeln, das auf Methoden der Netzwerkanalyse basiert. Grundlage dieser Arbeit sind der Artikel von Zweig und Kaufmann [Zweig:2011] sowie der Artikel von Ahn, Bagrow and Lehmann [Ahn:2010].

Der KDD-Cup ist ein jährlich stattfindender Wettbewerb zum Thema KDD und Data Mining. Er wird von der ACM Special Interest Group on Knowledge Discovery and Data Mining[1] *organisiert. Das Thema des Preisausschreibens aus dem Jahre 2007 war in Zusammenarbeit mit der Firma Netflix*[2] *entstanden. Es hieß "Consumer recommendations" und bestand aus zwei verschiedenen Aufgaben, bei denen es darum ging, basierend auf einem Datensatz aus der Vergangenheit (1998 bis 2005), Prognosen (Welcher Konsument wird im Jahre 2006 welchen Film bewerten? Wieviele zusätzliche Bewertungen werden bestimmte Filme im Jahre 2006 erhalten?) über Kundenbewertungen zu machen. Ende des Jahres 2006 hatte die Firma Netflix selbst bereits ein Preisgeld*[3] *ausgeschrieben für den besten Algorithmus zur Vorhersage von Kundenbewertungen. Natürlich waren die realen Kundenbewertungen aus dem Jahre 2006 zum Zeitpunkt des Preisausschreibens bekannt und somit konnte die Qualität der Prognosen gemessen werden.*

Der Netflix-Datensatz besteht aus 100.480.507 Millionen Bewertungen von 17.770 Filmen, die von über 480.189 zufällig ausgewählten Konsumenten erfasst wurden. In dieser Arbeit beschäftigen wir uns nicht mit den Aufgaben des Preisausschreibens, sondern wir benutzen diesen Datensatz lediglich als Basisdatensatz für die Erprobung unseres neuen Empfehlungssystems.

Die Aufgabe eines Empfehlungssystems befasst sich grob gesprochen mit der zentralen Frage: **Wenn ein Produkt oder eine Reihe von Produkten einem Kunden gefällt, welche weiteren Produkte kann man ihm noch empfehlen?**

Das prominenteste Beispiel ist der Online-Händler Amazon. Nach dem Kauf eines Produktes oder sogar schon bei der Suche nach einem Produkt bekommt der Kunde eine Liste von sehr ähnlichen oder dazu passenden Produkten angezeigt. Ein solches Empfehlungssystem bietet nicht nur dem Kunden einen besseren Service, sondern erhöht auch den Umsatz des Verkäufers.

Aus der Sicht der klassischen Warenkorbanalyse lässt sich die obige Frage so umformulieren: **Welche Produkte werden signifikanterweise öfter zusammen gekauft als per Zufall erwartet?**

[1] `http://www.sigkdd.org/kddcup`

[2] Netflix Inc. ist in den USA ein Anbieter von On-Demand Internet Streaming Medien und Online DVD-Verleih.

[3] Siehe `http://www.netflixprize.com` und `http://en.wikipedia.org/wiki/Netflix_Prize`

Zur Beantwortung solcher Fragen verwendet die klassische Warenkorbanalyse Methoden aus der Assoziationsanalyse. Dabei spielt der Begriff der Interessantheit einer Assoziationsregel eine wesentliche Rolle. Das einfachste Modell um die Interessantheit einer Assoziationsregel zu quantifizieren ist das in [Piatetsky-Shapiro:1991] beschriebene *Stochastic Independence Model (SIM)* (vgl. Abschnitt 3.5).

Wir verfolgen einen **netzwerkanalytischen Ansatz**. Dabei werden wir ein neues Interessantheitsmaß zur Bewertung von Assoziationsregeln einführen und die sogenannte einseitige Projektion (*one-mode-projection*) nach Zweig und Kaufmann [Zweig:2011] realisieren, die das Konsumenten-Produkt-Netzwerk in ein zusammenhängendes Produkt-Netzwerk projiziert. Danach werden wir in der Lage sein, Methoden aus der Clusteranalyse nach Ahn et. al. [Ahn:2010] zu benutzen, um ähnliche Produkte in diesem Produkt-Netzwerk zu kategorisieren. Die in dieser Arbeit beschriebenen Verfahren könnten demnach auch auf andere Datensätze angewandt werden.

In unserer Anwendung sind die Produkte Filme bzw. Staffeln von Fernsehserien. Um die einzelnen Schritte auf dem Weg zu unserem Empfehlungssystem präziser und anschaulicher erläutern zu können, gehen wir hier kurz auf das Format des Netflix-Datensatzes ein. Dieser besteht aus zwei Textdateien,

- einer **Liste von Bewertungen (Ratings)**: Das ist der Hauptdatensatz. Diese Datei hat eine Größe von $1,4$ Gigabyte. Sie hat in jeder Zeile das Format

  `<Film-Nr>,<Kunden-Nr>,<Bewertung>`

  und wird aufsteigend nach `<Kunden-Nr>` sortiert.

- und einer **Liste von Filmdaten**:

  `<Film-Nr>, <Entscheinungsjahr>, <Film-Titel>`

  Sie ist nach `<Film-Nr>` aufsteigend von 1 bis 17.770 sortiert.

Die beiden Dateien enthalten keine weiteren Informationen. Die Algorithmen des Empfehlungssystems machen nur von der Liste der Bewertungen Gebrauch und erhalten keine Kenntnisse über die Filmtitel. Die Liste mit den Filmtiteln wird *ausschließlich* zur Qualitätsbewertung des vorgestellten Verfahrens sowie für die Darstellung von Ergebnissen benötigt.

1.3 Überblick und Aufbau der Arbeit

Die einzelnen Schritte des Empfehlungssystems sind in Abbilung 1 skizziert.

1. Zusätzliche Vorselektierungen:
   (a) Wir betrachten aus Gründen der Performance jeweils einen reduzierten Datensatz von 20.000 nach dem Zufallsprinzip (d.h. zufällig mit gleicher Wahrscheinlichkeit) ausgewählten Konsumenten[4].
   (b) Es werden nur die Bewertungen berücksichtigt, die einen Wert größer als 3 haben.

2. Unser Datensatz besteht aus genau zwei Mengen von Objekten: Konsumenten und Produkten (Filmen). Zur mathematischen Modellierung ist es daher sehr naheliegend, einen bipartiten Graphen B zu verwenden.
   In den Kapiteln 2 und 3 führen wir wichtige Grundbegriffe aus der Graphentheorie und der Assoziationsanalyse ein.

3. Wir interessieren uns für Zusammenhänge zwischen den Objekten aus der Menge der Produkte (Filme). Der Zusammenhang wird dabei aus rein netzwerkanalytischen Mitteln gewonnen. Das bedeutet, es werden ausschließlich die Informationen aus dem Graphen (Netzwerk) B ausgenutzt um herauszufinden, ob zwei Filme besonders oft zusammen angeschaut und gut bewertet werden.

   Im Kapitel 4 stellen wir die **einseitige Projektion (One-Mode Projektion)** aus dem Artikel von Zweig und Kaufmann [Zweig:2011] vor, mit der wir aus dem bipartiten Graphen B einen zusammenhängenden Graphen G auf der Produktemenge erzeugen.

   Im Abschnitt 4.8 stellen wir eine modifizierte Version der einseitigen Projektion vor und erläutern ihre Vorzüge nach einem experimentellen Vergleich.

4. Nach der Generierung des Graphen G, also nachdem gewisse Zusammenhänge auf der Menge der Produkte hergestellt sind, gilt es nun, die Elemente von G, also die Produkte in **Clustern** (Gruppen) zusammenzufassen. In unserem Fall sind die Produkte Filme bzw. Staffeln. Wir schließen daraus, dass die auf diese Weise zusammengefassten Filme bzw. Staffeln inhaltlich verwandt sind.

[4] Zweig hat in ihrem Artikel [Zweig:2010] experimentell gezeigt, dass bereits ein Datensatz von 10.000 Konsumenten genügen würden, um global beste Freundepaare aufzufinden.

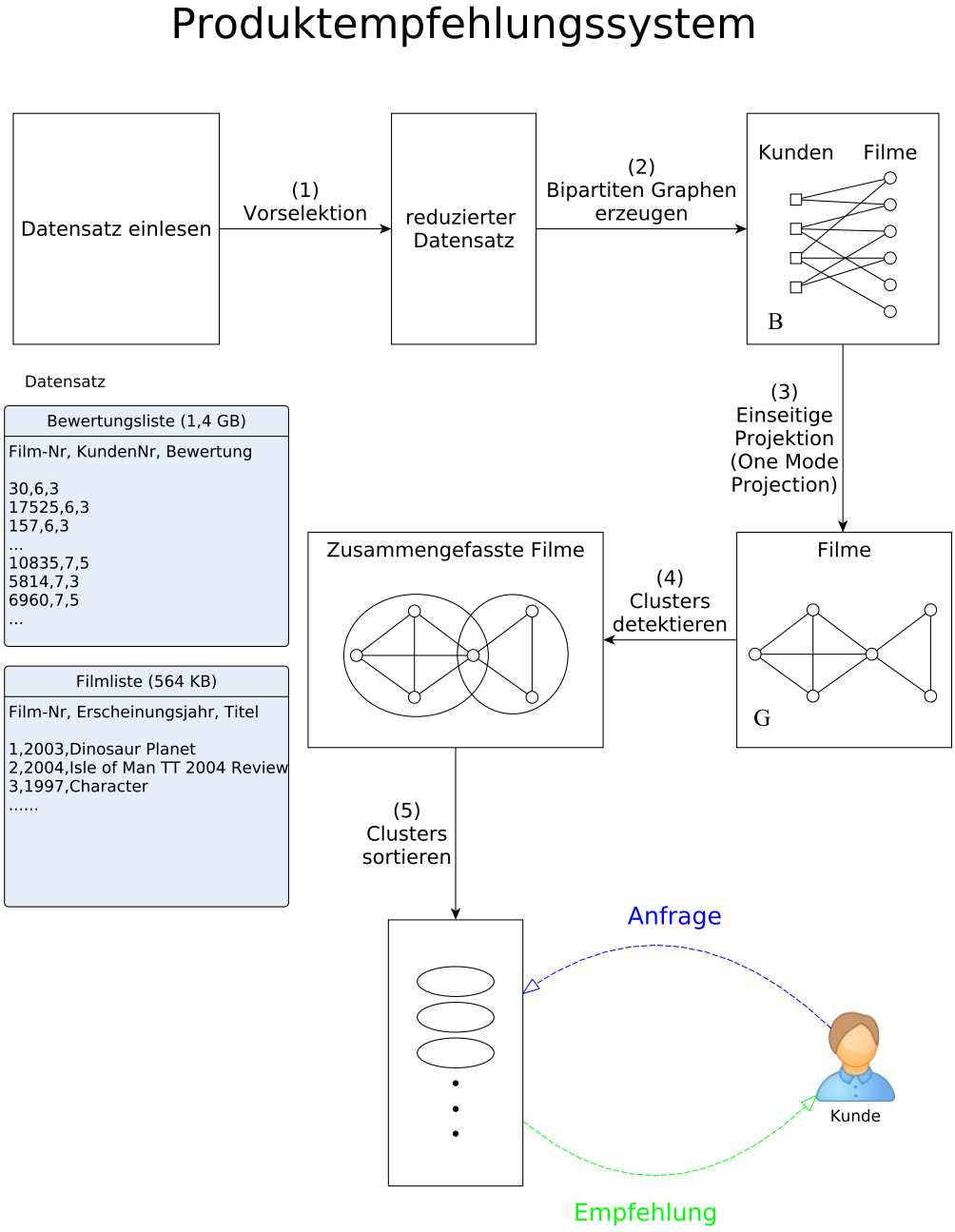


Abbildung 1: Empfehlungssystem

Im Kapitel 5 führen wir wichtige Kenngrößen von Netzwerk-Clustern ein und stellen die **Single-Linkage Hierarchical Clustering Methode** und den **Link-Community-Algorithmus** vor, mit denen wir in der Lage sein werden eine Hierarchie von Clustern für den Graphen G aufzubauen.

5. Die so gewonnenen Cluster müssen für unser Empfehlungssystem sortiert werden.
6. In dem fertigen Empfehlungssystem erhält ein Konsument durch Eingabe eines Produktes (hier: Film-Nr.) eine Liste von inhaltlich verwandten Filmen.

Der erste Schritt gehört noch zum *Preprocessing*, die Schritte 2 bis 4 gehören zum *Data Mining* und Schritt 5 gehört zum *Postprocessing*.

In Kapitel 6 erläutern wir Kriterien für die Bestimmung der Güte einer Clusterung.

In Kapitel 7 stellen wir eine heuristische Methode zur Bewertung der Qualität des Empfehlungssystems vor.

In Kapitel 8 werden die Verfahren zusammengefasst und Simulationsergebnisse diskutiert.

Im Kapitel 9 stellen wir eine Methode für die Clustersortierung vor.

Im Kapitel 10 gehen wir auf einige wichtige Aspekte der Implementierung ein.

Im Anhang werden die ersten 99 Cluster präsentiert.

2 Grundlegende Definitionen

Wir beginnen mit einer Reihe von grundlegenden Definitionen, um die wichtigsten Begrifflichkeiten für diese Arbeit abzustecken.

2.1 Grundbegriffe aus der Graphentheorie

Eine gute Einführung in die Graphentheorie bieten das Vorlesungsskript von [Reinelt] und das Buch von [Diestel:2006].

Ein **Graph** G ist ein Paar (V, E) bestehend aus einer nichtleeren Menge V und einer Menge E, die eine Teilmenge von $V \times V$ ist. Die Elemente $v \in V$ nennt man die **Knoten** (Vertices) und die Elemente $e \in E$ heißen die **Kanten** (Edges) des Graphen. Die Ordnung eines Graphen ist definiert als die Anzahl seiner Knoten: $|G| = |V|$. Ein Graph heißt **endlich**, wenn $|G| < \infty$ ist, andernfalls **unendlich**. Ein Knoten $v \in V$ und eine Kante $e \in E$ heißen **inzident** (oder inzidieren miteinander), wenn $v \in e$ ist. Die Menge aller mit v inzidierenden Kanten wird mit $E(v)$ bezeichnet. Der **Grad** eines Knotens ist die Anzahl der Kanten, mit denen er inzidiert. Wir bezeichnen den Grad eines Knoten v mit $\deg(v)$. Es gilt also

$$\deg(v) = |E(v)|.$$

Zwei *verschiedene* Knoten u, v von G sind **adjazent** (benachbart oder miteinander verbunden), wenn eine Kante $e \in E$ existiert mit $e = \{u, v\}$ oder $e = \{v, u\}$. Ist die Unterscheidung zwischen $\{u, v\}$ und $\{v, u\}$ irrelevant, so spricht man von einem **ungerichteten** Graph, andernfalls von einem **gerichteten** Graph (oder **Digraph**). Ein Graph ist **gewichtet**, falls jeder Kante $e \in E$ eine reelle Zahl $w(e) \in \mathbb{R}$ zugeordnet ist, andernfalls **ungewichtet**.

$\tilde{G} = (\tilde{V}, \tilde{E})$ ist ein **Teilgraph** von $G = (V, E)$, *kurz:* $\tilde{G} \subseteq G$, wenn sowohl $\tilde{V} \subseteq V$ als auch $\tilde{E} \subseteq E$ gilt.

Eine endliche Folge der Form

$$W = (v_0, e_1, v_1, e_2, \ldots, e_{k-1}, v_{k-1}, e_k, v_k), \quad k \geq 1$$

mit paarweise verschiedenen Knoten $\{v_1, \ldots, v_k\}$ und Kanten $e_i = \{v_{i-1}, v_i\}, i = 1, \ldots, k$, bildet einen **Weg**. Oft schreibt man auch *informell* $W = v_0 v_1 \ldots v_k$. Wir nennen W einen Weg von v_0 nach v_k. Dabei ist v_0 der **Anfangsknoten** und v_k der **Endknoten** des Weges. Die Anzahl k seiner Kanten ist seine **Länge**.

Zwei *verschiedene* Knoten $u, v \in G$ sind **zusammenhängend**, wenn es in G einen Weg von u nach v gibt. Falls jedes Paar von Knoten aus G zusammenhängend ist, so spricht man von einem **zusammenhängenden Graphen**.

Der **Abstand** $d(u, v)$ zwischen zwei *verschiedenen* Knoten $u, v \in G$ ist die Länge eines kürzesten Weges von u nach v. Falls kein solcher Weg existiert, wird $d(u, v) = \infty$ gesetzt.

Sei $G = (V, E)$ ein ungerichteter und ungewichteter Graph. Die Menge aller **(exklusiven) Nachbarn** eines Knotens $u \in V$ ist definiert als

$$n(u) := \big\{v \in V \big| d(u, v) = 1\big\}. \tag{1}$$

Wenn der Knoten u in der Menge seiner Nachbarn auch mitgezählt wird, so spricht man von der Menge der **inklusiven Nachbarn**

$$n_+(u) := \big\{v \in V \big| d(u, v) \leq 1\big\}. \tag{2}$$

Ist $W = v_0 v_1 \dots v_{k-1}$ ein Weg mit $k \geq 3$, so bildet die Folge $(W, \{v_{k-1}, v_0\}, v_0)$ einen **Kreis**. Enthält ein Graph keinen Kreis, so spricht man von einem **Wald**. Ein zusammenhängender Wald ist ein **Baum**.

Wir beschäftigen uns ausschließlich mit endlichen Graphen. Außerdem werden **Schlingen** (das sind Kanten, die einen Knoten mit sich selbst verbinden) und *Mehrfachkanten* zwischen zwei benachbarten Knoten ausgeschlossen, falls nicht ausdrücklich erwähnt.

Ein **bipartiter Graph** $B = (V_B, E_B)$ ist ein Graph, dessen Knotenmenge V_B in zwei disjunkte Teilmengen T, P mit $T \cup P = V_B$ und $T \cap P = \phi$ zerlegt werden kann, so dass keine zwei Knoten in T und keine zwei Knoten in P benachbart sind. Die Knotenmengen T und P nennt man eine **Bipartition**[5] von B.

Mit $L(T) = (deg(t_1), \dots, deg(t_l))$ bezeichnen wir die **Knotengradfolge** von T und mit $R(P) = (deg(p_1), \dots, deg(p_r))$ bezeichnen wir die Knotengradfolge von P. Man kann die Elemente aus T und P so anordnen, dass ihre Knotengradfolgen jeweils nicht-aufsteigend sind.

[5] In Anlehnung an die im nächsten Kapitel eingeführten Begriffe sei auf folgende Darstellung eines bipartiten Graphen B hingewiesen: (L)inks steht immer die Menge T der Transaktionen und (R)echts steht immer die Menge P der Produkte.

2.2 Zufallsgraphen und Netzwerkmotiv

Man kann allgemein für einen Graphen G die Menge aller Graphen $\mathcal{G}(G)$ betrachten, die bestimmte Struktureigenschaften von G erhalten. Wählt man rein zufällig einen Graphen G' aus $\mathcal{G}(G)$, so nennt man G' einen **zu G korrespondierenden Zufallsgraphen**.

Wir stellen bereits hier einige Modelle vor, auf die wir später in Kapitel 4 und Kapitel 5 zurückgreifen werden:

1. Gegeben sei ein Graph $G = (V, E)$ mit $n = |V|$ Knoten. Das Modell $\mathcal{G}(n, p_0)$, bekannt unter dem Namen Erdös-Rényi-Graphenmodell[6], enthält die Menge aller Graphen $G' = (V', E')$ mit folgenden Eigenschaften:
   (a) $|V'| = n$
   (b) Die Wahrscheinlichkeit, dass zwei beliebige Knoten $v, w \in V$ miteinander verknüpft sind, ist p_0:

$$prob\big((v, w) \in E'\big) = p_0 \qquad \forall v, w \in V \tag{3}$$

2. Gegeben sei ein bipartiter Graph $B = (T \cup P, E_B)$ mit der Knotengradfolge $R(P)$. Mit $\mathcal{G}(R)$ bezeichnen wir die Menge aller bipartiten Graphen $B' = (V'_B, E'_B)$ mit folgenden Eigenschaften:
   (a) $V'_B = T \cup P$
   (b) $|E'_B| = |E_B|$
   (c) $R'(P) = R(P)$
   (d) Jedes $p \in P$ ist mit der gleichen Wahrscheinlichkeit mit einem $t \in T$ verbunden:

$$prob\big((t, p) \in E'_B\big) = \frac{deg_B(p)}{|T|} \qquad \forall t \in T, p \in P \tag{4}$$

   Zweig und Kaufmann [Zweig:2011] nennen $\mathcal{G}(R)$ das **simple bipartite random graph model (SiRaBiG)**.

[6] Siehe Fortunato [Fortunato:2010], S. 94.

3. Gegeben sei ein bipartiter Graph $B = (T \cup P, E_B)$ mit beiden Knotengradfolgen $L(T)$ und $R(P)$. Mit $\mathcal{G}(L, R)$ bezeichnen wir die Menge aller bipartiten Graphen $B' = (V'_B, E'_B)$ mit folgenden Eigenschaften:

   (a) $V'_B = T \cup P$
   (b) $|E'_B| = |E_B|$
   (c) $L'(T) = L(T)$
   (d) $R'(P) = R(P)$

   $\mathcal{G}(L, R)$ enthält demnach alle Graphen, die die gleiche Knotenmenge, die gleiche Anzahl an Kanten und die gleiche Knotengradfolgen auf beiden Seiten haben wie B. Nur die Positionen der Kanten können variieren. Zweig und Kaufmann [Zweig:2011] nennen $\mathcal{G}(L, R)$ das **fixed degree sequence model (FDSM)**.

Ein **Netzwerkmotiv** ist ein Teilgraph, der in einem Graphen G signifikant öfter vorkommt als in einem korrespondierenden Zufallsgraphen $G' \in \mathcal{G}(G)$ [Zweig:2011]. Auf die Bedeutung von *"signifikant oft"* gehen wir in Abschnitt 4.2 genauer ein.

2.3 Partitionierungen

Für die Bildung von Clustern im weiteren Verlauf der Arbeit werden wir zwei wichtige Begriffe benötigen: Knoten- und Kantenpartitionierung.

Sei $G = (V, E)$ ein Graph mit $n = |V|$ Knoten und $m = |E|$ Kanten.

Durch eine disjunkte Zerlegung der Knotenmenge $V = V_1 \cup \ldots \cup V_N,\ V_i \cap V_j = \emptyset\ \forall\ i \neq j$, erhält man eine **Knotenpartitionierung**

$$\mathcal{K}_V := \{K_1, ..., K_N\} \text{ mit Teilgraphen } K_\mu = (V_\mu, E_\mu).$$

Eine **Kantenpartitionierung**

$$\mathcal{C}_E := \{C_1, ..., C_M\}$$

erhält man durch eine disjunkte Zerlegung der Kantenmenge $E = E_1 \cup \ldots \cup E_M,\ E_i \cap E_j = \emptyset\ \forall\ i \neq j$. Die Partitionen sind die zugehörigen Teilgraphen

$$C_\nu := (V_\nu, E_\nu),\ \text{wobei } V_\nu := \bigcup_{e_{ij} \in E_\nu} \{i, j\} \quad (\nu = 1, \ldots, M).$$

Jede Kante aus G kann nur in einer Kantenpartition enthalten sein kann. Zwei Kantenpartitionen, die *mindestens* einen gemeinsamen Knoten haben, heißen **benachbart**.

3 Warenkorbanalyse und Assoziationsanalyse

Einkaufshäuser und Supermärkte verfügen mit jedem Einkaufsbeleg über eine schier endlose Menge an Informationen, aus denen man das Konsumverhalten der Verbraucher und eventuelle Beziehungen zwischen den einzelnen Produkten extrahieren kann. Das ist die Aufgabe der sogenannten **Warenkorbanalyse (market basket analysis)**, die versucht, Fragen von der Art

- *Mit welchen Waren befülle ich meine Verkaufsregale optimal? Oder: Welche Waren sollte ich aus dem Sortiment entfernen?*
- *Für welches Produkt A sollte ich Rabatt geben, um den Umsatz eines anderen Produktes B zu erhöhen? Oder: Welche Produkte werden Umsatzeinbrüche erleiden, wenn der Preis eines bestimmten Produktes steigt?*
- *In welche Kategorien kann ich meine Kunden einordnen?*

zu beantworten, damit der Einzelhandel seinen Gewinn maximieren kann.

Die Warenkorbanalyse verwendet dabei Regeln, die z.B. folgende Form haben:

Beispiel 1: *Jemand, der Brot und Butter kauft, würde auch mit 66% Wahrscheinlichkeit Milch kaufen. Diese Regel trifft auf 50% der Kunden zu.*

Jeder Einkaufsbeleg stellt einen Warenkorb dar. Der Einfachheit halber interessieren wir uns nur dafür, was jeder Verbraucher gekauft hat, nicht aber wieviel er von jedem Produkt gekauft hat. D.h. wir nehmen an, dass jedes Produkt in einem Warenkorb entweder gar nicht oder genau einmal vorhanden ist. (Insofern wird ein Produkt technisch als ein binäres Attribut aufgefasst.)

Formell lässt sich die Warenkorbanalyse verallgemeinern auf die **Assoziationsanalyse**. Dabei nennt Agrawal [Agrawal:1993] jeden Warenkorb eine **Transaktion** und die Menge der Produkte heisst **Itemset**. Eine Regel von obiger Form heißt Assoziationsregel. Die genauere Definition folgt.

Die beiden Begriffe Warenkorb und Transaktion werden oft synomym benutzt. Im Gegensatz zu den Begriffen Käufer, Kunde, Verbraucher oder Konsument spiegelt ein Warenkorb in der Regel genau einen Einkauf wider. Soll aber das langfristige Verhalten eines Verbrauchers untersucht werden, also die Gesamtheit aller seiner Einkäufe über einen längeren Zeitraum, so kann der Begriff Transaktion auch für den Verbraucher stehen. Diese Unterscheidung sollte aus dem Zusammenhang klar ersichtlich sein. Für unsere Filmdatenbank wäre die zweite Betrachtungsweise zweckmäßiger.

3.1 Mathematisches Modell

Mathematisch lassen sich die Menge der Transaktionen (**Transaktionsdatenbank**) $T = \{t_1, \ldots, t_l\}$ und die Menge der Produkte $P = \{p_1, \ldots, p_r\}$ durch einen bipartiten Graphen

$$B = (T \cup P, E)$$

modellieren [Zweig:2011]. Dabei existiere zwischen einer Transaktion t_j und einem Produkt p_x eine verbindende Kante $e = (t_j, p_x) \in E$ dann und nur dann, wenn p_x in der Transaktion t_j enthalten ist.

Fassen wir p_x als ein binäres Attribut auf, können wir jede Transaktion t_j auch als einen binären Vektor aus $\mathbb{R}^r$ betrachten, für den gilt

$$t_j[p_x] = \begin{cases} 1, & \text{falls } (t_j, p_x) \in E, \\ 0 & \text{sonst.} \end{cases}$$

Reihen wir diese als Zeilenvektoren untereinander auf, erhalten wir die **Adjazenzmatrix** $\boldsymbol{A}_B \in \mathbb{R}^{l \times r}$ des bipartiten Graphen B.

Beispiel: Bipartiter Graph B mit zugehöriger Adjazenzmatrix $\boldsymbol{A}_B$:

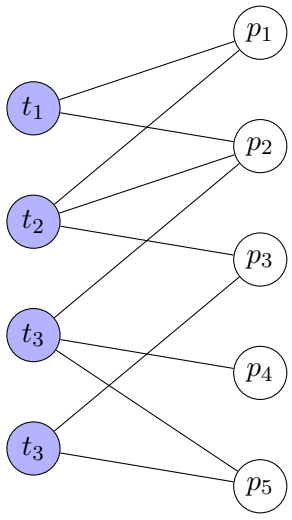


Abbildung 2: $B = (T \cup P, E)$

	p_1	p_2	p_3	p_4	p_5	L
t_1	1	1	0	0	0	2
t_2	1	1	1	0	0	3
t_3	0	1	0	1	1	3
t_4	0	0	1	0	1	2
R	2	3	2	1	2	10

Die Knotengradfolge $L = (2, 3, 3, 2)$ besteht gerade aus der Folge der Zeilensummen und die Knotengradfolge $R = (2, 3, 2, 1, 2)$ besteht gerade aus der Folge der Spaltensummen der Adjazenzmatrix $\boldsymbol{A}_B$. Und die Gesamtsumme der Zeilensummen (bzw. Spaltensummen) ist identisch mit der Gesamtzahl aller Einsen = $|E|$ = Gesamtzahl aller Kanten von E.

Wir stellen fest, dass $deg(p_x) \leq l$ für alle $p_x \in P$ gilt, da ein Produkt höchstens in l Warenkörben landen kann. Außerdem kann ein Warenkorb höchstens r Produkte aufnehmen: $deg(t_j) \leq r$ für alle $t_j \in T$.

Unter dem **Support** $supp(p_x)$ **eines Produktes** $p_x \in P$, verstehen wir den Anteil der Transaktionen, die p_x enthalten. Damit gilt:

$$supp(p_x) = \frac{deg(p_x)}{l} = \frac{deg(p_x)}{|T|} \tag{5}$$

Bemerkung: In einem bipartiten Graphen, in der ausschließlich die beiden Knotengradfolgen bekannt sind, ist dies die Wahrscheinlichkeit, dass eine Transaktion das Produkt p_x enthält [Zweig:2011].

Unter dem Begriff **Co-occurence** (Zusammentreffen) **zweier verschiedener Produkte** p_x und p_y verstehen wir die *absolute Anzahl* der Transaktionen t_j, die *beide* Produkte p_x und p_y enthalten: Sei $p \in P$ und $n(p) = \{t \in T | (t,p) \in E\}$ die **Nachbarschaft** von p im bipartiten Graphen B. Dann ist

$$coocc(p_x, p_y) = |n(p_x) \cap n(p_y)|. \tag{6}$$

Der Anteil der Transaktionen, die die beiden Produkte p_x und p_y enthalten, liefert demnach eine Definition für den **Support zweier verschiedener Produkte** p_x und p_y:

$$supp(p_x, p_y) = \frac{coocc(p_x, p_y)}{|T|} \tag{7}$$

Bemerkung: Dies ist die Wahrscheinlichkeit, dass ein rein zufällig ausgewählter Warenkorb die beiden Produkte p_x und p_y enthält.

Die beiden letzten Definitionen lassen sich in natürlicher Weise auf disjunkte Teilmengen $X, Y \subset P$ verallgemeinern:

$$coocc(X) = \left|\{t \in T : (t,p) \in E \quad \forall p \in X\}\right| \tag{8}$$

$$coocc(X,Y) = coocc(X \cup Y) = \left|\{t \in T : (t,p) \in E \quad \forall p \in X \cup Y\}\right| \tag{9}$$

Und für den Support gilt entsprechend:

$$supp(X) = \frac{coocc(X)}{|T|} \tag{10}$$

$$supp(X,Y) = supp(X \cup Y) = \frac{coocc(X,Y)}{|T|} \tag{11}$$

Übertrifft $supp(X)$ eine gegebene untere Schranke σ, d.h. erscheinen die Produkte aus X in mindestens $\sigma\%$ aller Transaktionen, so nennt man X eine häufig auftretende Itemset **(frequent itemset)**[7]. In diesem Fall nennt man die untere Schranke σ auch $minsupp(X)$ (minimaler Support von X).

3.2 Assoziationsregel

Seien X und Y zwei disjunkte Teilmengen von P. Eine **Assoziationsregel** ist eine Implikation von der Form $X \Longrightarrow Y$, gepaart mit zwei weiteren Kennzahlen:

- **Support** $0 < s < 100\%$ **einer Assoziationsregel**:

  $$s = supp(X \Longrightarrow Y) = supp(X \cup Y)$$

  In Anlehnung an die obige Schreibweise für zwei einzelne Produkte schreiben wir dafür auch

  $$s = supp(X, Y) = \frac{coocc(X, Y)}{|T|}$$

  .

- (Minimaler) **Konfidenzfaktor** $0 < c < 100\%$ **einer Assoziationsregel**:
  Die Regel $X \Longrightarrow Y$ ist in der Transaktionsdatenbank T mit dem Konfidenzfaktor c erfüllt genau dann, wenn mindestens $c\%$ der Transaktionen, die die Produkte aus X enthalten, auch die Produkte aus Y enthalten, oder anders formuliert:

  $$c \leq \frac{supp(X, Y)}{supp(X)}$$

Man nennt X auch die **Antezedenz** und Y die **Konsequenz** der Assoziationsregel.

Beispiel 1:

Die Antezendenz ist $X = \{Brot, Butter\}$ *und die die Konsequenz ist* $Y = \{Milch\}$*, die mit einer Wahrscheinlichkeit von* $c = 0.66$ *eintritt. Diese Regel trifft auf* $s = 50\%$ *der Transaktionen zu.*

3.3 Ein Algorithmus zum Auffinden von Assoziationsregeln

Die klassische Warenkorbanalyse hat zum Ziel, für eine gegebene Transaktionsdatenbank T alle Assoziationsregeln zu finden, die einen Support und eine Konfidenz haben, die größer

[7] Die Definition von "häufig" wird hier willkürlich durch die Wahl von σ festgelegt!

sind als die vom Benutzer vorgegebenen Minimalwerte für σ beziehungsweise c. Das Lehrbuch [Ester:2000] gibt eine gute Einführung in das Thema. Der Vollständigkeit halber geben wir hier nur die Grundideen der Methode aus dem Artikel [Agrawal:1993] wieder.

Die Aufgabe, häufig als das "Warenkorbproblem" bezeichnet, kann in zwei Teilaufgaben aufgespalten werden:

1.) Finde alle häufig auftretenden Itemsets $X \subset P$, die einen minimalen Support σ vorweisen.

2.) Bilde aus den gefundenen häufig auftretenden Itemsets X Assoziationsregeln der Form $A \Longrightarrow X \backslash A$, die mindestens den Konfidenzfaktor c erfüllen. Dabei sei A eine nicht-leere echte Teilmenge von X.

Zur Lösung der Teilaufgabe 1.) wäre es viel zu aufwändig, alle $2^{|P|}$ möglichen Teilmengen von $|P|$ in Erwägung zu ziehen. Der wohlbekannte Apriori-Algorithmus ([Agrawal:1993]) macht sich folgende Eigenschaft von häufig auftretenden Itemsets zunutze:

Jede Teilmenge eines häufig auftretenden Itemsets muss selbst wiederum ein häufig auftretendes Itemset sein.

Man bestimmt also die häufig auftretenden Itemsets der Größe k nach, indem man bei $k = 1$ anfängt und alle einelementigen Teilmengen auf den minimalen Support hin untersucht. Im Schritt $k + 1$ müssen nicht mehr alle Teilmengen der Kardinalität $k + 1$ untersucht werden, sondern nur solche, die aus der Vereinigung (*Join*) von zwei k-elementigen häufig auftretenden Itemsets entstehen.

Es versteht sich von selbst, dass die in Teilaufgabe 2.) gefundenen Assoziationsregeln automatisch den minimalen Support σ vorweisen können. Für die Prüfung der Konfidenz müssen die Terme

$$\frac{supp(A \cup (X \backslash A))}{supp(A)} = \frac{supp(X)}{supp(A)}$$

ausgewertet werden. Da der Support aller bisher ermittelten häufig auftretenden Itemsets schon bekannt ist, müssen keine neuen Datenbankdurchläufe vollzogen werden.

Es müssen auch nicht alle gefundenen Regeln auf ihre Konfidenz geprüft werden: Wenn eine Regel der Form $A \Longrightarrow X \backslash A$ den minimalen Konfidenzfaktor nicht erfüllt, dann kann auch keine Regel der Form $B \Longrightarrow X \backslash B$ mit $B \subset A$ diese Bedingung erfüllen.

3.4 Die Problematik

Die Art und Weise, wie die Assoziationsregeln bisher bewertet werden, weisen noch eine Reihe von Defiziten auf, die im Artikel [Raeder:2010] aufgelistet werden.

Die Transaktionen enthalten in der Regel Informationen auf dem Niveau von einzeln aufgelisteten Produkten (Kassenbons), so dass die meisten Assozationsregeln nur sehr geringen Support haben. Es gibt keine offensichtliche Methode um gute Werte für den minimalen Support σ und für den Konfidenzfaktor c zu wählen.

Wählt man σ zu hoch, so findet die klassiche Methode nur sehr wenige Assoziationsregeln und man verliert möglicherweise interessante Zusammenhänge. Wählt man σ zu niedrig, erhält man eine unüberschaubare Menge von schwachen Assoziationsregeln, die sehr wahrscheinlich überflüssige Informationen enthalten. Eine Abhilfe gegen Redundanz schaffen zwar Techniken zum Auffinden von *maximalen*[8] oder *geschlossenen*[9] Itemsets, doch diese erweisen sich als abhängig von der Zusammensetzung der Daten.

Im Allgemeinen verändern sich die Assoziationsregeln sehr empfindlich gegenüber kleinsten Veränderungen in σ oder c.

Zur Bewertung einer Assoziationsregel $X \Longrightarrow Y$ benötigen wir daher eine berechenbare Größe, die eine Aussage darüber macht, wie bedeutsam diese Regel in der Transaktionsdatenbank T hervorsticht.

Die Assoziationsregel im obigen Beispiel 1 wäre überhaupt nicht interessant, wenn ohnehin mehr als 66% der Transaktionen das Produkt $Y = \{Milch\}$ enthalten würden.

3.5 Interessantheitsmaß einer Assoziationsregel

Ein **Interessantheitsmaß** für eine Assoziationsregel sollte ein Maß für die *Abweichung von der stochastischen Unabhängigkeit* sein und gewisse *Monotonieeigenschaften* erfüllen. Siehe die Artikeln [Zweig:2011] und [Piatetsky-Shapiro:1991] für eine detaillierte Darstellung.

Der Einfachheit halber beschränken wir unsere Betrachtungen von Assoziationsregeln im Folgenden auf den Fall einelementiger Teilmengen von P, es sei also $X = \{p_x\}$ und $Y = \{p_y\}$.

[8] I heisst **geschlossen**, wenn es keine Obermenge $J \supset I$ gibt, so dass $supp(I) = supp(J)$ ist.

[9] I heisst **maximal** auf dem Supportlevel σ_0, wenn es keine Obermenge $J \supset I$ gibt mit Mindest-Supportlevel σ_0.

Stochastische Unabhängigkeit

Es sei daran erinnert (vgl. [Krengel:2005]), dass für die **bedingte Wahrscheinlichkeit** $prob(B|A)$ eines Ereignisses B bei gegebenem A ($prob(A) > 0$) gilt

$$prob(B|A) = \frac{prob(A \cap B)}{prob(A)}. \tag{12}$$

Ist das Ereignis B unabhängig von A, so gilt

$$prob(B|A) = prob(B). \tag{13}$$

Die Definition von $prob(B|A)$ setzt voraus, dass $prob(A) > 0$ ist. Deshalb wird die folgende Gleichung benutzt, um die **stochastische Unabhängigkeit** zweier Ereignisse A und B zu definieren:

$$prob(A \cap B) = prob(A) \cdot prob(B) \tag{14}$$

Ist $prob(A) > 0$, so ist (14) äquivalent zu (13).

Auf die Warenkorbanalyse bezogen können wir folgende Ereignisse betrachten:

Seien p_x und p_y zwei verschiedene Produkte aus P.

Ereignis A:	Eine Transaktion enthält das Produkt p_x.
Ereignis B:	Eine Transaktion enthält das Produkt p_y.
Ereignis $A \cap B$:	Eine Transaktion enthält die beiden Produkte p_x und p_y.

Offensichtlich gelten dann nach (5) und (7)

$$prob(A) = supp(p_x) = \frac{deg(p_x)}{|T|}, \qquad prob(B) = supp(p_y) = \frac{deg(p_y)}{|T|} \tag{15}$$

und

$$prob(A \cap B) = supp(p_x, p_y) \tag{16}$$

Sind die beiden Ereignisse A und B unabhängig, d.h. hängen die beiden Produkte p_x und p_y nicht zusammen, so gilt

$$prob(A \cap B) = supp(p_x) \cdot supp(p_y) = \frac{deg(p_x) \cdot deg(p_y)}{|T|^2} \tag{17}$$

Bekannte Interessantheitsmaße sind unter anderem der in Piatetsky-Shapiro [Piatetsky-Shapiro:1991] eingeführte **Leverage**

$$\begin{aligned} leverage(p_x, p_y) &= supp(p_x, p_y) - supp(p_x) \cdot supp(p_y) \\ &= \frac{coocc(p_x, p_y)}{|T|} - \frac{deg(p_x) \cdot deg(p_y)}{|T|^2} \end{aligned} \tag{18}$$

und die in Brin et. al. [Brin:1997] eingeführte **Conviction**

$$conviction(p_x, p_y) = \frac{supp(\{p_x\}) \cdot supp(\{p_y\}^C)}{supp(\{p_x\}, \{p_y\}^C)} \tag{19}$$

wobei mit $\{p_y\}^C$ die zu $\{p_y\}$ komplementäre Menge $P \backslash \{p_y\}$ gemeint ist.

Für zwei unkorrelierte Produkte p_x und p_y sollte

$$leverage(p_x, p_y) = 0$$

bzw.

$$conviction(p_x, p_y) = 1$$

gelten.

Treten X und Y dagegen überdurchschnittlich oft zusammen auf, also mehr als wir es von zwei unabhängigen Produkten erwarten würden, so ist $leverage(p_x, p_y) > 0$ bzw. $conviction(p_x, p_y) > 1$.

In beiden Definitionen (18) und (19) wird der *tatsächlich beobachtete Support* von zwei Teilmengen $X, Y \subset P$ mit dem Support von X und Y verglichen, den man *erwarten würde, wenn X und Y stochastisch unabhängig voneinander wären.* Dieses Modell der stochastischen Unabhängigkeit nennt man **stochastic independence model (SIM)**.

Interessantheitsmaße, die auf diesem Modell beruhen, liefern immer noch eine sehr große Anzahl von Assoziationsregeln. Desweiteren lassen sich solche Ergebnisse nicht immer klar interpretieren [Brin:1997].

Wir werden im nächsten Kapitel ein praxistauglicheres Modell der stochastischen Unabhängigkeit aus der Sicht der Netzwerkanalyse kennenlernen, um damit das Interessantheitsmaß Leverage neu zu definieren.

Es gibt viele verschiedene Interessantheitsmaße in der Literatur, und diese verschiedenen Maße liefern Assoziationsregeln, die unter Umständen widersprüchlich sein können [Tan:2004]. Es ist also nicht ganz klar, welches Interessantheitsmaß vorzuziehen ist, und auch nicht klar, wie eine minimale Schranke zu wählen ist.

Es ist auch nicht besonders hilfreich, eine Unmenge von Assoziationsregeln zu haben, deren Anzahl die Anzahl der Produkte weit übertrifft. Nützlicher wäre eine Klassifizierung der Produktmenge in ähnlichen Produkten nach dem Vorbild der *Community Detection* in sozialen Netzwerken, in denen bestimmte Gruppen von Teilnehmern (z.B. Freundeskreise) dichter miteinander verbunden sind als mit dem Rest des Netzwerks. Doch um die Methoden der *Clusteranalyse* (siehe Kapitel 5) auf der Produktmenge P anwenden zu können, muss zunächst ein Zusammenhang zwischen den einzelnen Elementen von P hergestellt werden. Darum geht es im nächsten Kapitel.

4 Einseitige Projektion (One Mode Projection)

Die **einseitige Projektion** (**one mode projection** oder kurz **OMP**) bildet einen bipartiten Graphen

$$B = (T \cup P, E)$$

auf einen neuen Graphen

$$G = (V, E_G)$$

ab. Dabei ist je nachdem, für welche Zusammenhänge wir uns interessieren, entweder $V = T$ oder $V = P$ zu wählen. Da wir ein Produktempfehlungssystem entwerfen wollen, beschreiben wir die Methode für den Fall $V = P$.

Die entscheidende Frage ist nun: Wie werden die Kanten von G gebildet?

4.1 Einfache OMP

Die einfachste Möglichkeit einen Zusammenhang zwischen den Elementen in der Produktemenge P herzustellen besteht darin, zwei verschiedene Produkte p_x und p_y immer dann durch eine Kante zu verbinden, wenn p_x und p_y in mindestens $\sigma\%$ aller Transaktionen $t \in T$ gemeinsam aufgetreten sind [Raeder:2010]:

$$(p_x, p_y) \in E_G \quad \Leftrightarrow \quad coocc(p_x, p_y) \geq \sigma \cdot |T|$$

Dies entspricht der Wahl einer Assoziationsregel mit minimalem Support.

4.2 Signifikanz eines Netzwerkmotivs

Wir haben gesehen, dass das Zusammentreffen (Co-occurence) zweier Produkte eine wesentliche Rolle spielt bei der Festlegung, ob zwei Produkte miteinander zusammenhängen. Aus der Sicht der Netzwerkanalyse kann man das Zusammentreffen auch als eine Art von Netzwerkmotiv (siehe Abschnitt 2.1) in einem passenden Graphenmodell $\mathcal{G}(B)$ interpretieren.

Statistische Signifikanz

Zur Beurteilung, ob $coocc(p_x, p_y)$ in B signifikant ist, d.h. ob $coocc(p_x, p_y)$ charakteristisch ist für B, wird dieser Wert verglichen mit dem zu erwarteten Wert von $coocc(p_x, p_y)$ in einem zu B korrespondierenden Zufallsgraphen $B' \in \mathcal{G}(B)$:

$$leverage(p_x, p_y) = coocc(p_x, p_y) - \mathbb{E}[coocc_{\mathcal{G}(B)}(p_x, p_y)] \tag{20}$$

Ist das Ergebnis dieser Auswertung positiv, werden die beiden Produkte p_x und p_y im projizierten Graphen G miteinander verbunden. Zwei auf diese Weise verbundene Produkte nennt man auch "**befreundet**" in Anlehnung an soziale Netzwerke. Und das Interessantheitsmaß steht sozusagen für den Grad der Freundschaft.

Damit wird ein Interessantheitsmaß definiert, der auf einer stochastischen Unabhängigkeit im Sinne der Netzwerkanalyse basiert. Die Wahl eines *passenden* Graphenmodells ist von entscheidender Bedeutung, um die Signifikanz eines Netzwerkmotivs korrekt auszuwerten.

4.3 Das SiRaBiG-Graphenmodell

In ihrer Arbeit [Zweig:2011] haben Zweig und Kaufmann gezeigt, dass man mit der Wahl

$$\mathcal{G}(B) = \mathcal{G}(R)$$

den Erwartungswert

$$\mathbb{E}[coocc_{\mathcal{G}(R)}(p_x, p_y)] = \frac{deg(p_x) \cdot deg(p_y)}{|T|^2} \tag{21}$$

erhält. Das SiRaBiG-Graphenmodell für die Netzwerk-basierte Methode ist also äquivalent mit der SIM-Methode.

Das folgende Beispiel in Abbildung 3 aus dem Artikel [Gionis:2007] zeigt einleuchtend, dass mit der SiRaBiG-basierten Methode, die ja nur die Knotengradfolge auf der einen Seite (hier: Produktseite) erhält, die Signifikanz nicht immer richtig berechnet wird. Erinnern wir uns daran (vgl. Abschnitt 2.3.1), dass jeder Graph B durch eine $l \times r$-Adjazenzmatrix $\boldsymbol{A}_B$ mit Einträgen 0 oder 1 dargestellt werden kann. Die beiden Produkte p_x und p_y werden in beiden Transaktionsmengen B_1 und B_2 gleich oft zusammen konsumiert. Man bekäme in beiden Fällen einen hohen Wert für das Interessantheitsmaß $leverage(p_x, p_y)$. Doch in der Transaktionsmenge B_2 ist der starke Zusammenhang nicht auf die Eigenheiten der beiden Produkte, also etwa auf den ähnlichen Inhalt zweier Filme zurückzuführen, sondern eher darauf, dass diese Kunden ohnehin fast alle Produkte kaufen bzw. fast alle Filme anschauen und bewerten.

B_1	p_x	p_y							
t_1	1	1	0	0	1	0	0	1	1
t_2	1	1	1	1	0	0	1	0	0
t_3	1	1	0	0	0	1	0	1	1
t_4	1	1	0	1	1	0	1	0	1
t_5	1	1	0	1	0	0	0	0	1
t_6	1	1	1	0	1	0	0	1	0
t_7	1	0	0	0	0	1	1	0	0
t_8	1	0	0	1	1	0	0	0	1
t_9	0	1	0	0	1	1	0	0	0
t_{10}	0	1	1	0	0	1	0	0	1
t_{11}	0	0	0	0	1	0	1	0	0
t_{12}	0	0	0	1	1	0	1	0	0

B_2	p_x	p_y							
t_1	1	1	1	1	1	1	1	1	1
t_2	1	1	1	1	1	1	1	1	1
t_3	1	1	0	1	1	1	1	1	1
t_4	1	1	1	1	1	1	1	1	1
t_5	1	1	1	1	1	1	0	1	1
t_6	1	1	1	1	1	1	1	1	1
t_7	1	0	0	0	0	1	1	0	0
t_8	1	0	0	1	1	0	0	0	1
t_9	0	1	0	0	1	1	0	0	0
t_{10}	0	1	1	0	0	1	0	0	1
t_{11}	0	0	0	0	1	0	1	0	0
t_{12}	0	0	0	1	1	0	1	0	0

Abbildung 3: Beispiel für eine unangemessene Signifikanz nach Gionis et. al. [Gionis:2007]

Außerdem wird durch die Bedingung (4) immer angenommen, dass jeder Film mit der gleichen Wahrscheinlichkeit von allen Kunden bewertet wird. Das kann aber nicht zutreffen, da diese Wahrscheinlichkeit zunimmt, je mehr Filme ein Kunde angeschaut hat.

Eine theoretische Begründung, warum die SIM-Methode für die meisten realen Anwendungen nicht geeignet ist, kann in Zweig und Kaufmann [Zweig:2011] nachgelesen werden.

4.4 Das FDSM Graphenmodell

Die Autoren von [Gionis:2007] und [Zweig:2011] argumentieren, dass die Erhaltung beider Knotengradfolgen angemessener ist und schlagen daher die Wahl

$$\mathcal{G}(B) = \mathcal{G}(L, R)$$

vor. Für Datensätze, bei denen die Knotengrade eine *long tail distribution* aufweisen, hat Zweig in [Zweig:2010] gezeigt, dass der Erwartungswert über FDSM berechnet werden muss. So gibt es in der Netflix Filmdatenbank einige wenige Kunden, die sehr viele (fast alle) Filme angeschaut haben, während der Großteil der Kunden nur wenige Filme angeschaut haben.

Der Erwartungswert lässt sich allerdings nicht mehr so einfach wie in (21) berechnen. Zunächst muss die Menge

$$\tilde{\mathcal{G}}(L,R) = \{B' | B' \text{ ist eine Realisierung einer Menge aus } \mathcal{G}(L,R)\} \tag{22}$$

generiert werden. Natürlich ist $\tilde{\mathcal{G}}(L,R) \subset \mathcal{G}(L,R)$. Dann wird (klassisch) der Mittelwert über alle $coocc(p_x, p_y)$-Werte in der Menge der **Realisierungen (Stichproben)** $\tilde{\mathcal{G}}(L,R)$ gebildet, um den gesuchten Erwartungswert zu schätzen:

$$\mathbb{E}[coocc_{\mathcal{G}(L,R)}(p_x,p_y)] \approx \frac{1}{|\tilde{\mathcal{G}}(L,R)|} \sum_{B' \in \tilde{\mathcal{G}}(L,R)} coocc_{B'}(p_x,p_y) \tag{23}$$

Wir werden diesen Erwartungswert in Anlehnung an Zweig und Kaufmann [Zweig:2011] mit

$$\mathbb{E}[coocc_{\text{FDSM}}(p_x,p_y)]$$

und das damit berechnete Interessantheitsmaß (20) mit

$$leverage_{\text{FDSM}}(p_x,p_y)$$

bezeichnen.

Nun stellen sich automatisch zwei Fragen:

1. Wie lassen sich solche Stichproben $B' \in \mathcal{G}(L,R)$ generieren?
2. Wieviele solcher Stichproben sind notwendig, um einen guten Schätzer für den Erwartungswert zu bekommen?

Für die tatsächliche Größe von $\mathcal{G}(L,R)$ ist noch keine geschlossene Formel bekannt. Jeder Graph $B' \in \mathcal{G}(L,R)$ lässt sich als eine $l \times r$-Adjazenzmatrix $\boldsymbol{A}_{B'} = \left(a'_{ij}\right)_{\substack{1 \le i \le l \\ 1 \le j \le r}}$ mit Einträgen 0 oder 1 auffassen, die eine vorgegebene Zeilensummenfolge L und Spaltensummenfolge R besitzt. $|\mathcal{G}(L,R)|$ ist also gleich der Anzahl aller solcher Matrizen. Der Artikel [Greenhill:2008] befasst sich mit der asymptotischen Abschätzung dieser Größe für den Fall $l, r \longrightarrow \infty$ und beschränkter maximaler Zeilen- und Spaltensumme $z = \max\limits_{1 \le i \le l} \left\{ \sum\limits_{i=1}^{r} a'_{ij} \right\}$ und $s = \max\limits_{1 \le j \le r} \left\{ \sum\limits_{i=1}^{l} a'_{ij} \right\}$.

Man kann vereinfacht zusammenfassen: $|\mathcal{G}(L,R)|$ wächst exponentiell mit $z \cdot s$.

In dem Buch [Liu:2002] (*Abschnitt 4.3 Counting 0-1 Tables with Fixed Margins*) gibt es ein Beispiel eines solchen Graphenmodells mit $l = 13$, $r = 17$, $z = 10$ und $s = 14$. Der

geschätzte Wert für $|\mathcal{G}(L,R)|$ ist 6.72×10^{16}. Obwohl l und r hier nicht besonders groß sind, ist es unmöglich, dass wir nur annähernd so viele Stichproben erzeugen können. Daher ist es wichtig, möglichst *gleichwahrscheinliche Stichproben* zu generieren.

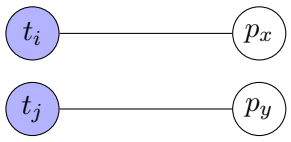
4.5 Markov-Ketten-Monte-Carlo (MCMC) und lokale Swaps

Die **Monte-Carlo-Simulation** ist eine bewährte Methode aus der Stochastik, bei dem sehr häufig durchgeführte Zufallsexperimente die Grundlage bildet. Wir verfolgen den in Cobb und Chen [Cobb:2003] und Gionis et. al. [Gionis:2007] beschriebenen Ansatz eines *Markov-Ketten-Monte-Carlo-Verfahrens*, um durch lokale Kantenvertauschungen aus dem Anfangszustand $B_0 = B$ eine Folge von Stichproben $B_0, B_1, B_2, ... \in \mathcal{G}(L,R)$ (*Markov-Kette*) zu konstruieren, die die gewünschte Gleichverteilung als ihre Grenzverteilung aufweist. Nach einer großen Anzahl von Konstruktionsschritten erhält man einen Zustand, den man als Stichprobe von $\mathcal{G}(L,R)$ benutzt. Die Qualität dieser Stichprobe sollte mit zunehmender Anzahl der Schritte steigen. Das Buch [Serfozo:2009] gibt eine solide Einführung in das Gebiet der stochastischen Prozesse.

Konstruktion mittels Swaps

Sei $B = (T \cup P, E) \in \mathcal{G}(L,R)$. Wählt man *nach dem Zufallsprinzip (d.h. zufällig mit gleicher Wahrscheinlichkeit)* jeweils zwei Knoten $t_i, t_j \in T$ und $p_x, p_y \in P$, so können die beiden Knoten in T entweder gar nicht oder auf verschiedene Weise durch maximal vier Kanten mit den beiden Knoten in P verbunden sein. Doch nur die beiden in Abbildung 4 dargestellten Fälle erlauben eine echte Kantenvertauschung (**lokaler Swap**), die die Knotengradfolgen L und R erhalten.

Zwei Knotenpaare in B:



Neue Knotenpaare B':

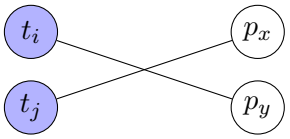


Abbildung 4: Swapbares Muster in B

Formell lässt sich ein lokaler Swap so ausdrücken:

$$B' \longleftarrow B \setminus \{(t_i, p_x), (t_j, p_y)\} \cup \{(t_i, p_y), (t_j, p_x)\} \tag{24}$$

Für die Adjazenzmatrix $\boldsymbol{A}_B$ bedeutet dies, dass die beiden zugehörigen Zeilen vertauscht werden, vorausgesetzt die Zeilen- und Spaltensummen bleiben erhalten:

$$\sum_j a_{ij} = \sum_j a'_{ij} \quad \forall i \qquad \text{und} \qquad \sum_i a_{ij} = \sum_i a'_{ij} \quad \forall j. \tag{25}$$

Abbildung 5: Swap von $\boldsymbol{A}_B$ nach $\boldsymbol{A}_{B'}$

Für beliebige Knotenpaare lässt sich dieser lokale Swap nicht durchführen, ohne die Bedingungen zur Knotengraderhaltung zu verletzen.

Abbildung 6: Nicht swapbares Muster

Wir geben zunächst einen naiven Algorithmus an, der ausgehend von $B_0 = B$ eine Folge $\{B_j\}_{j\geq 0}$ mit $B_j \in \mathcal{G}(L, R)$ erzeugt.

Eingabe:	Graph $B = (T \cup P, E) \in \mathcal{G}(L, R)$,
	Anzahl der randomisierten Swap-Schritte k_{swaps}
Ausgabe:	Graph $B' \in \mathcal{G}(L, R)$

function `naive_swaps` (B,k_{swaps})

1: `Setze` $B' = B$
2: **while** $k_{swaps} > 0$, **do**
3: **repeat** // *(Swapbarkeitstest)*
4: `Wähle nach dem Zufallsprinzip` $(t_i, p_x), (t_j, p_y) \in E$
5: **until** $((t_j, p_x) \notin E$ und $(t_i, p_y) \notin E)$
6: `Setze` $B' = B \setminus \{(t_i, p_x), (t_j, p_y)\} \cup \{(t_i, p_y), (t_j, p_x)\}$
7: `Setze` $k_{swaps} = k_{swaps} - 1$
8: **end while**
9: `Gebe` B' `zurück.`

Algorithmus 1 : Naiver Swap-Algorithmus

Markov-Ketten

Unter einer **endlichen Markov-Kette** $\mathcal{M}(\mathcal{S}, \mathbf{P})$ verstehen wir eine Folge von zeitdiskreten Zufallsexperimenten $X = (X^{(t)})_{t=0,1,2,\ldots}$ mit Werten im endlichen Zustandsraum (*state space*) $\mathcal{S} = \{s_1, \ldots, s_m\}$, deren Elemente **Übergangswahrscheinlichkeiten** p_{ij} vom Zustand i in den Zustand j haben, die ausschließlich vom vorangegangen[10] Zeitpunkt abhängt:

$$p_{ij} = prob(X^{(t+1)} = s_j | X^{(t)} = s_i) \tag{26}$$

Die Markov-Kette ist **homogen**, wenn p_{ij} für alle t gleich ist.

Die $m \times m$-Matrix

$$\mathbf{P} = \big(p_{ij}\big)_{i,j=1,\ldots,m}$$

ist eine stochastische Matrix (d.h. $p_{ij} \geq 0$ und $\sum_j p_{ij} = 1 \ \ \forall i$) und heisst **Übergangsmatrix**.

In unserem Fall ist $\mathcal{S} = \mathcal{G}(L, R)$. Alternativ kann man $\mathcal{S}$ auch als die Menge aller Adjazenzmatrizen $\mathcal{A}(L, R) = \{\mathbf{A}_B : B \in \mathcal{G}(L, R)\}$ auffassen. Übergänge von einem Zustand in einen anderen Zustand lassen sich durch einen lokalen Swap oder durch eine Folge von lokalen Swaps realisieren. Bezeichnen wir mit $\mathcal{T}$ die Menge aller lokalen Swaps. Dann kann man unsere Markov-Kette auch als einen **Übergangsgraphen** mit Knotenmenge $\mathcal{S}$ und Kantenmenge $\mathcal{T}$ darstellen:

$$G_{\mathcal{M}} = (\mathcal{S}, \mathcal{T}) \tag{27}$$

[10] Diese Eigenschaft nennt man *Gedächtnislosigkeit*.

Dabei sind zwei Knoten $s_1, s_2 \in \mathcal{S}$ genau dann verbunden, wenn es einen lokalen Swap vom Zustand s_1 in den Zustand gibt s_2.

Um gleichwahrscheinliche Stichproben zu erhalten, müssen gewisse Bedingungen an die Markov-Kette gestellt werden, für die wir ein paar weitere Begriffe definieren wollen.

Eine Markov-Kette heisst **irreduzibel**, wenn für alle $s_i, s_j \in \mathcal{S}$ eine positive Zahl $k = k(s_i, s_j)$ existiert, sodass

$$prob(X^k = s_j \big| X^0 = s_i) > 0. \tag{28}$$

D.h. jeder Zustand lässt sich von jedem anderen Zustand mit positiver Wahrscheinlichkeit mit einer endlichen Anzahl von Schritten erreichen. Oder anders ausgedrückt: Der Übergangsgraph $G_M = (\mathcal{S}, \mathcal{T})$ ist stark[11] zusammenhängend.

Durch den Zeilenvektor $\boldsymbol{\pi}^{(0)} \in \mathbb{R}^m$ mit den Einträgen

$$\pi_i^{(0)} = prob(X^{(0)} = s_i) \tag{29}$$

wird ein **Wahrscheinlichkeitsvektor** ($\pi_i^{(0)} \geq 0$ und $\sum_{i=1}^{m} \pi_i^{(0)} = 1$) defininert, der die **Anfangsverteilung** der Zustände beschreibt. Kennt man die Zustandsverteilung $\boldsymbol{\pi}^{(0)}$ zum Zeitpunkt $t = 0$, so lässt sich mit Hilfe der Übergangsmatrix $\mathbf{P}$ die Zustandsverteilung nach einem Schritt berechnen. Für $j = 1, ..., m$ gilt nämlich:

$$\pi_j^{(1)} = \sum_{i=1}^{m} prob(X^{(1)} = s_j | X^{(0)} = s_i) \cdot prob(X^{(0)} = s_i) = \sum_{i=1}^{m} p_{ij} \cdot \pi_i^{(0)} \tag{30}$$

Es gilt also

$$\boldsymbol{\pi}^{(1)} = \boldsymbol{\pi}^{(0)} \cdot \mathbf{P}. \tag{31}$$

Und nach n Schritten gilt

$$\boldsymbol{\pi}^{(n)} = \boldsymbol{\pi}^{(0)} \cdot \mathbf{P}^n. \tag{32}$$

Für die Elemente der n-Schritt-Übergangsmatrix $\mathbf{P}^n$ kann man auch schreiben:

$$p_{ij}^{(n)} = prob(X^{(n)} = s_j | X^{(0)} = s_i)$$

[11] Ein gerichteter Graph $G = (V, E)$ heißt stark zusammenhängend von einem Knoten v aus, falls es zu jedem Knoten w einen gerichteten Weg in G mit v als Startknoten und w als Endknoten gibt. G heißt stark zusammenhängend, falls G von jedem Knoten aus zusammenhängend ist. (`http://de.wikipedia.org/wiki/Zusammenhang_von_Graphen`)

Die **Periode** d_i eines Zustandes s_i ist definiert durch den größten gemeinsamen Teiler aller $n \geq 1$, für die $p_{ii}^{(n)} > 0$ ist. Der Zustand s_i heisst **aperiodisch**, wenn $d_i = 1$ ist. Eine Markov-Kette heisst aperiodisch, falls alle Zustände aperiodisch sind.

Gilt für eine Zustandsverteilung $\boldsymbol{\pi}$ die Beziehung

$$\boldsymbol{\pi} = \boldsymbol{\pi} \cdot \mathbf{P}, \tag{33}$$

so nennt man sie eine **stationäre Zustandsverteilung**.

Konvergieren die Zustandsverteilungen $\boldsymbol{\pi}^{(t)}$ einer Markov-Kette unabhängig von der Anfangsverteilung gegen eine **Grenzverteilung** $\boldsymbol{\pi}$, d.h. existiert der Grenzwert

$$\lim_{t \to \infty} \boldsymbol{\pi}^{(t)} = \boldsymbol{\pi}, \tag{34}$$

so ist diese Grenzverteilung nach Gleichung (31) stationär.

Eine irreduzible aperiodische Markov-Kette heisst **ergodisch**[12], wenn sie eine Grenzverteilung hat. In diesem Fall ist die Grenzverteilung positiv und die eindeutige stationäre Verteilung.

Eine homogene Markov-Kette heisst **reversibel**, wenn es eine Zustandsverteilung $\boldsymbol{\pi}$ gibt, die die *detaillierte Gleichgewichtsbedingung*

$$\pi_i p_{ij} = \pi_j p_{ji} \qquad \forall\ i \neq j \tag{35}$$

erfüllt.

In einer endlichen, homogen, reversiblen Markov-Kette erfüllt eine solche Zustandsverteilung wegen $\sum_j p_{ij} = 1$ die Gleichung

$$\pi_i = \sum_j p_{ji} \pi_j \qquad \forall\ i, \tag{36}$$

was gleichbedeutend ist mit (33).

Damit die Anforderung aus dem letzten Abschnitt, dass jede Stichprobe aus $\mathcal{G}(L, R)$ gleichwahrscheinlich ist, erfüllt werden kann, müsste der Swap-Algorithmus eine Markov-Kette $\mathcal{M}(\mathcal{G}(L, R), \mathbf{P})$ erzeugen, die

(a) irreduzibel und

[12] Wir haben hier lediglich die Äquivalenz in [Serfozo:2009], Korollar 60 ausgenutzt, um den Begriff *"ergodisch"* anschaulicher zu definieren. Die formale Definition erfordert noch den Begriff der *Rekurrenz*.

(b) ergodisch ist

(c) und in der alle Zustände gleichwahrscheinlich sind.

Nach Cobb und Chen [Cobb:2003] erfüllt der naive Swap-Algorithmus die Bedingungen (a) und (b). Die Bedingung (c) wird nur dann erfüllt, wenn alle Zustände im Übergangsgraphen $G_{\mathcal{M}} = (\mathcal{S}, \mathcal{T})$ den gleichen Knotengrad aufweisen. Dies hängt damit zusammen, dass die stationäre Zustandsverteilung in einer reversiblen Markov-Kette proportional zu den Knotengraden des Übergangsgraphen ist [Gionis:2007].

Die Bedingung (c) kann damit erreicht werden, in dem man einen Schritt im naiven Swap-Algorithmus auch dann als einen Swap-Schritt zählt, wenn die beiden Knotenpaare ein nicht-swapbares Muster bilden. Im Übergangsgraphen bedeutet ein solcher Schritt eine Schlinge (*self-loop*). Jede Schlinge erhöht in einem Graphen den Knotengrad um zwei.

Eingabe:	Graph $B = (T \cup P, E) \in \mathcal{G}(L, R)$,
	Anzahl der randomisierten Swap-Schritte k_{swaps}
Ausgabe:	Graph $B' \in \mathcal{G}(L, R)$
function `self_loop_swaps` (B,k_{swaps})	
1:	`Setze` $B' = B$
2:	**while** $k_{swaps} > 0$, **do**
3:	`Wähle nach dem Zufallsprinzip` $(t_i, p_x), (t_j, p_y) \in E$
4:	**if** $((t_j, p_x) \notin E$ **und** $(t_i, p_y) \notin E)$ **then**
5:	`Setze` $B' = B \setminus \{(t_i, p_x), (t_j, p_y)\} \cup \{(t_i, p_y), (t_j, p_x)\}$
6:	**end if**
7:	`Setze` $k_{swaps} = k_{swaps} - 1$
8:	**end while**
9:	`Gebe` B' `zurück.`

Algorithmus 2 : Self-Loop Swap-Algorithmus

Im Gegensatz zum naiven Swap-Algorithmus hat dieser Algorithmus auch noch den Vorteil, dass auf den Swapbarkeitstest (siehe Zeilen 3-5 im naiven Swap-Algorithmus) verzichtet werden kann. Auch der *Metropolis-Hastings-Algorithmus*, ein bekannter Algortihmus zur Erzeugung von ergodischen Markovketten, ist auf den Swapbarkeitstest angewiesen. Gionis et. al. [Gionis:2007] haben experimentell gezeigt, dass der Self-Loop Swap-Algorithmus immer effizienter ist als der Metropolis-Hastings-Algorithmus.

Konvergenz gegen die stationäre Verteilung

Leider gibt es keine praktische Methode zu bestimmen, wieviele lokalen Swap-Schritte tatsächlich notwendig sind, um die stationäre Verteilung der Markov-Kette zu erreichen. Gionis et. al. [Gionis:2007] haben für verschiedenartige Datensätze $B = (T \cup P, E)$ jeweils eine Markov-Kette von Stichproben B' generiert und die Anzahl der häufig auftretenden Itemsets $X \subset P$ mit $supp(X) > minsupp(X) =: \sigma$ im Zufallsgraphen B' ausgewertet. Diese Anzahl bezeichnen wir mit $|F(B')|$.

Diese wurden mit der Anzahl der häufig auftretenden Produkte[13] $X \subset P$ mit $supp(X) > \sigma$ im Datensatz B verglichen. Diese Anzahl bezeichnen wir mit $|F(B)|$.

Der Quotient $\dfrac{|F(B')|}{|F(B)|}$ wird hier zur Charakterisierung einer Stichprobe B' benutzt.

Konvergiert dieser stochastisch gegen einen Mittelwert, so ist die stationäre Zustandsverteilung der Markov-Kette erreicht. Bei den Versuchen von Gionis et. al. [Gionis:2007] stellt sich heraus, dass eine Durchmischungszeit (**"burn-in"-Phase**) von $\mathcal{O}(|E|)$ Swap-Schritten notwendig ist, bis sich der Quotient um einen Mittelwert einpendelt. Diese *experimentell* ermittelte Größenordnung kann als Richtwert für die Praxis genutzt werden. Ein mathematischer Beweis dafür ist nicht vorhanden.

In unserer reduzierten Filmdatenbank haben $|T|$, $|P|$ und $|E|$ folgende Größenordnungen: $|T| = 20.000$, $|P| = 17.770$ und $|E| \approx 2,3 \times 10^6$. Das ergibt eine Kantendichte von

$$\frac{|E|}{|T| \cdot |P|} \approx 0.55\%.$$

Die Datensätze in dem Artikel [Gionis:2007], die eine vergleichbare Kantendichte haben, benötigen eine Durchmischungszeit von $t_{\text{burn-in}} \leq 2 \cdot |E|$ Swap-Schritten.

Abbildung 7 zeigt ein ähnliches Verhalten für die stochastische Konvergenz gegen den stationären Zustand für unseren Datensatz. Wir haben den Konvergenztest für verschiedene Werte von $\sigma = minsupp(X)$ durchgeführt und die stationäre Zustandsverteilung nach $t_{\text{burn-in}} \leq 3 \cdot |E|$ Swap-Schritten erreichen können, auch für andere *minsupp*-Levels, die hier nicht dargestellt sind.

Wir schliessen daraus, dass die Wahl von $t_{\text{burn-in}} := 4 \cdot |E| = 8 \times 10^6$ Swap-Schritten für unseren Datensatz hinreichend sein dürfte. Danach entnehmen wir insgesamt 5000 Stichproben, die nacheinander in Abständen von jeweils $t_{sample} = \lfloor |T| \cdot \log |T| \rfloor \approx 200.000$ Swap-Schritten generiert werden.

[13] Zur Erinnerung: Die Menge $X \subset P$ wird durch die willkürliche Wahl des minimalen Supportlevels $\sigma = minsupp(X)$ festgelegt.

Im Kapitel 9 gehen wir auf die effiziente Implementierung der Berechnung der Werte von $coocc_{\text{FDSM}}(p_x, p_y)$ für jede einzelne Stichprobe B' ein.

`|F(B)|:`	# frequent itemsets in B
`|F(B')|:`	# frequent itemsets in der Stichprobe B' zum Zeitpunkt $0.5 \cdot j \cdot \|E\|$
`m:`	Mittelwert von `|F(B')|/|F(B)|` für `Swap-Schritte` $\geq 4\|E\|$
`s:`	Standardabweichung um `m`

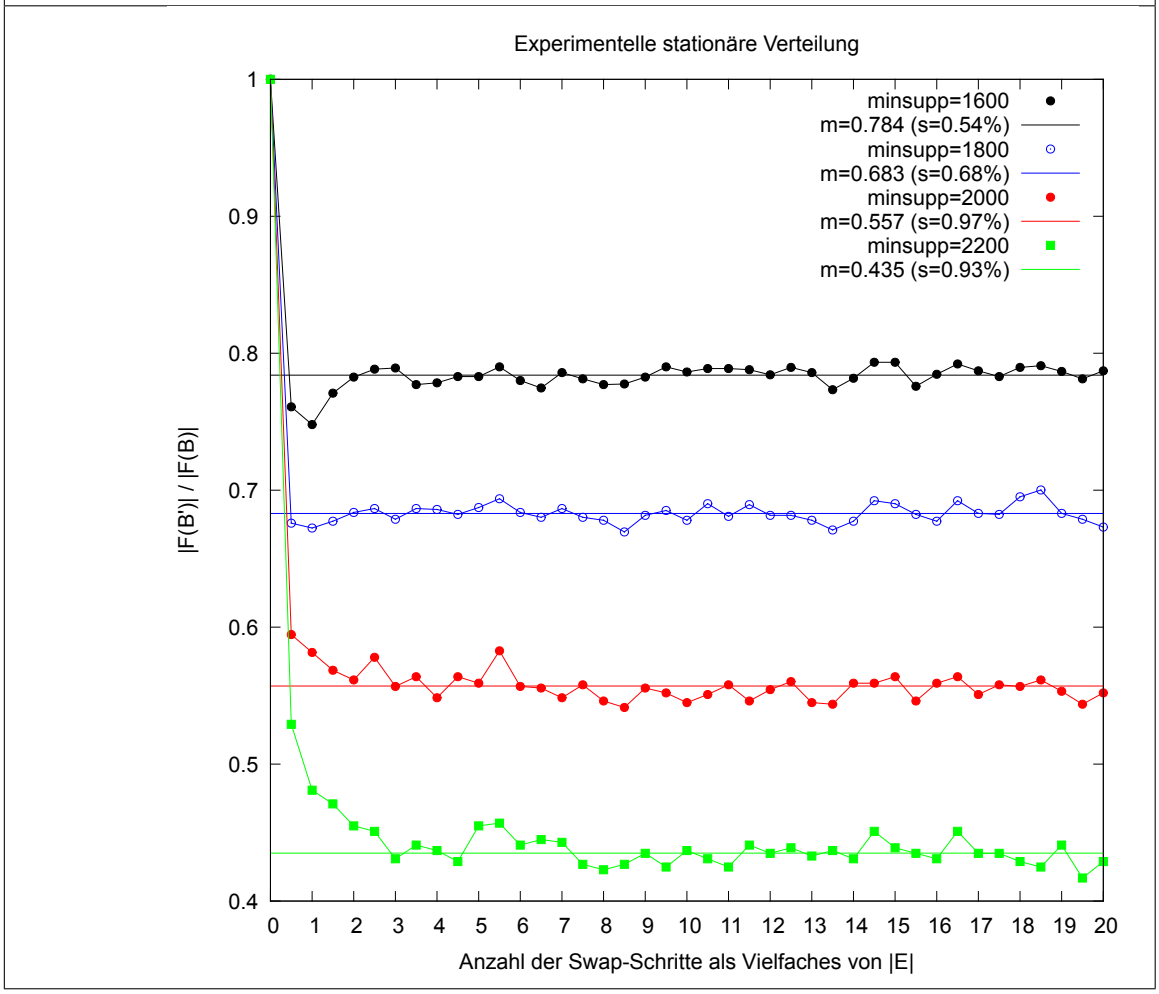


Abbildung 7: Stochastische Konvergenz gegen den stationären Zustand

4.6 Der Algorithmus

Nun können wir den Algorithmus zusammenfassen, der mit Hilfe des MCMC-Verfahrens für jedes Knotenpaar $p_x, p_y \in P,\ p_x \neq p_y$ einen Schätzer für den Erwartungswert

$$\mathbb{E}[coocc_{\text{FDSM}}(p_x, p_y)]$$

aus (23) liefert. Damit wird das Interessantheitsmaß

$$leverage_{\text{FDSM}}(p_x, p_y)$$

aus (20) berechnet.

Es entsteht eine Menge von $leverage_{\text{FDSM}}$-Werten, die wir in zwei Arten von Listen abspeichern,

1. in einer *globalen Liste*

$$GL := \big\{ \big(p_x, p_y, leverage_{\text{FDSM}}(p_x, p_y)\big) \;\big|\; p_x, p_y \in P \wedge leverage_{\text{FDSM}}(p_x, p_y) > 0 \big\}$$

2. und für jeden Knoten $p_x \in P$ in einer *lokalen Liste*

$$LL(p_x) := \big\{ \big(p_y, leverage_{\text{FDSM}}(p_x, p_y)\big) \;\big|\; p_y \in P \wedge leverage_{\text{FDSM}}(p_x, p_y) > 0 \big\}.$$

Nach dieser globalen Liste GL wird die Kantenmenge E_G für die einseitige Projektion $G = (P, E_G)$ festgelegt (vgl. Abschnitt 4.2). Die lokalen Listen speichern wir "auf Vorrat" mit, sie werden für die Diskussion des lokalen Leverage-Rankings im Abschnitt 4.7.2 und später im Abschnitt 5.1.1 für den Algorithmus 5 benötigt.

Eingabe:	$B = (T \cup P, E)$ und n_{samples}, die Anzahl der Stichproben
Zwischenergebnisse:	GL und $LL(p_x)$ für alle $p_x \in B$
Ausgabe:	$G = (P, E_G)$

```
function OMP(B, n_samples)
01:  setze GL = {}; // globale Liste initialisieren
02:  for each p_x ∈ B do
03:      setze LL(p_x) = {}; // lokale Listen initialisieren
04:  end for
05:  for each p_x, p_y ∈ B, p_x ≠ p_y do
06:      berechne coocc(p_x, p_y) nach (6);
07:  end for
08:  // Burn-in Phase:
09:  setze B_0 = B; setze t_burn-in := 4 · |E|;
10:  berechne B_1 = self_loop_swaps(B_0, t_burn-in);
11:  for each p_x, p_y ∈ B_1, p_x ≠ p_y do
12:      berechne coocc_1(p_x, p_y) nach (6);
13:      setze σ(p_x, p_y) = coocc_1(p_x, p_y); // Summationsvariable initialisiert
14:  end for
15:  // Sampling Phase:
16:  setze t_sample := ⌊|T| · log |T|⌋;
17:  setze i = 0;
18:  while i < n_samples do
19:      B_{i+1} = self_loop_swaps(B_i, t_sample);
20:      for each p_x, p_y ∈ B_{i+1}, p_x ≠ p_y do
21:          berechne coocc_{i+1}(p_x, p_y) nach (6);
22:          // Summationsvariable aufaddieren:
23:          berechne σ(p_x, p_y) = σ(p_x, p_y) + coocc_{i+1}(p_x, p_y);
24:      end for
25:      setze i = i + 1;
26:  end while
27:  for each p_x, p_y ∈ B_{i+1}, p_x ≠ p_y do
28:      // Erwartungswerte nach (23):
29:      berechne μ(p_x, p_y) = σ(p_x, p_y)/n_samples;
30:      // Interessantheitsmaß nach (20):
31:      berechne leverage(p_x, p_y) = coocc(p_x, p_y) − μ(p_x, p_y);
32:      if leverage(p_x, p_y) > 0 then
33:          füge leverage(p_x, p_y) der Liste GL hinzu;
34:          füge leverage(p_x, p_y) der Liste LL(p_x) hinzu;
35:      end if
36:  end for
37:  sortiere in allen erzeugten Listen die Einträge nach den
     leverage-Werten in nicht aufsteigender Reihenfolge
38:  baue G = (P, E_G), so dass für alle p_x, p_y ∈ P gilt:
     (p_x, p_y) ∈ E_G :⇔ leverage(p_x, p_y) > 0
```

Algorithmus 3 : OMP-Algorithmus mit Interessantheitsmaß *leverage*

4.7 Experimentelle Ergebnisse

Die Anwendung der einseitigen Projektion auf unsere reduzierte Filmdatenbank mit $|T| = 20.000$ Kunden, $|P| \approx 17.770$ Filmen und $|E| \approx 2,3$ Mio. Bewertungen[14] liefert ein globales Ranking aller Filmepaare nach den *leverage*-Werten und für jeden einzelnen Film jeweils eine Liste seiner besten "Freunde"(vgl. Abschnitt 4.2).

4.7.1 Globales Leverage Ranking

Tabelle 1 listet die ersten 20 Filmepaare mit den höchsten *leverage*-Werten der globalen Liste GL. Wir stellen fest[15]: Jedes der Filmepaare enthält tatsächlich zwei inhaltlich verwandte Filme. In dieser Liste kommen nur Filme vor, die einen hohen Bekanntheitsgrad haben.

Der Graph $G = (P, E_G)$ der einseitigen Projektion entsteht nach Abschnitt 4.2 durch die Verknüpfung aller Filmepaare (p_x, p_y) aus GL, für die $leverage(p_x, p_y) > 0$ ist. Für jede Kante $(p_x, p_y) \in E_G$ bildet $leverage(p_x, p_y)$ das Kantengewicht.

Man kann intuitiv annehmen, dass ein höherer *leverage*-Wert auch einen stärkeren inhaltlichen Zusammenhang zwischen zwei Filmen widerspiegelt.

Wenn wir statt der kompletten Liste GL für die einseitige Projektion nur die Top 10 heranziehen würden, bekämen wir für G die in Abbildung 8 dargestellten vier Zusammenhänge. Eine Erweiterung dieser Liste auf die Top 20 würde einen Graphen G liefern, in der alle Versionen aus der Trilogie *"Lord of the Rings"* einen zusammenhängenden Teilgraphen bilden. Eine Hinzunahme weiterer Filmepaare aus GL wird nicht nur diesen Teilgraphen für *"Lord of the Rings"* vollständig (d.h. jeder Knoten ist mit jedem anderen Knoten des Teilgraphen verbunden) machen, sondern es kommen weitere Zusammenhänge über weniger bekannte Filme hinzu. Auch im Hinblick auf die im Kapitel 6 vorgestellten Cluster-Algorithmen ist es sinnvoll und erwünscht, dass die Knoten innerhalb einer "Clique" besser miteinander verbunden sind als mit der "Außenwelt"(d.h. mit anderen Cliquen).

Mit der vollständigen Liste GL erhält man natürlich auch bei diesem Verfahren Zusammenhänge, die einen geringen Informationsgehalt haben, da ein niedriger *leverage*-Wert *in der Regel* nicht viel aussagt. Es gibt aber auch Fälle, in denen auch Filme mit niedrigem *leverage*-Wert eine Clique bilden können, wie das zweite Beispiel im folgenden Abschnitt zeigt.

[14] Die Bewertungsskala läuft von 1 (sehr schlecht) bis 5 (sehr gut). Es wurden nur Bewertungen > 3 in Betracht gezogen.

[15] nach unseren bescheidenen Erfahrungen als Kinobesucher

Ranking	*leverage*	*deg*	Jahr	**Filmtitel**
1	2037	5542	2002	Lord of the Rings: The Two Towers
		5401	2001	Lord of the Rings: The Fellowship of the Ring
2	1833	4974	2003	Lord of the Rings: The Return of the King
		5542	2002	Lord of the Rings: The Two Towers
3	1746	4974	2003	Lord of the Rings: The Return of the King
		5401	2001	Lord of the Rings: The Fellowship of the Ring
4	1693	2881	2001	The Lord of the Rings: The Fellowship of the Ring: Extended Edition
		2916	2002	Lord of the Rings: The Two Towers: Extended Edition
5	1627	3204	1983	Star Wars: Episode VI: Return of the Jedi
		3422	1980	Star Wars: Episode V: The Empire Strikes Back
6	1592	3130	1977	Star Wars: Episode IV: A New Hope
		3422	1980	Star Wars: Episode V: The Empire Strikes Back
7	1571	2872	2003	Lord of the Rings: The Return of the King: Extended Edition
		2916	2002	Lord of the Rings: The Two Towers: Extended Edition
8	1543	2872	2003	Lord of the Rings: The Return of the King: Extended Edition
		2881	2001	The Lord of the Rings: The Fellowship of the Ring: Extended Edition
9	1455	3130	1977	Star Wars: Episode IV: A New Hope
		3204	1983	Star Wars: Episode VI: Return of the Jedi
10	1449	5048	1989	Indiana Jones and the Last Crusade
		4471	1981	Indiana Jones: Raiders of the Lost Ark
11	1376	3407	2001	Harry Potter and the Sorcerer's Stone
		3550	2002	Harry Potter and the Chamber of Secrets
12	1361	3783	2003	Kill Bill: Vol. 1
		3373	2004	Kill Bill: Vol. 2
13	1265	4974	2003	Lord of the Rings: The Return of the King
		2916	2002	Lord of the Rings: The Two Towers: Extended Edition
14	1228	3988	1972	The Godfather
		2580	1974	The Godfather, Part II
15	1226	4974	2003	Lord of the Rings: The Return of the King
		2881	2001	The Lord of the Rings: The Fellowship of the Ring: Extended Edition
16	1217	4639	2001	Monsters, Inc. (Animation)
		5104	2003	Finding Nemo (Animation)
17	1212	2872	2003	Lord of the Rings: The Return of the King: Extended Edition
		4974	2003	Lord of the Rings: The Return of the King
18	1160	5542	2002	Lord of the Rings: The Two Towers
		2916	2002	Lord of the Rings: The Two Towers: Extended Edition
19	1146	5048	1989	Indiana Jones and the Last Crusade
		3327	1984	Indiana Jones and the Temple of Doom
20	1146	3327	1984	Indiana Jones and the Temple of Doom
		4471	1981	Indiana Jones: Raiders of the Lost Ark

Tabelle 1: Top 20 der Filmpaare in der globalen Liste nach *leverage*-Werten sortiert

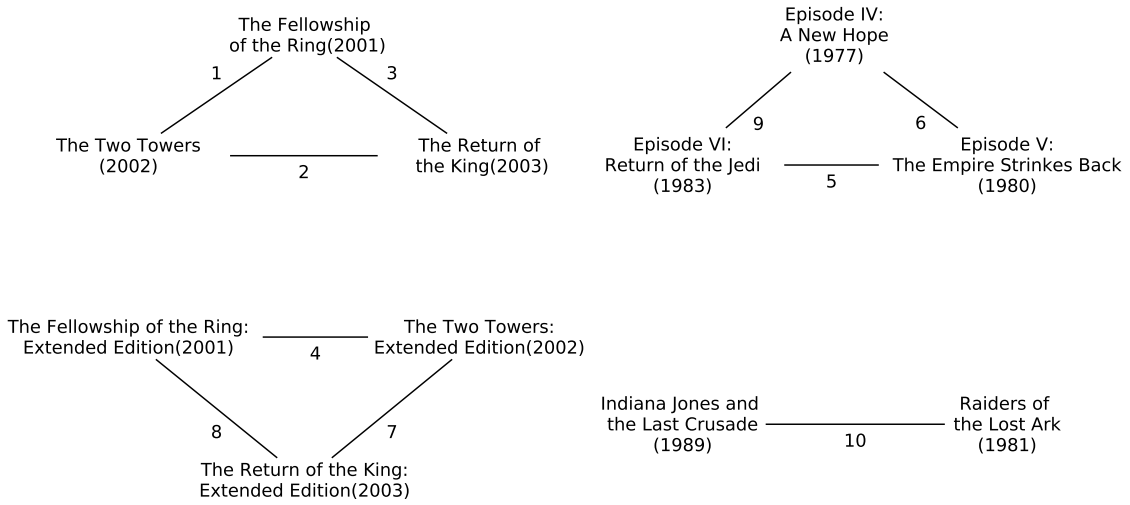


Abbildung 8: OMP enstanden aus den Top-10 der globalen Liste GL

4.7.2 Lokales Leverage Ranking

Die *leverage*-Werte in der lokalen Liste aus Tabelle 3 sind sehr klein im Vergleich zu den Werten der lokalen Liste aus Tabelle 2. Nichtsdestotrotz enthält die Tabelle 3 sehr zutreffende Informationen: Genauso wie der Film "Chinese Odyssey 1 (1995)" stammen neun seiner zehn am besten "Freunde" aus Hong Kong und haben alle denselben kantonesischsprachigen Schauspieler (Stephen Chow) als Hauptdarsteller. Der zehnte Film stammt aus Japan.

In der globalen Liste würden diese im englischsprachigen Raum eher unbekannten Filme sehr weit hinten aufgelistet werden. Das hängt damit zusammen, dass niedrige Knotengrade von zwei befreundeten Knoten auch einen niedrigen *leverage*-Wert nach sich ziehen:

Seien $p_x, p_y \in P$ zwei befreundete Knoten. Dann ist $leverage_{\text{FDSM}}(p_x, p_y) > 0$. Der Erwartungswert $\mathbb{E}[coocc_{\text{FDSM}}(p_x, p_y)]$ ist grundsätzlich ≥ 0.

Wegen (20) gilt

$$\begin{aligned} 0 < leverage_{\text{FDSM}}(p_x, p_y) &= coocc(p_x, p_y) - \mathbb{E}[coocc_{\text{FDSM}}(p_x, p_y)] \\ &\leq coocc(p_x, p_y) \\ &\leq \min\left\{deg(p_x), deg(p_y)\right\} \end{aligned} \tag{37}$$

Ranking	*leverage*	**deg**	Jahr	**Filmtitel**
1	2037	5401	2001	Lord of the Rings: The Fellowship of the Ring
2	1833	4974	2003	Lord of the Rings: The Return of the King
3	1160	2916	2002	Lord of the Rings: The Two Towers: Extended Edition
4	1139	2881	2001	The Lord of the Rings: The Fellowship of the Ring: Extended Edition
5	1101	2872	2003	Lord of the Rings: The Return of the King: Extended Edition
6	933	4834	1999	The Matrix
7	844	6420	2003	Pirates of the Caribbean: The Curse of the Black Pearl
8	804	3422	1980	Star Wars: Episode V: The Empire Strikes Back
9	777	4471	1981	Raiders of the Lost Ark
10	710	3204	1983	Star Wars: Episode VI: Return of the Jedi

Tabelle 2: Die zehn besten "Freunde" von *"Lord of the Rings: The two Towers(2002)"*

Ranking	*leverage*	**deg**	Jahr	**Filmtitel**
1	4	7	1995	Chinese Odyssey 2: Cinderella
2	2	4	1992	Moon Warriors
3	2	3	1990	All for the Winner
4	2	6	1992	Royal Tramp
5	2	3	1991	God of Gamblers III: Back to Shanghai
6	2	5	1991	God of Gamblers II
7	1	2	1997	Lawyer Lawyer
8	1	3	2002	Colorful!
9	1	5	1992	The East Is Red: Swordsman 3
10	1	10	1992	King of Beggars

Tabelle 3: Die zehn besten "Freunde" von *"Chinese Odyssey 1(1995)"*

4.8 Skalierter *leverage*-Wert

Zweig [Zweig:2010] beschränkt sich auf globale und lokale Leverage Rankings, aber hier gibt es eine Möglichkeit die Signifikanzberechnung zu verbessern. Nach den oben gemachten Überlegungen sollte man *leverage*-Werte nicht global vergleichen. Damit ist es nicht sinnvoll, eine untere Schranke für *leverage*-Werte zu wählen, um die globale Liste GL für die einseitige Projektion nach unten abzuschneiden.

Da nach Ungleichung (37) der *leverage*-Wert nach oben durch den minimalen Knotengrad beschränkt ist, bietet sich zunächst einmal an, als neues Interessantheitsmaß für die globale

Liste den skalierten Wert

$$s_{\min}(p_x, p_y) \quad := \quad \frac{leverage_{\text{FDSM}}(p_x, p_y)}{\min\left\{deg(p_x), deg(p_y)\right\}}$$

zu wählen. Wir werden jedoch in den nachfolgenden Zahlenbeispielen sehen, daß die Wahl

$$s_{\max}(p_x, p_y) \quad := \quad \frac{leverage_{\text{FDSM}}(p_x, p_y)}{\max\left\{deg(p_x), deg(p_y)\right\}} \tag{38}$$

zu bevorzugen ist, wenn wir nach dem Prinzip vorgehen:

"Lieber wenige richtige als viele falsche Informationen!"

Damit würden alle Filmepaare aus den Tabellen 2 und 3 ein Interessantheitsmaß mit vergleichbaren Werten zwischen 0 und 1 annehmen:

Für die sehr bekannten Filme $p_x =$ *"Lord of the Rings: The Two Towers"* und
$p_y =$ *"Lord of the Rings: The Fellowship of the Ring"* erhalten wir mit
$deg(p_x) = 5542$, $deg(p_y) = 5401$ und $leverage_{\text{FDSM}}(p_x, p_y) = 2037$ den Wert

$$s_{\max} = 2037/\max\{5542, 5401\} = 0.368.$$

Für die weniger bekannten Filme $p_x =$ *"Chinese Odyssey 1"* und
$p_y =$ *"Chinese Odyssey 2"* erhalten wir mit
$deg(p_x) = 5$, $deg(p_y) = 7$ und $leverage_{\text{FDSM}}(p_x, p_y) = 4$ den Wert

$$s_{\max} = 4/\max\{5, 7\} = 0.571.$$

Warum $s_{\max}$ besser als $s_{\min}$ ist

Angenommen die 5 Kunden, die *"Chinese Odyssey 1"* angeschaut haben, wären auch unter den 5542 Kunden, die *"Lord of the Rings: The Two Towers"* angeschaut haben. Dann hätten wir

$deg(p_x) = 5$, $deg(p_y) = 5542$ und $coocc(p_x, p_y) = 5 = \min\left\{deg(p_x), deg(p_y)\right\}$.

Der Erwartungswert $\mathbb{E}[coocc_{\text{FDSM}}(p_x, p_y)]$ kann maximal nur 5 sein, weil auch $coocc_{\text{FDSM}}(p_x, p_y)$ durch $\min\left\{deg(p_x), deg(p_y)\right\}$ nach oben beschränkt ist. Der wahrscheinlichere Fall ist aber, dass der Erwartungswert einen Wert annimmt, der ein wenig größer als 0 ist. Dann wäre $0 < leverage_{\text{FDSM}}(p_x, p_y) \leq 5$ nahe bei 5 und $s_{min} < 1$ nahe beim maximalen Wert 1, was einen starken Zusammenhang zwischen den beiden Filmen suggerieren würde. Mit der

Wahl $s_{\max}$ hätten wir von vornherein mit der Division durch den maximalen Knotengrad 5542 einen sehr kleinen Wert.

Ein weiterer Beweggrund für die Wahl von $s_{\max}$ als Interessanheitsmaß ist der in den beiden Tabellen 4 und 5 dargestellte Vergleich von Simulationsergebnissen. Im Gegensatz zu der mit s_{min} erzeugten Liste entstammen die mit $s_{\max}$ gefunden zehn besten "Freunde" von *"The X-Files: Season 1"* fast alle der gleichen Serie.

Ranking	$s_{\min}$	*deg*	Jahr	**Filmtitel**
1	0.93	2	1980	Shakespeare Tragedies: Hamlet
2	0.92	2	2005	Lamb of God: Killadelphia
3	0.91	2	1977	The People That Time Forgot
4	0.91	2	2001	John Waters Collection: Extras
5	0.91	2	2003	The Hulk: Bonus Material
6	0.90	2	1984	Flashpoint
7	0.90	2	1945	Classic Cartoon Favorites: Extreme Adventure Fun
8	0.89	2	2001	O: Bonus Material
9	0.74	287	2000	The X-Files: Season 3
10	0.73	199	2000	The X-Files: Season 8

Tabelle 4: Die zehn besten "Freunde" von *"The X-Files: Season 1"*, nach s_{min} ermittelt

Ranking	$s_{\max}$	*deg*	Jahr	**Filmtitel**
1	0.61	364	2000	The X-Files: Season 2
2	0.52	287	2000	The X-Files: Season 3
3	0.47	269	1997	The X-Files: Season 5
4	0.46	265	1998	The X-Files: Season 6
5	0.45	296	1996	The X-Files: Season 4
6	0.38	218	1999	The X-Files: Season 7
7	0.36	199	2000	The X-Files: Season 8
8	0.29	432	1998	The X-Files: Fight the Future
9	0.29	163	2001	The X-Files: Season 9
10	0.16	358	2001	Buffy the Vampire Slayer: Season 6

Tabelle 5: Die zehn besten "Freunde" von *"The X-Files: Season 1"*, nach $s_{\max}$ ermittelt

Auch in der mit $s_{\max}$ erzeugten globalen Liste erscheinen unter den Top 20 eher zusammengehörige Filmepaare als in der mit s_{min} erzeugten globalen Liste (siehe Tabellen 6 und 7 auf den nachfolgenden Seiten).

Abschließend fassen wir im Algorithmus 4 die einseitige Projektion basierend auf $s_{\max}$ zusammen.

range	$s_{\max}$	deg	year	movie title
1	1.0	2	1999	Power Play
		2	1998	Naked Lies
2	1.0	2	2001	Amazons and Gladiators
		2	2000	Battle Queen 2020
3	1.0	2	1999	Aladdin and the Adventure of All Time
		2	2000	American Tragedy
4	1.0	2	1994	Saved by the Bell: The New Class: Season 2
		2	1993	Saved by the Bell: The New Class: Season 1
5	1.0	2	1980	Ram Balram
		2	1999	Sooryavansham
6	1.0	2	1945	Val Lewton: Isle of the Dead / Bedlam
		2	1956	Rodan
7	1.0	2	1988	Kimagure Orange Road Movie: I Want to Return to That Day
		2	1989	City Hunter: .357 Magnum
8	1.0	2	2002	The Day of the Wacko
		2	1995	Nothing Funny
9	1.0	2	1998	Nazca: Blades of Fate
		2	1998	Nazca: Betrayal of Humanity
10	1.0	2	1998	Nazca: Blades of Fate
		2	1998	Nazca: Eternal Power
11	1.0	2	1991	To Play or to Die
		2	2004	Daydream Obsession 2: Infidelities
12	1.0	2	1919	Chaplin: The Collection: Vol. 3
		2	1965	Buster Keaton Rides Again/The Railrodder
13	1.0	2	1999	Barney: Let's Play School
		2	2000	Barney's Super Singing Circus
14	1.0	2	1981	Road Games
		2	1988	Maniac Cop
15	1.0	2	2003	K-Hole
		2	2003	The Singing Forest
16	1.0	2	2003	K-Hole
		2	2003	Revenge in Olympia
17	1.0	2	1987	Hello Kitty Saves the Day
		2	2004	Hello Kitty & Friends
18	1.0	2	1991	Legend of the Dragon Kings: Black Dragon
		2	1991	Legend of the Dragon Kings: White Dragon
19	1.0	2	1947	The Perils of Pauline
		2	1999	Brian Wilson: Imagination
20	1.0	2	2001	Beyblade: G Revolution
		2	2000	Pokemon: The Advanced Master's Guide

Tabelle 6: Top 20 der Filmpaare in der globalen Liste nach $s_{\max}$-Werten sortiert

range	s_{min}	deg	year	movie title
1	1.0	18	1987	Extreme Prejudice
		2	1977	The People That Time Forgot
2	1.0	3	1997	Music for Montserrat
		2	2000	Paul McCartney and Friends: The PETA Concert for Party Animals
3	1.0	2	1999	Power Play
		2	1998	Naked Lies
4	1.0	2	2001	Transfixed
		4	1993	Pretty Boy
5	1.0	6	1932	White Zombie
		2	1987	Gothic
6	1.0	2	2004	Daydream Obsession 2: Infidelities
		4	1998	A Change of Heart
7	1.0	2	1991	Carnal Crimes
		7	1996	Price of Desire
8	1.0	2	2000	Women of the Night
		3	2003	Emmanuelle in Rio
9	1.0	2	2001	Amazons and Gladiators
		2	2000	Battle Queen 2020
10	1.0	6	2002	ECW: Anarchy Rulz '99
		2	2003	Smack: Vol. 1
11	1.0	2	1999	Aladdin and the Adventure of All Time
		2	2000	American Tragedy
12	1.0	2	1977	Alaap
		4	1972	Bawarchi
13	1.0	2	2001	Yaadein
		4	2002	Na Tum Jaano Na Hum
14	1.0	5	1968	The Scalphunters
		2	1933	W.C. Fields: Six Short Films
15	1.0	2	1994	Saved by the Bell: The New Class: Season 2
		2	1993	Saved by the Bell: The New Class: Season 1
16	1.0	2	1980	Ram Balram
		2	1999	Sooryavansham
17	1.0	2	1945	Val Lewton: Isle of the Dead / Bedlam
		8	1938	Flash Gordon's Trip to Mars
18	1.0	2	1945	Val Lewton: Isle of the Dead / Bedlam
		2	1956	Rodan
19	1.0	2	1988	Kimagure Orange Road Movie: I Want to Return to That Day
		2	1989	City Hunter: .357 Magnum
20	1.0	2	2002	Global Warning: Australia
		5	2003	WWE: 'Cause Stone Cold Said So

Tabelle 7: Top 20 der Filmpaare in der globalen Liste nach s_{min}-Werten sortiert

Eingabe:	$B = (T \cup P, E)$ und n_{samples}, die Anzahl der Stichproben
Zwischenergebnisse:	GL und $LL(p_x)$ für alle $p_x \in B$
Ausgabe:	$G = (P, E_G)$

```
function OMP(B, n_samples)
01:  setze GL = {}; // globale Liste initialisieren
02:  for each p_x ∈ B do
03:      setze LL(p_x) = {}; // lokale Listen initialisieren
04:  end for
05:  for each p_x, p_y ∈ B, p_x ≠ p_y do
06:      berechne coocc(p_x, p_y) nach (6);
07:  end for
08:  // Burn-in Phase:
09:  setze B_0 = B; setze t_burn-in := 4 · |E|;
10:  berechne B_1 = self_loop_swaps(B_0, t_burn-in);
11:  for each p_x, p_y ∈ B_1, p_x ≠ p_y do
12:      berechne coocc_1(p_x, p_y) nach (6);
13:      setze σ(p_x, p_y) = coocc_1(p_x, p_y); // Summationsvariable initialisiert
14:  end for
15:  // Sampling Phase:
16:  setze t_sample := ⌊|T| · log |T|⌋;
17:  setze i = 0;
18:  while i < n_samples do
19:      B_{i+1} = self_loop_swaps(B_i, t_sample);
20:      for each p_x, p_y ∈ B_{i+1}, p_x ≠ p_y do
21:          berechne coocc_{i+1}(p_x, p_y) nach (6);
22:          // Summationsvariable aufaddieren:
23:          berechne σ(p_x, p_y) = σ(p_x, p_y) + coocc_{i+1}(p_x, p_y);
24:      end for
25:      setze i = i + 1;
26:  end while
27:  for each p_x, p_y ∈ B_{i+1}, p_x ≠ p_y do
28:      // Erwartungswerte nach Gleichung (23):
29:      berechne μ(p_x, p_y) = σ(p_x, p_y)/n_samples;
30:      // Interessantheitsmaß nach Gleichung (20):
31:      berechne leverage(p_x, p_y) = coocc(p_x, p_y) − μ(p_x, p_y);
32:      if leverage(p_x, p_y) > 0 then
33:          berechne s_max(p_x, p_y) nach (38);
34:          füge s_max(p_x, p_y) der Liste GL hinzu;
35:          füge s_max(p_x, p_y) der Liste LL(p_x) hinzu;
36:      end if
37:  end for
38:  sortiere in allen erzeugten Listen die Einträge nach den s_max-Werten in
     nicht aufsteigender Reihenfolge und speichere sie ab.
39:  baue G = (P, E_G), so dass für alle p_x, p_y ∈ P gilt:
     (p_x, p_y) ∈ E_G :⇔ s_max(p_x, p_y) > 0
```

Algorithmus 4 : *Modifizierter* OMP-Algorithmus mit Interessantheitsmaß $s_{\max}$

5 Clustering mit Link Communities

Ziel unseres Empfehlungssystems ist es, zu jedem Film eine Liste von Filmen aufzufinden, die einen möglichst ähnlichen Inhalt haben. Wir verlassen uns dabei ausschließlich auf die positiven Bewertungen der Kunden. Die bisher berechneten Interessantheitsmaße geben zwar Assoziationsregeln der Form $p_x \Longrightarrow p_y$ wieder. Doch Assoziationsregeln sind weder reflexiv (d.h. aus $p_x \Longrightarrow p_y$ folgt nicht unbedingt $p_y \Longrightarrow p_x$) noch transitiv (d.h. aus $p_x \Longrightarrow p_y$ und $p_y \Longrightarrow p_z$ folgt nicht unbedingt $p_x \Longrightarrow p_z$). Daher können die lokalen Listen nur ein Anhaltspunkt für eine Empfehlungsliste sein. Für eine systematische Klassifizierung benötigen wir die Methoden der Clusteranalyse.

Im Gegensatz zu einem Zufallsgraphen (nach dem Modell 1 aus Abschnit 2.2) enthält ein reales Netzwerk in der Regel ausgeprägte Clusterstrukturen. Sei $\hat{G} = (P, \hat{E}_G)$ der aus $B = (T \cup P, E)$ durch die einseitige Projektion entstandene Graph. Dieser Produktgraph $\hat{G}$ ist in der Regel größtenteils zusammenhängend. Wir wollen nun die Filme aus P nach einem bestimmten *Ähnlichkeitsmaß* in Gruppen (**Clustern**)[16] zusammenfassen. Dabei dürfen nur die Informationen benutzt werden, die in der Kantenmenge $\hat{E}_G$ stecken (wiederum ein rein netzwerkanalytischer Ansatz).

Als Ergebnis dieser Clusterung erhalten wir eine Hierarchie von Clustern. Wir schliessen daraus, dass jedes Cluster möglichst nur die Filme enthält, die ähnlichen Inhalt haben.

Grundlage dieses Kapitels ist der Artikel von Ahn et. al. [Ahn:2010] von Ahn, Bagrow und Lehmann, in dem eine neue Methode zur Kantenpartitionierung mit verschiedenen bekannten Verfahren zur Knotenpartitionierung verglichen wird. Wir werden den *Link-Community-Algorithmus für ungewichtete (und ungerichtete) Graphen* vorstellen und diesen auf einen (weiter) reduzierten Graphen $G = (P, E_G)$ anwenden, mit dessen Konstruktion wir jetzt beginnen.

5.1 Umgewichtung der einseitigen Projektion

Der gewichtete OMP-Graph $\hat{G} = (P, \hat{E}_G)$ muss also in einen ungewichteten Graphen $G = (P, E_G)$ umgewandelt werden. Wir stellen hier zwei heuristische Ansätze vor, die Kantenmenge $\hat{E}_G$ zu reduzieren, und zwar bei der *leverage*-basierten OMP nach dem *Prinzip der Gegenseitigkeit (Reziprozität)* und bei der $s_{\max}$-basierten OMP mit Hilfe des *mittleren Clusterkoeffizienten*.

[16] Die Begriffe *Cluster, Partition, Community* werden alle zu demselben Zweck benutzt: Kategorisierung der Knoten eines Graphen in Gruppen.

5.1.1 Reziprozität

Der *leverage*-basierte OMP-Algorithmus 3 aus Abschnitt 4.6 liefert nicht nur die globale Liste GL, aus der die Kantenmenge $\hat{E}_G$ für $\hat{G}$ bestimmt wird, sondern zu jedem Knoten $p_x \in P$ auch die lokale Liste $LL(p_x)$ seiner Freunde. Doch diese Art von "Freundschaft" muss nicht unbedingt auf Gegenseitigkeit beruhen, d.h. wenn p_y in der Top 10 von $LL(p_x)$ aufgelistet ist, heisst es noch nicht, dass auch p_x zu den besten Freunden von p_y zählt.

Für die Konstruktion des ungewichteten *leverage*-basierten OMP-Graphen $G = (P, E_G)$ fordern wir, dass ausschließlich solche Knoten miteinander verknüpft sind, die sich gegenseitig zu den besten Freunden zählen. Danach können die Kantengewichte in E_G weggelassen werden.

Zunächst legen wir fest, welche Freunde von p_x zu seinen besten Freunden zählen soll. Sei $l_{\max}$ der größte *leverage*-Wert in der lokalen Liste $LL(p_x)$. Dann ist

$$LL_{\text{best}}(p_x) := \left\{ p_y \in LL(p_x) \;\middle|\; leverage(p_y, p_x) > \delta_R \cdot l_{\max} \right\}$$

die Menge der besten Freunde von p_x, wobei der Parameter $\delta_R \in (0, 1)$ sein sollte. Wir wählen für unsere Tests $\delta_R = \frac{1}{2}$ bzw. $\delta_R = \frac{1}{3}$.

Für die Reziprozität fordern wir, dass

$$(p_x, p_y) \in \hat{E}_G \quad :\Longleftrightarrow \quad p_y \in LL_{\text{best}}(p_x) \;\wedge\; p_x \in LL_{\text{best}}(p_y).$$

Alle anderen Kanten aus $\hat{E}_G$ werden damit entfernt.

---

1) Sei $\hat{G} = (P, \hat{E}_G)$ der aus $B = (T \cup P, E)$ durch die *leverage*-basierte OMP (nach Algorihmus 3) entstandene Graph.

2) Iteriere über alle Knoten p_x von $\hat{G}$ und überprüfe für alle $p_y \in LL_{\texttt{best}}(p_x)$, ob auch $p_x \in LL_{\texttt{best}}(p_y)$ gilt. Falls nein, entferne die Kante zwischen p_x und p_y. Am Ende bleibt ein Graph $G = (P, E_G)$ mit reduzierter Kantenmenge E_G.

3) Alle Kantengewichte von E_G auf 1 setzen.

Algorithmus 5 : Umgewichtung nach dem Prinzip der Reziprozität

5.1.2 Clusterkoeffizienten

Um einen Maßstab dafür zu haben, inwiefern die Knoten eines Netzwerkes dazu tendieren, eine Gruppierung zu bilden, haben Watts und Strogatz [Watts:1998] für die Definition eines neuen Zufallsgraphenmodells zur mathematischen Untersuchung des 'small-world'-Experiments von Milgram [Milgram:1967] den *lokalen* und *mittleren Clusterkoeffizienten* eingeführt.

Sei $G = (V, E)$ ein ungerichteter und ungewichteter Graph ohne isolierte Knoten. Sei $i \in V$ ein Knoten und sei $n(i)$ die Menge seiner Nachbarn (siehe Abschnitt 2.1). Dann gibt es maximal $|n(i)| \cdot (|n(i)| - 1)$ mögliche Kanten zwischen den Nachbarn untereinander. Der **lokale Cluster-Koeffizient** $C(i)$ ist der Anteil tatsächlich existierender Kanten zwischen den exklusiven Nachbarknoten von i:

$$C(i) \quad := \quad \frac{\left|\left\{e_{jk} \in E\right\}_{j,k \in n(i)}\right|}{\left|n(i)\right| \cdot \left(\left|n(i)\right| - 1\right) \Big/ 2} \qquad \text{mit} \quad n(i) := \left\{x \in V \middle| d(i,x) = 1\right\}$$

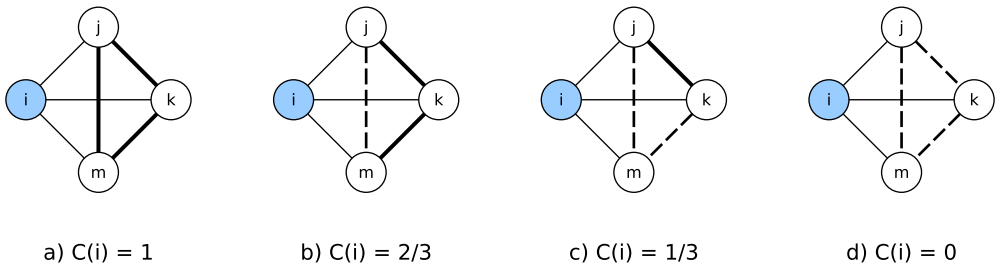


Abbildung 9: Beispiele aus `http://en.wikipedia.org/wiki/Clustering_coefficient`

Der **mittlere Cluster-Koeffizient** ist der Durchschnitt über alle $i \in V$:

$$\overline{C} = \frac{1}{|V|} \sum_{i \in V} C(i) \tag{39}$$

Der $s_{\max}$-basierte OMP-Algorithmus 4 aus Abschnitt 4.8 liefert uns eine globale Liste GL mit $s_{\max}$-Werten zwischen 0 und 1, die die Kantengewichte darstellen. Für die weitere Reduktion auf einen ungewichteten Graphen gehen wir folgendermaßen vor:

1) Sei $\hat{G} = (P, \hat{E}_G)$ der aus $B = (T \cup P, E)$ durch die $s_{\max}$-basierte OMP (nach Algorihmus 4) entstandene Graph. Die in der globalen Liste GL sortierten $s_{\max}$-Werte seien $0 < s_1 < s_2 < ... < s_\lambda < 1$.

2) Iteration über $s \in \{s_1, s_2, ..., s_\lambda\}$: Sukzessive Reduktion der Kantenzahl von G durch Weglassen von Kanten, für welche $s_{\max} < s$ gilt. Das liefert uns: $\hat{G} \rightsquigarrow \hat{G}_1 \rightsquigarrow \hat{G}_2 \rightsquigarrow ... \rightsquigarrow \hat{G}_\lambda$

3) Iteration über $j = 1, 2, ..., \lambda$: Kantengewichte von $\hat{G}_j$ auf 1 setzen liefert G_j. Berechne den mittleren Cluster-Koeffizienten $\overline{C}$ für den jeweiligen Graphen G_j.

4) Threshold für *minimale Freundschaftsbeziehung* zwischen allen Knoten untereinander ist definiert durch ein $s_j \in (0, 1)$, für welches der mittlere Cluster-Koeffizient $\overline{C}$ maximal wird. Für G wählen wir den zugehörigen Graphen G_j aus:

Algorithmus 6 : Umgewichtung mit maximalem mittleren Cluster-Koeffizienten

5.2 Link Communities

Klassische Methoden zur Gruppierung von Netzwerkknoten basieren auf einer Partitionierung der Knotenmenge, so dass am Ende jeder Knoten genau in einer Partition enthalten ist. Eine intuitive und weit verbreitete Definition einer Partition ist gegeben dadurch, dass jeder Knoten innerhalb einer Partition mehr Verbindungskanten mit Knoten innerhalb derselben Partition haben soll als mit Knoten außerhalb dieser Partition. Für eine weiterführende quantitative Definition sei auf Kapitel 3 von [Fortunato:2010] verwiesen[17].

Der Begriff **Link Community** heißt wörtlich übersetzt soviel wie Verbindungsgemeinschaft. In einem sozialen Netzwerk sind zwei Individuen miteinander verbunden, wenn sie beispielsweise zu derselben Familie gehören, miteinander befreundet sind, zusammenarbeiten oder die gleiche Sportart betreiben. Eine Menge von Individuen, die miteinander verbunden sind, bilden eine Gemeinschaft. Nun kann jedes Individuum nicht nur Mitglied in einer einzigen Gemeinschaft sein, sondern gleichzeitig in mehreren Gemeinschaften. In einem solchen Fall überlappen verschiedene Gemeinschaften (**overlap**) und die obige intuitive Definition einer Partition kann nicht für eine Gemeinschaft benutzt werden, denn ein Individuum innerhalb einer bestimmten Gemeinschaft kann mehr Verbindungen mit der Außenwelt haben als innerhalb dieser Gemeinschaft. Ahn et. al. [Ahn:2010] definieren eine Community als eine Menge von Kanten aus E_G, die *eng miteinander verknüpft* sind.

[17] Der Artikel [Fortunato:2010] umfasst eine Reihe von Clustering-Algorithmen, auf die wir nicht eingehen.

Wie das genau gemeint ist, wollen wir im Folgenden erläutern.

5.2.1 Ähnlichkeit zweier Kanten

Sei $G = (V, E)$ ein ungerichteter und ungewichteter Graph. Seien $i, j, k \in V$ drei Knoten in G, die über zwei Kanten $e_{ik} = \{i, k\}$ und $e_{jk} = \{i, k\}$ miteinander verbunden sind. Man kann auch sagen, dass die beiden Kanten $e_{ik}, e_{jk} \in E$ über einen Knoten k miteinander verbunden sind. Dann wird aufbauend auf (2) durch

$$Sim(e_{ik}, e_{jk}) := \frac{|n_+(i) \cap n_+(j)|}{|n_+(i) \cup n_+(j)|} \tag{40}$$

ein Maß für die Ähnlichkeit (**Similarity**)[18] der beiden Kanten definiert. Für zwei Kanten $e_{ik}, e_{jl} \in E$, die nicht miteinander verknüpft sind, setzt man

$$Sim(e_{ik}, e_{jl}) := 0. \tag{41}$$

Es gilt also $0 \leq Sim(e, f) \leq 1$ für je zwei Kanten $e, f \in E$. Zwei Kanten sind desto enger miteinander verknüpft, je höher ihre Ähnlichkeit ist.

Interpretiert man eine einzelne Kante e_{ik} als zwei zusammengefallene Kanten e_{ik} und e_{jk}, so dass $i = j$ wird, so lässt sich die Ähnlichkeit einer Kante mit sich selbst definieren:

$$Sim(e_{ik}) := Sim(e_{ik}, e_{jk}) = Sim(e_{ik}, e_{ik}) = \frac{|n_+(i) \cap n_+(i)|}{|n_+(i) \cup n_+(i)|} = \frac{|n_+(i)|}{|n_+(i)|} = 1. \tag{42}$$

Bemerkung:

Man hätte ausgehend von der vermeintlich intuitiveren Definition (1) für das Ähnlichkeitsmaß auch den Ausdruck

$$\widetilde{Sim}(e_{ik}, e_{jk}) := \frac{|n(i) \cap n(j)|}{|n(i) \cup n(j)|} \tag{43}$$

wählen können. Das folgende (elementare) Beispiel illustriert, warum die Wahl auf (40) gefallen ist.

[18] Diese Zahl wird auch Jaccard-Index genannt (siehe die Artikeln [Jaccard:1901] und [Ahn:2010]).

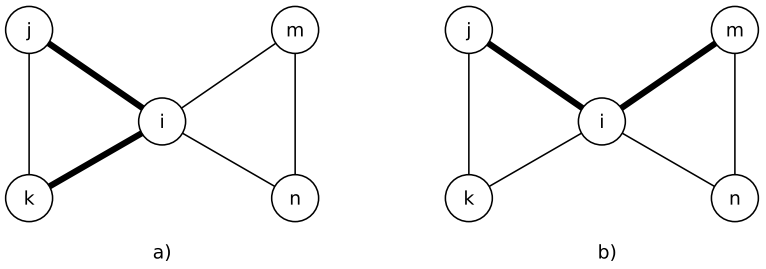


Abbildung 10: Vergleich der Ähnlichkeitsmaße

Die in Abbildung 10 dargestellten Kanten e_{ji} und e_{ki} bilden zusammen mit der Kante e_{jk} eine geschlossene Gruppe, während e_{ji} und e_{mi} eher zu verschiedenen Communities gehören.

$$\text{a) } \widetilde{Sim}(e_{ji}, e_{ki}) = \frac{|\{k,i\}| \cap |\{j,i\}|}{|\{k,i\}| \cup |\{j,i\}|} = \frac{1}{3} \qquad \text{b) } \widetilde{Sim}(e_{ji}, e_{mi}) = \frac{|\{k,i\}| \cap |\{i,n\}|}{|\{k,i\}| \cup |\{i,n\}|} = \frac{1}{3}$$

Diese Zugehörigkeit wird von $\widetilde{Sim}$ nicht erkannt, wohl aber von Sim:

$$\text{a) } Sim(e_{ji}, e_{ki}) = \frac{|\{k,i,j\}| \cap |\{j,i,k\}|}{|\{k,i,j\}| \cup |\{j,i,k\}|} = 1 \qquad \text{b) } Sim(e_{ji}, e_{mi}) = \frac{|\{k,i,j\}| \cap |\{i,n,m\}|}{|\{k,i,j\}| \cup |\{i,m,n\}|} = \frac{1}{5}$$

5.2.2 Ähnlichkeit zwischen zwei Clustern

In einer Link Community ist jedes Cluster eine Kantenpartition (vgl. Abschnitt 2.3). Ein *agglomerativer* Algorithmus zum hierarchischen Clustering fängt auf unterster Ebene mit den zu gruppierenden Objekten an (jede Kante ist ein Cluster) und verschmilzt ähnliche Objekte (in unserem Fall sind es die benachbarten Kanten) sukzessive zu immer größeren Clustern (also Kantenpartitionen C_ν) zusammen, bis beim vorletzten Schritt der Hier-

archie nur noch eine kleine Anzahl von Cluster übrigbleiben, die nicht miteinander zusammenhängen. Im letzten Schritt werden diese Cluster dann zu einem einzigen Cluster (dem gesamten Graphen) verschmolzen. Dabei muss zunächst festgelegt werden, wie die Ähnlichkeit zwischen zwei Clustern definiert ist. (Siehe auch das Online-Tutorial [Matteucci]):

Sei $\mathcal{C}_E = \{C_\nu\}_{1 \leq \nu \leq M}$ eine Kantenpartitionierung von $G = (V, E)$.

Seien $C_1 = (V_1, E_1)$ und $C_2 = (V_2, E_2)$ zwei Cluster (Kantenpartitionen), die benachbart sind, d.h. $V_1 \cap V_2 \neq \emptyset$. Beim sogenannten **single-linkage clustering** wird die Ähnlichkeit zwischen C_1 und C_2 definiert durch

$$Sim(C_1, C_2) := \begin{cases} \max\limits_{k \in V_1 \cap V_2} \left\{ \max\limits_{\substack{e_{ik} \in E_1 \\ e_{jk} \in E_2}} \{Sim(e_{ik}, e_{jk})\} \right\} & V_1 \cap V_2 \neq \emptyset \\ \\ 0 & \text{sonst} \end{cases} \tag{44}$$

5.2.3 Partitionsdichten

Für die Bewertung der Qualität einer Partitionierung $\mathcal{C}_E$ benötigen wir noch den Begriff der Partitionsdichte (vgl. [Ahn:2010]).

Seien $m_\nu = |E_\nu|$ und $n_\nu = |V_\nu|$. Dann ist $m = |E| = \sum\limits_{k=1}^{M} m_\nu$.
Die **lokale Partitionsdichte** eines zusammenhängenden Clusters C_ν ist

$$D_\nu = \begin{cases} \frac{m_\nu - (n_\nu - 1)}{n_\nu(n_\nu - 1)/2 - (n_\nu - 1)}, & n_\nu > 2 \\ \\ 0, & n_\nu = 2 \end{cases} \tag{45}$$

Die **globale Partitionsdichte** ist

$$D = \sum_{k=1}^{M} \frac{m_\nu}{M} \cdot D_\nu = \frac{2}{M} \sum_{k=1}^{C} m_\nu \cdot \frac{m_\nu - (n_\nu - 1)}{(n_\nu - 2)(n_\nu - 1)} \tag{46}$$

Für Bäume ($n_\nu > 2$) gilt wegen $m_\nu = n_\nu - 1$ immer $D_\nu = 0$. Für vollständige Teilgraphen gilt wegen $m_\nu = n_\nu(n_\nu - 1)/2$ immer $D_\nu = 1$.

5.2.4 Der Link-Community-Algorithmus

Sei $G = (P, E_G)$ der nach Algorithmus 5 oder 6 gebildete ungewichtete Produktgraph. In der initialen Partitionierung bildet jede Kante zusammen mit ihren beiden Knoten ein Cluster:

$$\mathcal{C}_E^{(0)} = \left\{ \left(\{i,j\}, e_{ij} \right)_{e_{ij} \in E} \right\}$$

---

1) Finde alle Paare von benachbarten Clustern $C_1^{(0)}, C_2^{(0)} \in \mathcal{C}_E^{(0)}$ und berechne den Wert $Sim\left(C_1^{(0)}, C_2^{(0)}\right)$. Erzeuge aus den Tripeln

$$\left(C_1^{(0)}, C_2^{(0)}, Sim\left(C_1^{(0)}, C_2^{(0)}\right) \right)$$

eine Similarity-Liste SL, sortiert nach Sim in nicht aufsteigender Reihenfolge.

2) Betrachte nun folgenden gewichteten Graphen:

$$\tilde{G} := (\mathcal{C}_E^{(0)}, S_C).$$

Die Knotenmenge von $\tilde{G}$ ist gerade die Clustermenge von G und die Kantenmenge S_C von $\tilde{G}$ wird über die Nachbarschaftsverhältnisse der Cluster von G definiert, d.h. die Kante $(C_1^{(0)}, C_2^{(0)}) \in S_C$ existiert dann und nur dann, wenn $Sim\left(C_1^{(0)}, C_2^{(0)}\right) > 0$ ist.

3) Wende die *Single-Linkage Hierarchical Clustering Methode* (siehe unten) auf $\tilde{G}$ an, um eine hierarchische Folge (genannt **Dendrogramm**) von Knoten-Partitionerungen $\left(\mathcal{K}_V^{(r)}\right)_{1 \le r \le H}$ für $\tilde{G}$ zu konstruieren (hier: $V = \mathcal{C}_E^{(0)}$). Diese ist äquivalent zu einer hierarchischen Folge von Kanten-Partitionerungen $\left(\mathcal{C}_E^{(r)}\right)_{1 \le r \le H}$ für G.

4) Wähle als Richtwert für den **Similarity Threshold** die Dendrogramm-Stufe d_r, bei der die globale Partitionsdichte $D^{(r)}$ maximal ist.

Algorithmus 7 : Link-Community-Algorithmus nach Ahn et. al. [Ahn:2010]

Das Dendrogramm kann bei dieser Schranke abgeschnitten werden. Damit erhalten wir das Clustering Ergebnis. Um das Ergebnis zu verbessern, kann diese Schranke im Anschluß noch variiert werden.

5.2.5 Single-Linkage Hierachical Clustering Methode

Wir führen für den Algorithmus 7 Schritt 3 eine Knotenpartitionierung (nach Johnson [Johnson:1967]) für $\tilde{G}$ durch. Diese wird dann genutzt als eine Kantenpartitionierung für G.

3a) Zu Beginn besteht $\mathcal{K}_V^{(0)}$ gerade aus der Knotenmenge von $\tilde{G}$, d.h. jeder Knoten ist ein Cluster. Die Anzahl der Cluster ist $N_C = |\mathcal{C}_E^{(0)}| = |E|$.

3b) Setze $r = 1$.

3c) Finde zwei Cluster (solange $N_C > 1$, müssen diese benachbart sein!) mit maximaler Similarity $0 \leq Sim_{\max} \leq 1$ und fasse diese zu einem neuen Cluster zusammen. Dadurch verringert sich N_c um 1. Setze $N_c = N_c - 1$.

3d) Berechne die Similarity-Werte zwischen dem neuen Cluster und seinen Nachbarn.

3e) Wiederhole Schritt 3c), bis alle Cluster mit Similarity $\geq Sim_{\max}$ zusammengefasst und isoliert sind.
(Alle übrigen Cluster bleiben unverändert.)
Setze die Dendrogramm-Stufe auf $d_r = Sim_{\max}$.
Die so entstandene Partitionierung ist $\mathcal{C}_E^{(r)}$.
Berechne dafür die globale Partitionsdichte $D^{(r)}$.

3f) Setze $r = r + 1$.

3g) Solange $N_c > 1$ ist, gehe nach Schritt 3c).

Algorithmus 8 : Single-Linkage Hierachical Clustering nach Johnson [Johnson:1967]

5.2.6 Zugehörigkeit zu einem Cluster

Gegenüber hierarchischen Knotenpartitionierungsalgorithmen hat der Link-Community-Algorithmus einen entscheidenden Vorteil: Er kann mit Overlap-Knoten umgehen.

In einer Kantenpartitionierung gehört jede Kante zu genau einem Cluster, aber ein Knoten

kann zu verschiedenen Clustern gehören (**Overlap-Knoten**). Ein Knoten $u \in G$ gehört zu einem Cluster $C_j \in \mathcal{C}_E$, wenn seine berührende Kante (u, v) zum Cluster C_j gehört.

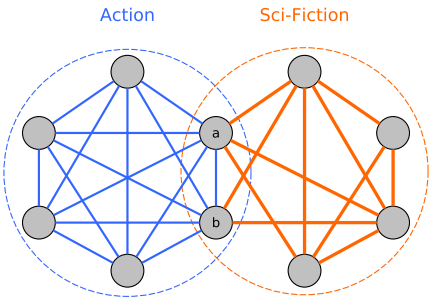


Abbildung 11: Overlap-Knoten

Beispiel: In Abbildung 11 gehört die Kante (a, b) zum Cluster "Action"-Filme, aber die beiden Filme a und b kann man sowohl zu den "Action"-Filmen als auch zu den "Science-Fiction"-Filmen zählen. Clusterzugehörigkeiten: $z_{Action}(a) = 5/8$, $z_{Sci-Fiction}(a) = 3/8$.

Der Vollständigkeit halber wollen wir hier noch ein Maß erwähnen, mit dem man die Zugehörigkeit eines Overlap-Knotens zu einem Cluster bewerten kann. Er wird (wie in [Ahn:2010] für sogenannte "Fuzzy-Community"-Methoden angedeutet) über folgende Gewichtung definiert:

Sei $\mathcal{C}_E = \{C_\nu\}_{1 \leq \nu \leq M}$ eine Kantenpartitionierung von $G = (V, E)$. Sei $u \in \bigcap_{j=1}^{\mu} C_j$ ein Overlap-Knoten. Für $k \in \{1, ..., \mu\}$ sei $deg_k(u)$ der Knotengrad von u innerhalb des Clusters C_k. Dann ist

$$z_k(u) := \frac{deg_k(u)}{deg(u)} \tag{47}$$

ein Gewicht für die **Clusterzugehörigkeit** von u.

Bemerkung: Wegen $\sum_{k=1}^{\mu} deg_k(u) = deg(u)$ gilt $\sum_{k=1}^{\mu} z_k(u) = 1$.

Da unklar ist, wie man am besten eine OMP baut, ist es notwendig, die resultierenden Cluster in ihrer Qualität zu bewerten.

6 Qualitätsmerkmale zur Bewertung der Clustering-Ergebnisse

6.1 Allgemeine Kriterien

Sei $G = (V, E)$ ein ungewichteter und ungerichteter Graph. Sei $\mathcal{C}_E = \{C_\nu\}_{1 \leq \nu \leq M}$ eine Kantenpartitionierung von $G = (V, E)$.

Zwei Sorten von Maßen (Qualität und Abdeckung) für die zwei Arten von Eigenschaften eines Knotens in einer Kantenpartitionierung (die reinen Cluster-Eigenschaften und die Overlap-Eigenschaften) werden von Ahn et. al. [Ahn:2010] vorgestellt.

Qualitätsmaße benötigen eine Reihe von **Metadaten**. Metadaten sind bespielsweise einzelne Stichwörter (**Keywords**), die bereits Informationen zur Kategorisierung der Knoten oder zur Anzahl ihrer Zugehörigkeiten zu Kategorien enthalten. Je mehr Metadaten verfügbar sind, desto besser kann man die Qualität der Clusterergebnisse bewerten.

Wir stellen hier zur Illustration nur die *Community Quality* vor. Es sei erwähnt, dass es noch die *Overlap Quality* gibt.

6.1.1 Qualität eines Clusters (Community Quality)

Basierend auf einem verfügbaren Metadatum sei ein Ähnlichkeitsmaß $0 \leq \mu(u, v) \leq 1$ zwischen zwei beliebigen Knoten $u, v \in G$ definiert.

Dann ist der *Anreicherungsfaktor (enrichment factor)* eines Clusters $C \in \mathcal{C}_E$ definiert durch den Quotienten[19]

$$\frac{\langle \mu(u,v) \rangle_{(u,v) \in C}}{\langle \mu(u,v) \rangle_{(u,v) \in G}}. \tag{48}$$

Dieser Quotient ist in der Regel ≥ 1. Je größer dieser Wert ist, desto enger ist die Bindung der Knoten innerhalb des Clusters C.

6.1.2 Community Coverage

Wir haben bei der Kantenpartitionierung in der untersten Stufe des Dendrogramms damit angefangen, jede einzelne Kante als Cluster aufzufassen. Für die Bewertung des Link-

[19] $\langle \mu(u,v) \rangle_{u,v \in X}$ ist der Mittelwert von $\mu(u, v)$ über alle vorhandenen Kanten $(u, v) \in X$

Community-Algorithmus ist es daher sinnvoll, diese trivialen Cluster, die genau zwei Knoten enthalten, nicht mitzuzählen. Wir betrachten deshalb nur nicht-triviale Cluster, die mindestens drei Knoten bzw. mindestens zwei Kanten enthalten.

Sei $V^* \subset V$ die Menge aller nicht-isolierten Knoten[20] von G. Sei n_C die Anzahl aller Knoten, die in nicht-trivialen Clustern enthalten sind. Dann ist die Maßzahl für die Community Coverage definiert durch

$$\frac{n_C}{|V^*|}. \tag{49}$$

Mit dieser Zahl können wir sehen, welcher Anteil des Produktnetzwerks G durch den Link-Community-Algorithmus abgearbeitet wurde.

Bemerkung:

Wenn man in diesem Zusammenhang die Leistung der einseitigen Projektion, die ja gerade den Anfangszustand (die Menge der trivialen Cluster) generiert hat, mit berechnen wollte, müsste man eigentlich die Knoten in den trivialen Clustern mitzählen. Doch dann wäre wegen $n_C = |V^*|$ diese Maßzahl immer gleich 1 und nicht aussagekräftig.

6.1.3 Overlap Coverage

Sei $u \in V^*$ und sei $m_C(u)$ die Anzahl aller nicht-trivialen Cluster, in denen u enthalten ist. Nach Ahn et. al. [Ahn:2010] ist die Maßzahl für den Overlap Coverage definiert durch

$$\frac{\sum\limits_{u \in V^*} m_C(u)}{|V^*|}. \tag{50}$$

Mit dieser Maßzahl kann man sehen, wie hoch der Informationsgehalt des Teils von G ist, den der Link-Community-Algorithmus abgearbeitet hat.

Bemerkung:

Um die Leistung der einseitigen Projektion mitzubewerten, zählen wir (für unsere Filmedatenbank) die trivialen Cluster mit, d.h. wir wählen für den Ausdruck $m_C(u)$ die *Anzahl aller Cluster*, die u enthalten.

[20] Verschiedene Abwandlungen der einseitigen Projektion können natürlich verschiedene Werte für $|V^*|$ hervorbringen.

6.2 Ground Truth

Neben den oben erwähnten (bekannten und erprobten Maßen) haben wir zusätzlich ein neues Maß entwickelt. Da bei unserer Filmedatenbank bis auf den Filmtitel sonst keine Metadaten wie etwa das Genre oder der Name der Hauptdarsteller oder des Regisseurs zur Verfügung stehen, können wir die Maßzahl für Community Quality (48) nicht verwenden. Wir greifen daher auf das folgende in Zweig [Zweig:2010] vorgeschlagene Kriterium zurück:

Für eine Staffel einer Serie sollten ihre besten Freunde möglichst aus der gleichen Serie stammen.

Wir formulieren diese Idee um in die Sprache des Clustering:

Die Filme einer Reihe bzw. die Staffeln einer Serie sollten möglichst vollständig in einem Cluster enthalten sein.

Dabei differenzieren wir noch die beiden besten Fälle

- **vollständige Inklusion:** Alle Filme einer Reihe bzw. alle Staffeln einer Serie sind *vollständig* in einem Cluster enthalten.
- **fast-vollständige Inklusion:** Alle Filme einer Reihe bzw. alle Staffeln einer Serie *bis auf eine* sind in einem Cluster enthalten.

Demnach darf ein Cluster selbstverständlich auch die Staffeln von mehrere Serien enthalten oder eine vollständige Serie als Overlap-Knoten in verschiedenen Clustern vorkommen. Uns interessiert im Prinzip nur, ob die Staffeln, die zu einer Serie gehören, aufgesplittet werden oder nicht.

Die fast-vollständige Inklusion wollen wir in unsere Betrachtung miteinbeziehen, damit auch solche Serien berücksichtigt werden, bei denen eine Staffel p_x verspätet erschienen ist, so dass die Kunden noch nicht die nötige Zeit hatten sich diese anzuschauen und zu bewerten. In so einem Fall würde der Knotengrad $deg(p_x)$ sehr klein sein und dadurch wegen der Ungleichung (37) den *leverage*-Wert (und somit auch den $s_{\max}$-Wert) *fälschlicherweise* nach unten drücken, so dass nach der einseitigen Projektion (nach Abschnitt 5.1.1 bzw. 5.1.2) die Verbindungen zu p_x wegfallen würden.

Als Maßzahlen für die *Ground-Truth* definieren wir den Anteil der Serien, die als solche vollständig bzw. fast-vollständig erkannt werden.

Bemerkung:

Das Kriterium der vollständigen Inklusion ist nicht ganz unproblematisch: Eine hierarchische Kantenpartitionierung kann theoretisch auch dazu führen, dass man am Ende ein einziges Cluster mit dem gesamten Produktnetzwerk als Inhalt erhält. In so einem Fall wäre die Bedingung der vollständigen Inklusion immer erfüllt. Man könnte diesen Spezialfall algorithmisch von vorn herein ausschließen, indem man eine Mindestanzahl von Clustern N_C im Algorithmus 7 vorschreibt.

7 Qualitätsmerkmal zur Bewertung des Empfehlungssystems

Root Mean Square Error (RMSE)

Zur Bewertung der Güte unserer ermittelten Cluster müssen wir einen Weg finden, das Ergebnis mit den Daten aus der originalen Filmedatenbank zu vergleichen.

In der mathematischen Statistik dient die mittlere quadratische Abweichung der Messung der Abweichung eines Schätzers von dem zu schätzenden Wert. Wir halten uns im Wesentlichen an die Definition des Schätzer der Varianz einer Messgröße [Krengel:2005], wobei hier nicht auf die Erwartungstreue eingegangen wird.

Wir gehen folgendermaßen vor: Als den zu schätzenden Wert wählen wir das Rating, das ein Kunde $t_j \in T$ für ein Film $p_x \in P$ geben würde, der in der reduzierten Filmedatenbank $B = (T \cup P, E)$ gar nicht vorkommt, da er in der Originaldatenbank ≤ 3 war. Diesen Schätzwert μ vergleichen wir dann mit den vom Kunden t_j tatsächlich abgegebenen Ratings, sofern sie vorhanden sind und quadrieren die Abweichungen. Diesen Vorgang wiederholen wir für jeden einzelnen Kunden addieren die Terme auf und bilden am Ende davon den mittleren quadratischen Fehler nach Division durch die Gesamtzahl aller vorhandenen Summanden.

Anhand eines kleinen (ausgedachten!) Zahlenbeispiels soll die Idee verdeutlicht werden:

Der Kunde t_1 habe folgendes Rating abgegeben:

Film-Id:	25	27	28	55	58	72	88	90
Rating:	5	4	2	3	4	5	1	2

In der reduzierten Datenbank B bliebe davon nur noch

Film-Id:	25	27	28	55	58	72	88	90
Rating:	5	4	×	×	4	5	×	×

was zu einem Schätzwert von $\mu = (5 + 4 + 4 + 5)/4 = 4.5$ für den Kunden t_1 führt.

Nun betrachten wir für jeden Film p_x, der in der zweiten Liste eine Bewertung erhält, die Liste der Filme in dem zugehörigen Cluster $C(p_x)$. Sei also etwa

C(25)	27	28	55	72	95	
C(27)	25	55	72	91	101	...
C(58)	18	25	72	91	102	...
C(72)	21	28	55	88	104	...

Uns interessieren aus dem ersten Cluster $C(25)$ nur solche Filme, die vom Kunden t_1

wirklich bewertet wurden, also die Filme $27, 28, 55$ und 72 mit den echten Ratings $4, 2, 3$ und 5. Davon bilden wir die quadratischen Abweichungen des Schätzwertes $\mu_1 = 4.5$ von den echten Ratings:

$$\tau = (4.5 - 4)^2 + (4.5 - 2)^2 + (4.5 - 3)^2 + (4.5 - 5)^2$$

Zu diesem Wert addieren wir aus dem zweiten Cluster $C(27)$ die quadratischen Abweichungen für die Filme $25, 55, 72$ mit den echten Ratings $5, 3, 5$:

$$\tau = \tau + (4.5 - 5)^2 + (4.5 - 3)^2 + (4.5 - 5)^2$$

Diese Prozedur wiederholen wir für alle weiteren Kunden $t_j \in T$, um τ weiter aufzuaddieren. Selbstverständlich werden diese andere Schätzwerte μ_j generieren.

Am Ende bestimmen wir den mittleren quadratischen Fehler (mean square error MSE) durch Division durch die Anzahl N_S aller Summanden.

Der **root mean square error** ist dann gegeben durch

$$\text{RMSE} = \sqrt{\frac{\tau}{N_S}}$$

8 Vergleich der vorgestellten Verfahren

Basierend auf unserem reduzierten Netflix-Datensatz $B = (T \cup P, E)$ mit $|P| = 17.770$ Filmen[21], die von $|T| = 20.000$ Kunden insgesamt etwa $|E| \approx 2,3$ Millionen Bewertungen[22] erhalten haben, führen wir Clustering-Tests nach den beiden beschriebenen OMP-Verfahren durch und vergleichen die Ergebnisse.

Zusammenfassung der gesamten Clustering-Verfahren (bisher):

Eingabe: $B = (T \cup P, E)$

1. **Verfahren**:
   (a) **OMP-Algorithmus 3** liefert $\hat{G} = (P, \hat{E}_G)$ mit *leverage* als Kantengewichte.
   (b) Reduktion und Umgewichtung der Kanten nach **Algorithmus 5** (Reziprozität) liefert $G = (P, E_G)$.
   (c) Hierarchisches Clustering von $G = (P, E_G)$ durch **Algorithmus 7** und Abschneiden des Dendrogramms.
   (d) Bewertung des Clustering nach den Maßzahlen für die vollständige und fastvollständige Inklusion und für die Community und Overlap Coverage.
2. **Verfahren**:
   (a) **OMP-Algorithmus 4** liefert $\hat{G} = (P, \hat{E}_G)$ mit $s_{\max}$ als Kantengewichte.
   (b) Reduktion und Umgewichtung der Kanten nach **Algorithmus 6** (maximaler Cluster-Koeffizient) liefert $G = (P, E_G)$.
   (c) Hierarchisches Clustering von $G = (P, E_G)$ durch **Algorithmus 7** und Abschneiden des Dendrogramms.
   (d) Bewertung des Clustering nach den Maßzahlen für die vollständige und fastvollständige Inklusion und für die Community und Overlap Coverage.

Die Implementierungen sind in der Programmiersprache JAVA (Version 1.6) realisiert worden. Alle Tests sind auf einem Desktop-PC mit einer Intel Pentium Quad-Core Q9550 CPU mit $4 \times 2,83$ GHz und 8 GB RAM durchgeführt worden. Die längste Rechenzeit (etwa 5 Stunden) nahm die Generierung der 5000 Realisierungen[23] der FDSM Zufallsgraphen für den Schritt (a) in Anspruch.

[21] Eine Staffel einer Serie wird hier auch als ein Film gezählt.

[22] Die Bewertungsskala läuft von 1 (sehr schlecht) bis 5 (sehr gut). Es wurden nur Bewertungen > 3 in Betracht gezogen.

[23] In den Tests wurden für 1a) und 2a) selbstverständlich die gleichen Realisierungen genommen, um eine Vergleichbarkeit der beiden Verfahren zu gewährleisten.

1a) und 2a) führen beide auf die gleiche Kantenzahl $|\hat{E}_G| = 22.729.936$ für das Produktnetzwerk $\hat{G} = (P, \hat{E}_G)$, aber verschiedene Kantengewichte.

	Test 1: Verfahren 1 ($\delta_R = 1/2$)	Test 2: Verfahren 1 ($\delta_R = 1/3$)	Test 3: Verfahren 2
$\|E_G\|$ (Anteil v. $\|\hat{E}_G\|$)	24835 (0.1%)	119263 (0.5%)	65710 (0.3%)
# aller Cluster	7789	48856	11703
# nicht-triv. Cluster	1948	8370	3416
# isolierter Filme	8866	4274	9134
Maßzahl vollst. Inkl.	$86/116 \approx 0.74$	$102/116 \approx 0.88$	$98/116 \approx 0.84$
Maßzahl fast-vollst. Inkl.	$29/116 \approx 0.25$	$13/116 \approx 0.11$	$16/116 \approx 0.14$
Community Coverage	59%	65%	69%
Overlap Coverage	2.37	7.03	4.23
RMSE	0.8	1.008	0.72

Tabelle 8: Ein Vergleich der Methoden

Zum Verfahren 1: Gegenüber Test 1 wird im Test 2 das Kriterium für die Liste der besten Freunde abgeschwächt. Dadurch entstehen insgesamt mehr Kanten. Das führt unweigerlich zu einer höheren Anzahl von Cluster und zu einem höheren durchschnittlichen Overlap pro Knoten. Für einen Film werden dadurch in der Regel mehr verschiedene Empfehlungen gegeben.

Zum Verfahren 2: Der Test 3 liefert die höchste Abdeckung des Produktnetzwerks durch die Clusteranalyse. Das Kriterium der vollständigen Inklusion ist hinreichend gut erfüllt, besser als beim Verfahren 1 mit $\delta_R = 1/2$. Doch es liefert auch die größte Anzahl an isolierten Knoten. Das Verfahren 1 mit $\delta_R = 1/3$ liefert nach den allgmeinen Qualitätsmaßen (aus Abschnitt 6.1) scheinbar das beste Ergebnis mit den wenigsten isolierten Knoten, doch bei Hinzunahme des letzten Kriteriums (RMSE aus Abschnitt 6.3), in dem wir mit Informationen aus dem Originaldatensatz B arbeiten, was uns ja schließlich zur Verfügung steht, sehen wir, dass insgesamt das Verfahren 2 am besten abschneidet. Das Verfahren 2 liefert im Großen und Ganzen Empfehlungen für weniger Filme, dafür arbeitet es genauer.

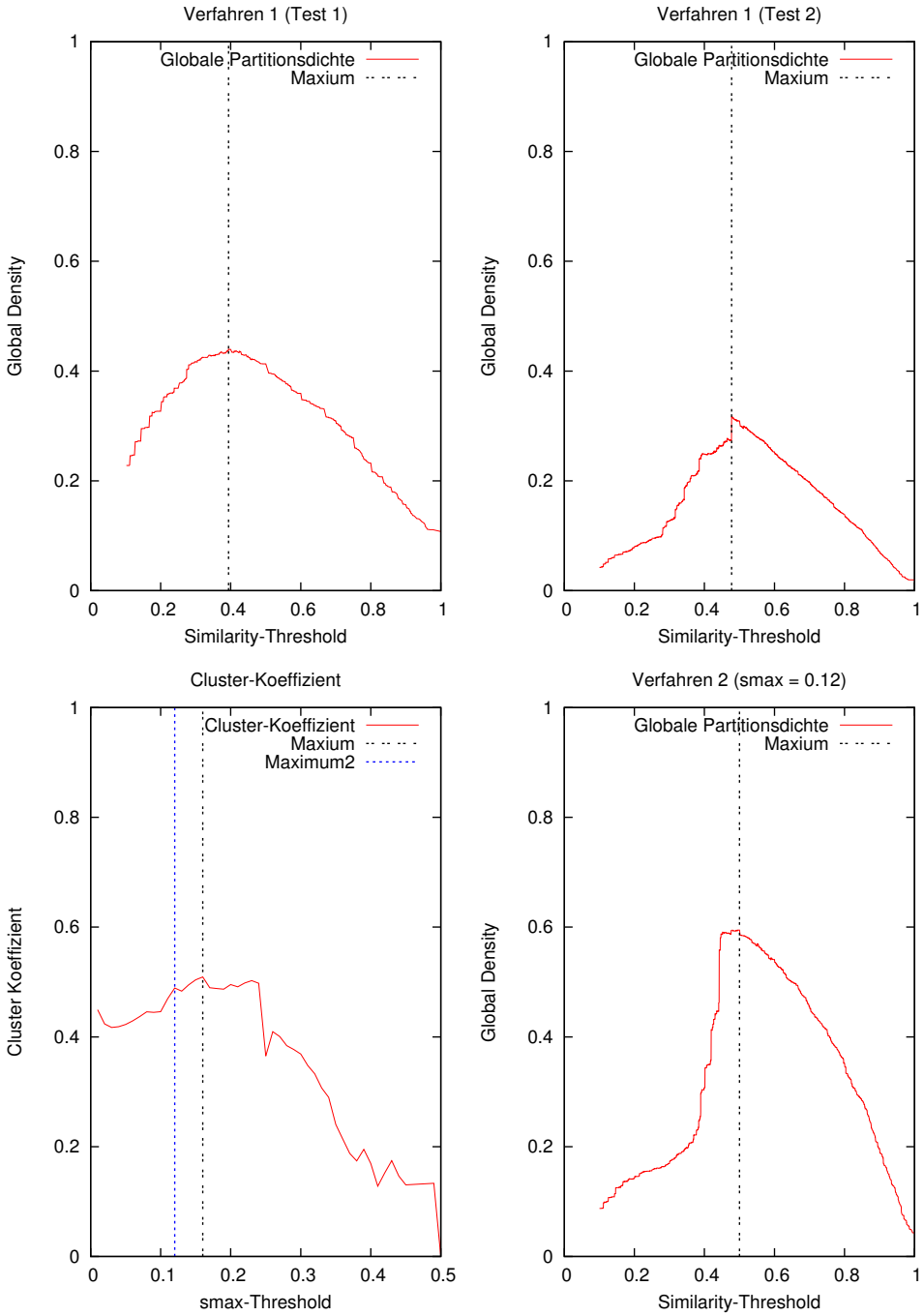


Abbildung 12: **Oben: Verfahren 1**, unterschiedliche Verteilungen der globalen Partitionsdichten für $\delta_R = 1/2$ (links) bzw. $\delta_R = 1/3$ (rechts). Ermittlung der Similarity-Stufe nach der Maximalitätsbedingung in Algorithmus 7 Schritt 4. **Unten: Verfahren 2**, Maximalitätsbedingung in Algorithmus 6 Schritt 4: Wir wählen das nächstgelegene lokale Maximum bei $s_{\max} = 0.12$, weil für das globale Maximum bei $s_{\max} = 0.16$ die Anzahl der Kanten im Produktnetzwerk $G = (P, E_G)$ zu gering ist (links). Ermittlung der Similarity-Stufe nach der Maximalitätsbedingung in Algorithmus 7 Schritt 4 (rechts).

9 Sortierte Darstellung der Cluster

Das Empfehlungssystem hat die Aufgabe, bei Eingabe eines Filmtitels eine Liste von Filmen mit ähnlichem Inhalt auszugeben. Auf das kantenpartitionierte Produktnetzwerk $G = (P, E_G)$ übertragen bedeutet das: Finde zu einem Knoten p_x dasjenige Cluster $C_j \in \mathcal{C}_E$, das am ehesten die Knoten enthält, die ähnlich zu p_x sind, und sortiere anschliessend die Cluster der Reihe nach (nicht aufsteigend). Für eine solche Bewertung bieten sich auf dem ersten Blick zwei bereits vorgestellte Maße an: die lokale Partitionsdichte D_j und die Clusterzugehörigkeit $z_j(p_x)$.

Eine Serie, die nur aus zwei Staffeln besteht oder ein Film, der zwei Teile hat, könnte zu einem minimalen Cluster C_j mit einer Kante und zwei Knoten p_x und p_y führen. Diese haben in der Regel den größtmöglichen inhaltlichen Zusammenhang, doch wird das von der lokalen Partitionsdichte ($D_j = 0$ wegen $n_j = 2$) falsch dargestellt. Wäre p_x zufällig noch Teil eines anderen größeren Clusters C_k, in dem p_x über mehr als eine Verknüpfung verfügt, würde die Gewichtung der Clusterzugehörigkeit zugunsten von C_k ausfallen: $z_k(p_x) > z_j(p_x)$.

Wir führen daher ein neues Maß $\Phi(C)$ ein, um die inhaltliche Korreliertheit von Filmen innerhalb eines Clusters C zu quantifizieren. Dabei gehen wir auf eine Idee zurück, die wir schon in Abschnitt 4.2 kennengelernt haben. Wir vergleichen die *Wirkung* des Clusters auf den gegebenen bipartiten Graphen B mit der *Wirkung* des Clusters auf seinen korrespondierenden Zufallsgraphen $B' \in \tilde{\mathcal{G}}(L, R)$ (vgl. Definition (22) aus Abschnitt 4.4).

Gegeben:

- Bipartiter Graph $B = (T \cup P, E)$.
- Eine Folge von (durch Random Swaps entstandene) bipartite Graphen $B'_j = (T \cup P, E'_j)$.
- Die durch den Link-Community-Algorithmus 7 entstandene Kantenpartitionierung $\mathcal{C}_E$ des ungewichteten OMP-Produktgraphen $G = (P, E)$.
- Wahl eines Threshold $\theta = 60\%$ für die **Attraktivität** eines Clusters.

Für ein gegebenes Cluster $C \in \mathcal{C}_E$ definieren wir:

- α: Anzahl der Kunden aus B, die mindestens θ% Filme aus C mögen.
- β: Anzahl der Kunden aus B, die mindestens einen Film aus C mögen.

- α': Gesamtzahl der Kunden aus allen $B'_j \in \tilde{\mathcal{G}}(L,R)$, die mindestens θ% Filme aus C mögen.
- β': Gesamtzahl der Kunden aus allen $B'_j \in \tilde{\mathcal{G}}(L,R)$, die mindestens einen Film aus C mögen.
- $\Phi(C)$ ist das Doppelverhältnis zwischen α/β und α'/β', also:

$$\Phi(C) = (\alpha \cdot \beta')/(\alpha' \cdot \beta).$$

α/β beschreibt in gewisser Weise die Wahrscheinlichkeit dafür, dass ein Kunde aus B mindestens θ% der Filme aus C mag, wenn er mindestens einen Film aus C mag.

α'/β' beschreibt die gleiche Wahrscheinlichkeit in einer grösseren Menge von Realisierungen von B'.

$\Phi(C)$ beschreibt in gewisser Weise die Korreliertheit von Filmen aus einem Cluster C. Am Ende sortieren wir alle Cluster C nach $\Phi(C)$ in nicht aufsteigender Reihenfolge.

Aus Platzgründen listen wir nur einen Teil der auf diese Weise sortierten Cluster für ein Verfahren (nämlich das $s_{\max}$-basierte Verfahren) auf. Im Anhang sehen wir die ersten 99 Cluster.

10 Einige Aspekte der Implementierung

Zu guter Letzt wollen wir noch auf die wichtigsten Punkte der Programmierung kommen, da diese einen nicht unerheblichen Teil einer praxisorientierten Arbeit darstellt.

10.1 Effiziente Ermittlung der $coocc_{\mathbf{FDSM}}(p_x, p_y)$-Werte

Für die Generierung der Stichproben $B' \in \tilde{\mathcal{G}}(L, R)$ (siehe Gleichung 22) nach dem Self-Loop-Algorithmus (Algorithmus 2) haben wir uns im Abschnitt 4.5 dafür entschieden, dass wir für unseren Datensatz pro Stichprobe etwa $t_{sample} = \lfloor |T| \cdot \log |T| \rfloor \approx 200.000$ Swap-Schritte durchführen. Es sei daran erinnert, dass diese Zahl auch die *self-loops* im Übergangsgraphen mitzählt. Die Anzahl s_0 der *echten* Swaps ist bei unseren Test deutlich geringer: $s_0 \approx \frac{2}{3} \cdot t_{sample} \leq 134.000$.

Um für jede einzelne Stichprobe $B' = (T \cup P, E')$ den Wert von $coocc_{\text{FDSM}}(p_x, p_y)$ für alle Knotenpaare $p_x, p_y \in P$ zu ermitteln, gibt es mindestens drei Methoden:

1.) Für jede neu erzeugte Stichprobe B' berechnen wir diesen Wert für alle Knotenpaare $p_x, p_y \in P$ neu. Nach dem naiven Ansatz würde man der Reihe nach jeden Knoten aus P mit jedem anderen Knoten aus P vergleichen und

$$\frac{|P| \cdot (|P| - 1)}{2} \cdot |T|$$

Vergleiche pro Stichprobe benötigen.

Bei uns beträgt dieser Wert etwa $3,16 \cdot 10^{12}$, da $|T| = 20.000$ und $P = 17.770$.

2.) Nützt man die Information über die vorhandenen Kanten E' aus, so würde man nur die indirekt über T vorbundenen Knotenpaare aus P betrachten. Praktisch bedeutet das, man vergleicht nur die in der Transaktionsdatenbank B' zeilenweise aufgeführten Knoten miteinander. Zur Veranschaulichung eine kleine Illustration:

t_1	p_1	p_2	p_3	p_4	p_5	...
t_2	p_1	p_3	p_6	p_8	p_9	...
$\vdots$						

Der Kunde t_1 hat insgesamt $deg(t_1)$ Filme bewertet. Für die Filme in seiner Liste genügen

$$\frac{1}{2} \cdot deg(t_1) \cdot (deg(t_1) - 1)$$

Zählungen, um die *coocc*-Werte für jedes verbundene Knotenpaar zu inkrementieren. Im Schnitt werden pro Kunde $t \in T$ also

$$\frac{1}{2} \cdot \overline{deg(t)} \cdot (\overline{deg(t)} - 1)$$

Zählungen benötigt, wobei $\overline{deg(t)}$ der mittlere Knotengrad von T darstellt. Insgesamt würde man also

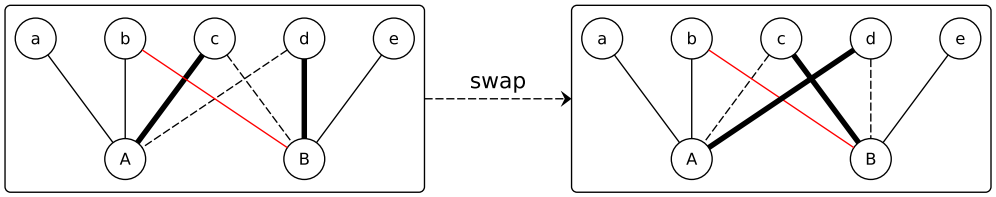
$$\frac{1}{2} \cdot \overline{deg(t)} \cdot (\overline{deg(t)} - 1) \cdot |T| \tag{51}$$

Zählungen pro Stichprobe B' benötigen.

Bei uns beträgt dieser Wert etwa $136 \cdot 10^6$*, da* $\overline{deg(t)} = 117$ *und* $|T| = 20.000$*.*

3.) Wir schlagen eine weitere Methode vor, die ausgehend von den Informationen aus der ersten Stichprobe $B_1 \in \tilde{\mathcal{G}}(L, R)$, von der wir ja einmalig eine *coocc*-Matrix $\mathbf{C}_1$ nach der Methode 2 anfertigen können[24], unter Ausnutzung der Swap-Information die Einträge von $\mathbf{C}_1$ nach jedem echten Swap-Schritt in eine neue Matrix $\mathbf{C}'$ überführt. Nach genau s_0 echten Swap-Schritten erhält man die *coocc*-Matrix $\mathbf{C}_2$, die zur zweiten Stichprobe $B_2 \in \tilde{\mathcal{G}}(L, R)$ gehört.

Zur Veranschaulichung dieser Idee eine kleine Skizze mit den Kunden $A, B \in T$ und den Filmen $a, b, c, d, e \in P$:



a) b)

Die dick markierten Kanten seien die tatsächlich existierenden Kanten vor bzw. nach dem Swap-Schritt. Die Einträge in der *coocc*-Matrix würden sich folgendermaßen verändern:

$coocc(a, d) \uparrow$	$coocc(a, c) \downarrow$
$coocc(b, c) \downarrow$	$coocc(b, d) \uparrow$
$coocc(c, e) \uparrow$	$coocc(d, e) \downarrow$

[24] $\mathbf{C}_1$ kann man als eine obere Dreiecksmatrix (Diagonaleinträge = 0) der Dimension $|P| \times |P|$ aufstellen, die sämtliche $coocc_{\text{FDSM}}(p_x, p_y)$-Werte für alle Knotenpaare $p_x, p_y \in P$ enthält.

Das Symbol $\uparrow$ bedeutet eine Inkrementierung $(+1)$ und $\downarrow$ bedeutet eine Dekrementierung (-1) des jeweiligen *coocc*-Wertes.

Zusätzlich gilt aus der Sicht von A: $coocc(b,c)\downarrow$ $coocc(b,d)\uparrow$
und aus der Sicht von B: $coocc(b,d)\downarrow$ $coocc(b,c)\uparrow$
Aus der Sicht von b hat sich also nichts geändert. Dem wäre nicht so, würde die rote Kante (b,B) in der Skizze fehlen!

Die Summe der Veränderungen nach einem Swapschritt ist gleich 0, deshalb bleibt die Gesamtsumme aller Einträge der *coocc*-Matrix auch unverändert.

Aus der Sicht von A heisst das, dass die Menge seiner (exklusiven) Nachbarn sich folgendermaßen durch den Swap verändern:

$$n(A) = \{a,b,c\} \;\rightsquigarrow\; n(A) = \{a,b,d\}$$

Das sind (maximal) $deg(A) - 1$ Inkrementierungen und genauso viele Dekrementierungen für die unveränderten Knoten a und b.

Entsprechend verändert sich die Menge der Nachbarn von B:

$$n(B) = \{b,d,e\} \;\rightsquigarrow\; n(B) = \{b,c,e\}$$

Das ergibt (maximal) $2(deg(B) - 1)$ Veränderungen in der *coocc*-Matrix.

Im Durchschnitt sind (maximal) $4 \cdot (\overline{deg(t)} - 1)$ Veränderungen in der *coocc*-Matrix pro Swap-Schritt notwendig. Da insgesamt s_0 Swap-Schritte für den Übergang von einer Stichprobe zur nächsten vorgesehen sind, haben wir hier einen Aufwand von

$$4 \cdot (\overline{deg(t)} - 1) \cdot s_0 \tag{52}$$

pro Stichprobe B'.

Bei uns ist dieser Wert kleiner als $62,2 \cdot 10^6$, da $\overline{deg(t)} = 117$ und $s_0 \leq 134.000$. Ein Vergleich mit der Methode 2 bestätigt unsere Beobachtung, da die Rechenzeit für die Methode 2 etwa doppelt so hoch ist wie für die Methode 3.

Die Methode 3 ist der Methode 2 vorzuziehen, wenn (vgl. (52) mit (51))

$$4 \cdot (\overline{deg(t)} - 1) \cdot s_0 < \frac{1}{2} \cdot \overline{deg(t)} \cdot (\overline{deg(t)} - 1) \cdot |T|$$

ist oder, anders ausgedrückt, wenn

$$s_0 < \frac{1}{8} \cdot \overline{deg(t)} \cdot |T|$$

ist. Diese Bedingung ist für unseren reduzierten Datensatz mit $|T| = 20.000$, $\overline{deg(t)} = 117$ und $s_0 \leq 134.000$ erfüllt.

Hinweise zur Implementierung in JAVA

Zur Speicherung der *coocc*-Matrix verwenden wir in JAVA ein `int`-Array der Dimension 2:

```
int P=17770;
int[][] cooccurence_matrix = new int[P][P];
```

Dabei werden für die aktuellen $coocc(p_x, p_y)$-Werte der Knotenpaare (p_x, p_y) das linke obere Dreieck `cooccurence_matrix[i][j]` mit Indizes `i < j` verwendet. Im rechten unteren Dreieck `cooccurence_matrix[i][j]` mit Indizes `i > j` speichern wir für jedes Knotenpaar (p_x, p_y) die Zwischensumme für die Berechnung des zu schätzenden Erwartungswertes (23).

Zur Speicherung des bipartiten Graphen B bietet es sich an, für jeden einzelnen Kunden einen Vektor von `boolean`-Werten

```
int P=17770;
int T=20000;
boolean[][] t = new boolean[T][P];
```

zu verwenden, da es sich ja um einen 0-1-Datensatz (Hat der Kunde einen Film positiv bewertet oder nicht?) handelt. Diese Darstellung würde von vornherein $17770 \cdot 20000$ Bytes reservieren. In JAVA verwendet man für diesen Zweck geschickterweise einen `BitSet`-Vektor[25] [Krueger:2009]

```
int T=20000;
BitSet[] t = new BitSet[T];
```

der

- pro Film nur ein Bit zur Markierung benötigt (Speicherverbrauch also nur $\frac{1}{8}$ soviel wie für `boolean` [Ullenboom:2009]) und
- für jeden einzelnen Kunden $t_j \in T$ dynamisch zur Laufzeit nur soviel Speicher reserviert, wie für die Darstellung der bewerteten Filmeliste nötigt ist.

Durch das Vorhandensein von mengenorientierten Operationen ist es besonders einfach, auf die Existenz von Elementen hin zu überpüfen (erforderlich für Swapbarkeitstest).

[25] Die `BitSet`-Klasse ist Teil von `java.util.*`.

10.2 Trove - eine Bibliothek für High Performance Computing

Das Ergebnis der einseitigen Projektion (Algorithmen 5 bzw. 6), also das ungewichtete Produktnetzwerk $G = (P, E_G)$ wollen wir in einer Datenstruktur auf dem Hauptspeicher unterbringen, so dass wir jederzeit darauf zugreifen können während der Berechnung der Similarities und bei der Verschmelzung der Kanten zu immer größeren Clustern im Algorithmus 7.

Eine Speicherung von $G = (P, E_G)$ in Form einer vollen Adjazenzmatrix mit Hilfe eines zweidimensionalen Arrays wäre sehr ineffizient, weil darin sehr viele Nullen enthalten wären. Die JAVA-Containerklasse[26]

```
java.util.HashMap<KeyType,MappedValueType>
```

bietet eine dynamische Möglichkeit, Schlüssel-Werte-Paare zusammenhängend zu speichern. Als Schlüsselwerte vom Type `KeyType` wählen wir die Elemente $p_x \in P$ (in Form der Film-IDs von 1 bis 17.770) und als Werte zu jedem einzelnen Schlüssel vom Typ `MappedValueType` definieren wir die Liste der exklusiven Nachbarn $n(p_x)$ (natürlich auch in Form der Film-IDs) durch

```
java.util.HashSet<KeyType>
```

Isolierte Knoten, für die es gar keine Nachbarn gibt, müssen gar nicht mit aufgenommen werden. Für die Darstellung der ganzen Zahlen von 1 bis 17.000 genügt im Prinzip der Datentyp `short` (16 Bits). Leider akzeptieren diese beiden Klassen ausschließlich die Wrapperklasse `java.lang.Short` und nicht den speichereffizienteren primitiven Datentypen `short`.

Um unter anderem dieses Manko zu beseitigen, wurde Trove [Friedman] entwickelt[27]. Von dieser Bibliothek verwenden wir

```
import gnu.trove.set.hash;
import gnu.trove.map.hash;
 ...
TShortObjectHashMap<TShortHashSet> hashmap =
   new TShortObjectHashMap<TShortHashSet>();
```

Damit erreichen wir einen um 2/3 verminderten Speicherbedarf im Gegensatz[28] zu den JAVA Containerklassen (2 GB gegenüber 6 GB für unser Produktnetzwerk)!

[26] vgl. `http://docs.oracle.com/javase/6/docs/api/java/util/HashMap.html`
[27] vgl. `http://trove4j.sourceforge.net/html/overview.html`
[28] vgl. `http://trove4j.sourceforge.net/html/benchmarks.shtml`

10.3 Effiziente Berechnung der Similarities und Verschmelzungen

Im ersten Schritt von Algorithmus 7 müssen wir für jedes Clusterpaar $(C_1, C_2) \in \mathcal{C}_E^{(0)}$ den Wert $Sim(C_1, C_2)$ ermitteln.

Bemerkung:

Existieren wie etwa in Abbildung 13 zwischen zwei Knoten i ind j mehrere verbindende "Zwischenknoten" a, b, c, d, so gilt

$$\begin{aligned} Sim(e_{ia}, e_{ja}) &= Sim(e_{ib}, e_{jb}) = \\ Sim(e_{ic}, e_{jc}) &= Sim(e_{id}, e_{jd}). \end{aligned}$$

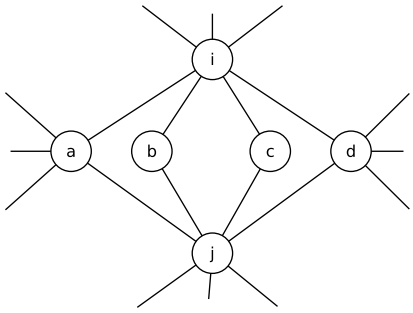


Abbildung 13: Ähnlichkeit zwischen zwei Knoten

Die Definition (40) wird in der Literatur [Ahn:2010] auch dazu benutzt, die Ähnlichkeit zwischen zwei Knoten zu bestimmen. Diese sollte natürlich unabhängig sein von der Wahl der verbindenden Kanten. Man bekommt damit auf einen Schlag die Ähnlichkeit aller Kantenpaare e_{ik}, e_{jk} für zwei Knoten i und j, die über die Menge der Zwischenknoten k miteinander verbunden sind.

Es genügt daher, "nur" für jedes Knotenpaar $(p_x, p_y) \in P$ den Similarity-Wert $Sim(p_x, p_y)$ zu ermitteln werden. Wir erstellen daraus eine globale Liste SL_P von Similarity-Werten, sortiert nach Sim in nicht aufsteigender Reihenfolge. Das Threshold $Sim_{\max}$ im Schritt 3e von Algorithmus 8 stellt immer nur eine untere Schranke dar, ab der die Verschmelzung zu einem neuen Cluster stattfindet. Ein Blick auf die Plots der globalen Partitionsdichten in Abbildung 12 verrät uns, dass für die Wahl unserer Dendrogramm-Stufen Similarity-Werte unterhalb von 0.1 keine Rolle spielen. Das reduziert unsere globale Similarityliste SL_P beträchtlich, da fast 90% der Similarity-Werte kleiner als 0.1 betragen. Über die Hashmap sind wir in der Lage, die jeweiligen Similarity-Werte $Sim(C_1, C_2)$ zwischen zwei Clustern C_1 und C_2 zu bestimmen.

Bei der Realisierung der Schritte 3c) und 3d) aus Algorithmus 8 machen wir wiederum Gebrauch von der vorhandenen Sortierung in SL_P und arbeiten diese Liste von oben nach unten ab. Dadurch erübrigt sich die Suche nach der maximalen Similarity und das Berechnen von Similarity-Werten zwischen einem neuen Cluster mit seinen Nachbarn. Abbildung 14 gibt ein anschauliches Beispiel zu unserer Verschmelzungstrategie. Der Aufwand beträgt maximal $|P|^2 \cdot \overline{deg(p)}$, wobei $\overline{deg(p)}$ der durchschnittliche Knotengrad eines Knotens in P darstellt.

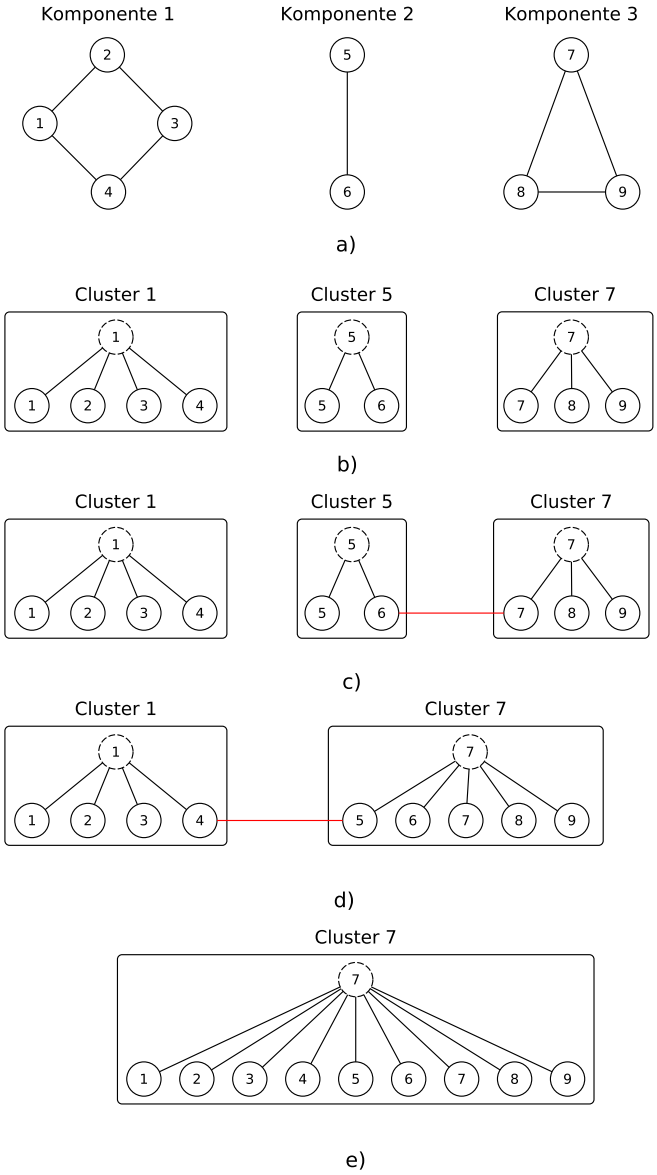


Abbildung 14: a) Seien die drei Komponenten Teil des Graphen $\tilde{G}$. b) Für jedes Cluster legen wir einen virtuellen Knoten mit der kleinsten ID, um das Cluster zu identifizieren. c-e) Unsere globale Similarity-Liste SL_P verrät uns, welche bislang noch nicht abgearbeiteten Knotenpaare die höchste Similarity hat und damit auch, welches Clusterpaar wir als nächstes zu einem neuen Cluster verschmelzen sollen. Dabei gilt das Prinzip: Das größere Cluster gewinnt.

Zusammenfassung und Ausblick

Wir haben moderne Methoden der Netzwerkanalyse zum Aufbau eines Empfehlungssystems angewandt, nämlich die einseitige Projektion nach Zweig und Kaufmann [Zweig:2011] und die Link-Community-Analyse nach Ahn et. al. [Ahn:2010].

Das in dieser Arbeit vorgestellte Verfahren (Verfahren 2) ist in der Lage, beispielsweise in Cluster Nr.43 alle Filme des bekannten Schauspielers John Wayne oder in Cluster Nr.59 alle James Bond Filme aufzuspüren, obwohl die Kenntnis über ihren inhaltlichen Zusammenhang aus den Filmtiteln nicht direkt ersichtlich ist (siehe Anhang).

Die grössten Herausforderungen werden durch die begrenzten Speicherkapazitäten und die begrenzte Rechenleistung gestellt. Die große Datenmenge macht eine sparsame Speicherverwaltung sowie die effiziente Implementierung der Algorithmen unentbehrlich.

Verbesserungsbedarf gibt es sicherlich in verschiedenen Bereichen:

1. Zur Berechnung der *coocc*-Werte für alle Knotenpaare aus P haben wir eine *coocc*-Matrix der Dimension $|P| \times |P|$ als statisches Array implementiert. Für $|P| = 17.770$ mag das noch möglich sein, aber die Matrix wächst quadratisch mit $|P|$. Auch eine Implementierung mit einer Hashmap könnte bald an seine Grenzen stoßen. Dann sind andere Algorithmen gefragt.

2. Weitere Datenbereinigungen in der Vorselektierung sind möglich: Beispielsweise könnten spezielle Kunden aussortiert werden, die eine überdurchschnittlich hohe Anzahl an Filmen gut bewertet haben. Dies würde zu einer weiteren Reduktion der Kantenmengen führen.

3. Die Anwendung des Link-Community-Algorithmus auf ungewichtete Graphen ist relativ einfach zu implementieren, effizient, hat einen geringen Speicherbedarf und ist deshalb besonders geeignet für große Netzwerke. Allerdings werden zu viele Kanten ausgefiltert, es entstehen zu viele isolierte Knoten, also Filme, für die es keine Empfehlung gibt. In diesen Fällen könnte man auf die lokalen Listen zurückgreifen. Diese geben gewisse Anhaltspunkte, welche weiteren Filme für eine Empfehlung in Betracht gezogen werden können.

   Besser wäre unter Umständen die Arbeit mit dem gewichteten OMP-Graphen. Alle Kanten blieben erhalten, doch die große Anzahl an Kantengewichten müssten mit abgespeichert werden, die Similarity-Werte wären komplizierter zu berechnen. Auch für die Partitionsdichte müsste man sich eine geeignete Definition überlegen.

Für die Bewertung der Qualität standen uns kaum Metadaten zur Verfügung. In der Regel sind aber solche Metadaten (Genre, Künstler, Kategorien, Serien, ...) vorhanden und die Anzahl der Daten auch viel größer. Dabei kann man auch einen Teil der bereits aus den Metadaten bekannten zusammengehörigen Daten (wie etwa Staffeln einer Serie) von vornherein zu einem Knoten zusammenfassen und den anderen Teil der Metadaten zur Qualitätsbewertung hinzuziehen.

Doch die beste Qualitätsbewertung ist das Feedback der Kunden, denn über den Geschmack lässt sich bekanntlich streiten.

Anhang

Ein kleiner Auschnitt aus der Menge der Cluster nach dem $s_{\max}$-basierten Verfahren

Cluster 1 (#Filme = 22, Φ = 348.85)
1993, Star Trek: The Next Generation: Season 7
1992, Star Trek: The Next Generation: Season 6
1995, Star Trek: Voyager: Season 1
1987, Star Trek: The Next Generation: Season 1
1993, Star Trek: Deep Space Nine: Season 1
1999, Star Trek: Voyager: Season 6
1997, Star Trek: Voyager: Season 4
2000, Star Trek: Voyager: Season 7
1998, Star Trek: Deep Space Nine: Season 7
1994, Star Trek: Deep Space Nine: Season 3
2001, Star Trek: Enterprise: Season 1
1996, Star Trek: Deep Space Nine: Season 5
1997, Star Trek: Deep Space Nine: Season 6
1991, Star Trek: The Next Generation: Season 5
1996, Star Trek: Voyager: Season 3
1998, Star Trek: Voyager: Season 5
1990, Star Trek: The Next Generation: Season 4
1989, Star Trek: The Next Generation: Season 3
1995, Star Trek: Deep Space Nine: Season 4
1988, Star Trek: The Next Generation: Season 2
1993, Star Trek: Deep Space Nine: Season 2
1995, Star Trek: Voyager: Season 2

Cluster 2 (#Filme = 12, Φ = 344.98)
1998, Buffy the Vampire Slayer: Season 3
2003, Angel: Season 5
2002, Angel: Season 4
1999, Buffy the Vampire Slayer: Season 4
1997, Buffy the Vampire Slayer: Season 2
2001, Buffy the Vampire Slayer: Season 6
1999, Angel: Season 1
1997, Buffy the Vampire Slayer: Season 1
2000, Buffy the Vampire Slayer: Season 5
2002, Buffy the Vampire Slayer: Season 7
2000, Angel: Season 2
2001, Angel: Season 3

Cluster 3 (#Filme = 44, Φ = 299.09)
1963, The Twilight Zone: Vol. 2
1960, The Twilight Zone: Vol. 41
1964, The Twilight Zone: Vol. 17
1963, The Twilight Zone: Vol. 30
1961, The Twilight Zone: Vol. 36
1960, The Twilight Zone: Vol. 20
1967, The Twilight Zone: Vol. 32
1963, The Twilight Zone: Vol. 5
1962, The Twilight Zone: Vol. 12
1968, The Twilight Zone: Vol. 33
1964, The Twilight Zone: Vol. 23
1960, The Twilight Zone: Vol. 18
1960, The Twilight Zone: Vol. 39
1962, The Twilight Zone: Vol. 42
2000, The Twilight Zone: Vol. 29
1961, The Twilight Zone: Vol. 9
1962, The Twilight Zone: Vol. 1
1963, The Twilight Zone: Vol. 15
1964, The Twilight Zone: Vol. 25
1960, The Twilight Zone: Vol. 34
1961, The Twilight Zone: Vol. 4
1963, The Twilight Zone: Vol. 21
1962, The Twilight Zone: Vol. 22
1962, The Twilight Zone: Vol. 40
1961, The Twilight Zone: Vol. 35
1963, The Twilight Zone: Vol. 11
1960, The Twilight Zone: Vol. 19
1961, The Twilight Zone: Vol. 10
1964, Treasures of the Twilight Zone
1964, The Twilight Zone: Vol. 24
1963, The Twilight Zone: Vol. 16
1964, The Twilight Zone: Vol. 28
1963, The Twilight Zone: Vol. 14
1964, The Twilight Zone: Vol. 27
1962, The Twilight Zone: Vol. 3
1964, More Treasures of the Twilight Zone
1961, The Twilight Zone: Vol. 7
1961, The Twilight Zone: Vol. 13
1962, The Twilight Zone: Vol. 38
1961, The Twilight Zone: Vol. 8
1964, The Twilight Zone: Vol. 26
1961, The Twilight Zone: Vol. 37
1962, The Twilight Zone: Vol. 6
1963, The Twilight Zone: Vol. 31

Cluster 4 (#Filme = 10, Φ = 284.68)
1995, Babylon 5: Season 3
1999, Babylon 5: A Call to Arms
1998, Babylon 5: Thirdspace
1998, Babylon 5: In the Beginning
1994, Babylon 5: Season 2
1998, Babylon 5: Season 5
1993, Babylon 5: The Gathering
1996, Babylon 5: Season 4
1998, Babylon 5: The River of Souls
1994, Babylon 5: Season 1

Cluster 5 (#Filme = 14, Φ = 217.9)
1995, Sharpe 8: Sharpe's Sword
1995, Sharpe 6: Sharpe's Gold
1996, Sharpe 11: Sharpe's Mission
1997, Sharpe 14: Sharpe's Waterloo
1994, Sharpe 4: Sharpe's Enemy
1993, Sharpe 2: Sharpe's Eagle
1997, Sharpe 12: Sharpe's Revenge
1996, Sharpe 10: Sharpe's Siege
1994, Sharpe 5: Sharpe's Honour
1994, Sharpe 3: Sharpe's Company
1996, Sharpe 9: Sharpe's Regiment
1995, Sharpe 7: Sharpe's Battle
1997, Sharpe 13: Sharpe's Justice
1993, Sharpe 1: Sharpe's Rifles

Cluster 6 (#Filme = 8, Φ = 209.36)
2003, Stargate SG-1: Season 7
1998, Stargate SG-1: Season 2
1999, Stargate SG-1: Season 3
1997, Stargate SG-1: Season 1
2000, Stargate SG-1: Season 4
2002, Stargate SG-1: Season 6
2001, Stargate SG-1: Season 5
2004, Stargate Atlantis: Rising (Pilot Episode)

Cluster 7 (#Filme = 6, Φ = 206.72)
1993, Red Dwarf: Series 6

1988, Red Dwarf: Series 2
1988, Red Dwarf: Series 1
1992, Red Dwarf: Series 5
1988, Red Dwarf: Series 3
1988, Red Dwarf: Series 4

Cluster 8 (#Filme = 8, Φ = 179.56)
2002, Farscape: Season 4
1998, Babylon 5: In the Beginning
1995, Babylon 5: Season 3
2004, Farscape: The Peacekeeper Wars
2001, Farscape: Season 3
1996, Babylon 5: Season 4
1994, Babylon 5: Season 1
1994, Babylon 5: Season 2

Cluster 9 (#Filme = 8, Φ = 174.57)
1972, MASH: Season 1
1973, MASH: Season 2
1976, MASH: Season 5
1979, MASH: Season 8
1977, MASH: Season 6
1974, MASH: Season 3
1975, MASH: Season 4
1978, MASH: Season 7

Cluster 10 (#Filme = 6, Φ = 158.45)
1987, Married... with Children: Season 1
1987, Married... with Children: The Most Outrageous Episodes: Vol. 2
1988, Married... with Children: Season 3
1987, Married... with Children: The Most Outrageous Episodes: Vol. 1
1987, Married... with Children: Season 2
1989, Married... with Children: Season 4

Cluster 11 (#Filme = 12, Φ = 156.71)
1996, Cadfael: A Morbid Taste for Bones
1994, Cadfael: Monk's Hood
1994, Cadfael: The Sanctuary Sparrow
1995, Cadfael: The Virgin in the Ice
1997, Cadfael: The Rose Rent
1994, Cadfael: One Corpse Too Many
1997, Cadfael: St. Peter's Fair
1998, Cadfael: The Pilgrim of Hate
1997, Cadfael: The Raven in the Foregate
1998, Cadfael: The Potter's Field
1996, Cadfael: The Devil's Novice
1998, Cadfael: The Holy Thief

Cluster 12 (#Filme = 7, Φ = 144.79)
1933, Duck Soup
1935, A Night at the Opera
1932, Horse Feathers
1929, The Cocoanuts
1931, Monkey Business
1930, Animal Crackers
1937, A Day at the Races

Cluster 13 (#Filme = 6, Φ = 139.58)
1998, Dawson's Creek: Season 2
2001, Dawson's Creek: Season 5
2000, Dawson's Creek: Season 4
2003, Dawson's Creek: Series Finale
1999, Dawson's Creek: Season 3
1998, Dawson's Creek: Season 1

Cluster 14 (#Filme = 9, Φ = 139.15)
1994, Highlander: Season 4
1994, Highlander: Season 3
1994, Highlander: Counterfeit
1993, Highlander: Season 2
1996, Highlander: Season 5
1992, Highlander: Season 1
1995, Highlander: Finale
1997, Highlander: Season 6
2004, Highlander: Unholy Alliance

Cluster 15 (#Filme = 13, Φ = 137.94)
1999, SpongeBob SquarePants: Tide and Seek
1999, SpongeBob SquarePants: Lost at Sea
2000, SpongeBob SquarePants: Season 2
2004, SpongeBob SquarePants: Spongeguard on Duty
1999, SpongeBob SquarePants: Season 1
2002, SpongeBob SquarePants: Halloween
1999, SpongeBob SquarePants: SpongeBob Goes Prehistoric
2002, SpongeBob SquarePants: Nautical Nonsense / Sponge Buddies
2002, SpongeBob SquarePants: Sea Stories
1999, SpongeBob SquarePants: The Seascape Capers
2004, SpongeBob SquarePants: Sponge for Hire
2002, SpongeBob SquarePants: Tales From the Deep
2005, SpongeBob SquarePants: Home Sweet Pineapple

Cluster 16 (#Filme = 10, Φ = 135.04)
2000, The X-Files: Season 1
2000, The X-Files: Season 2
1996, The X-Files: Season 4
1998, The X-Files: Fight the Future
1998, The X-Files: Season 6
1999, The X-Files: Season 7
2001, The X-Files: Season 9
2000, The X-Files: Season 3
2000, The X-Files: Season 8
1997, The X-Files: Season 5

Cluster 17 (#Filme = 18, Φ = 132.64)
1997, Friends: Season 4
1995, Friends: Season 2
1994, The Best of Friends: Vol. 3
2002, Friends: Season 9
1994, Friends: Season 1
1994, The Best of Friends: Vol. 4
2004, Friends: The Series Finale
1999, Friends: Season 6
1994, The Best of Friends: Vol. 1
1998, Friends: Season 5
1994, The Best of Friends: Season 1
1994, The Best of Friends: Season 2
1994, The Best of Friends: Vol. 2
1996, The Best of Friends: Season 3
1997, The Best of Friends: Season 4
1996, Friends: Season 3
1999, Friends: Season 7
2001, Friends: Season 8

Cluster 18 (#Filme = 7, Φ = 123.99)
1997, As Time Goes By: Series 6
1994, As Time Goes By: Series 3
1996, As Time Goes By: Series 5
1992, As Time Goes By: Series 1 and 2
1998, As Time Goes By: Series 7
1992, As Time Goes By: Series 4
2002, As Time Goes By: You Must Remember This

Cluster 19 (#Filme = 6, Φ = 123.41)
1996, Absolutely Fabulous: Absolutely Special
2003, Absolutely Fabulous: Series 5
1992, Absolutely Fabulous: Series 1
2001, Absolutely Fabulous: Series 4
1995, Absolutely Fabulous: Series 3
1994, Absolutely Fabulous: Series 2

Cluster 20 (#Filme = 8, Φ = 122.39)
1997, Fire Down Below
1990, Hard to Kill
1995, Under Siege 2: Dark Territory
1988, Above the Law
1991, Out for Justice
1996, The Glimmer Man
1990, Marked For Death
1994, On Deadly Ground

Cluster 21 (#Filme = 23, Φ = 121.98)
1985, Friday the 13th: Part 5: A New Beginning
1988, A Nightmare on Elm Street 4: The Dream Master
1986, Friday the 13th: Part 6: Jason Lives
1994, Wes Craven's New Nightmare
1995, Halloween 6: The Curse of Michael Myers
1985, A Nightmare on Elm Street 2: Freddy's Revenge
1991, Freddy's Dead: The Final Nightmare
1993, Jason Goes to Hell
1989, A Nightmare on Elm Street 5: The Dream Child
2002, Halloween: Resurrection
1989, Halloween 5: The Revenge of Michael Myers
1982, Friday the 13th: Part 3
1988, Child's Play
1988, Hellbound: Hellraiser II
1989, Friday the 13th: Part 8: Jason Takes Manhattan
1987, A Nightmare on Elm Street 3: Dream Warriors
1998, Halloween: H2O
1981, Friday the 13th: Part 2
1988, Halloween 4: The Return of Michael Myers
1984, Friday the 13th: Part 4: The Final Chapter
1988, Friday the 13th: Part 7: The New Blood
1990, Child's Play 2: Chucky's Back
1981, Halloween II

Cluster 22 (#Filme = 4, Φ = 119.55)
1998, Felicity: Season 1
1999, Felicity: Season 2
2000, Felicity: Season 3
2001, Felicity: Season 4

Cluster 23 (#Filme = 6, Φ = 116.53)
1988, Police Academy 5: Assignment - Miami Beach
1994, Police Academy 7: Mission to Moscow
1989, Police Academy 6: City Under Siege
1987, Police Academy 4: Citizens on Patrol
1985, Police Academy 2: Their First Assignment
1986, Police Academy 3: Back in Training

Cluster 24 (#Filme = 4, Φ = 113.42)
1998, Lain #2: Knights
1995, Neon Genesis Evangelion: Death and Rebirth
1999, Lain #3: Deus
1999, Lain #4: Reset

Cluster 25 (#Filme = 5, Φ = 113.38)
1991, 35 Up
1998, 42 Up
1985, 28 Up
1977, 21 Up
1964, Seven Up / 7 Plus Seven

Cluster 26 (#Filme = 45, Φ = 111.63)
2003, Dragon Ball Z: Garlic Jr.
2000, Dragon Ball Z: The History of Trunks
2002, Dragon Ball: Red Ribbon Army Saga
2005, Dragon Ball Z: Bio-Broly
1995, Dragon Ball: Piccolo Jr. Saga: Part 1
2002, Dragon Ball Z: The Return of Cooler
2003, Dragon Ball Z: Fusion
2002, Dragon Ball: Fortune Teller Baba Saga
2000, Dragon Ball Z: Androids
2003, Dragon Ball Z: Babidi
1986, Dragon Ball: King Piccolo Saga: Part 1
2001, Dragon Ball: Tournament Saga
1995, Dragon Ball: Piccolo Jr. Saga: Part 2
2002, Dragon Ball Z: Kid Buu Saga
1992, Dragon Ball Z: Super Android 13
2001, Dragon Ball: Mystical Adventure
1999, Dragon Ball Z: Great Saiyaman: Gohan's Secret
1993, Dragon Ball Z: Broly: The Legendary Super Saiyan
1998, Dragon Ball Z: The World's Strongest
1997, Dragon Ball Z: Dead Zone
1991, Dragon Ball Z: Lord Slug
1998, Dragon Ball Z: Vol. 17: Super Saiyan
1996, Dragon Ball Z: Broly's Second Coming
1997, Dragon Ball Z: Vol. 2: The Saiyans
2003, Dragon Ball GT: A Hero's Legacy
1999, Dragon Ball Z: Great Saiyaman: Final Round
2002, Dragon Ball: General Blue Saga
1996, Dragon Ball Z: Bojack Unbound
2003, Dragon Ball Z: Frieza
2003, Dragon Ball GT
2000, Dragon Ball Z: Trunks Saga
2003, Dragon Ball Z: Perfect Cell
2003, Dragon Ball: The Path to Power
2003, Dragon Ball Z: Majin Buu
1989, Dragon Ball Z: Imperfect Cell Saga
2002, Dragon Ball: Commander Red Saga
2000, Dragon Ball Z: Captain Ginyu Saga
2000, Dragon Ball: The Saga of Goku
1986, Dragon Ball: King Piccolo Saga: Part 2
2003, Dragon Ball Z: Cell Games
2000, Dragon Ball Z: Bardock: The Father of Goku
2002, Dragon Ball: Tien Shinhan Saga
2001, Dragon Ball Z: World Tournament
2002, Dragon Ball Z: Cooler's Revenge
2003, Dragon Ball GT: The Lost Episodes

Cluster 27 (#Filme = 4, Φ = 110.55)
2002, Coupling: Season 3
2001, Coupling: Season 2
2004, Coupling: Season 4
2000, Coupling: Season 1

Cluster 28 (#Filme = 9, Φ = 109.83)
1991, Law & Order: Season 2
2003, Law & Order: Criminal Intent: Season 3
2003, Law & Order: Season 14
2003, Law & Order: Special Victims Unit: The Fifth Year
1992, Law & Order: Season 3
2001, Law & Order: Criminal Intent: The First Year
1999, Law & Order: Special Victims Unit: The First Year
2000, Law & Order: Special Victims Unit: The Second Year
1990, Law & Order: Season 1

Cluster 29 (#Filme = 6, Φ = 108.17)

2003, Star Trek: Enterprise: Season 3
2002, Star Trek: Enterprise: Season 2
1999, Star Trek: Voyager: Season 6
2000, Star Trek: Voyager: Season 7
2001, Star Trek: Enterprise: Season 1
1998, Star Trek: Voyager: Season 5

Cluster 30 (#Filme = 10, Φ = 107.94)
2004, Prime Suspect 6
1995, Prime Suspect 4
2003, Foyle's War: Set 2
1995, House of Cards Trilogy III: The Final Cut
1991, Prime Suspect 1
1993, Prime Suspect 3
1990, House of Cards Trilogy I: House of Cards
1994, House of Cards Trilogy II: To Play the King
1996, Prime Suspect 5
1992, Prime Suspect 2

Cluster 31 (#Filme = 7, Φ = 105.14)
1994, Homicide: Life on the Street: Season 3
1998, Homicide: Life on the Street: Season 7
1994, Homicide: Life on the Street: Season 4
2000, Homicide: The Movie
1997, Homicide: Life on the Street: Season 6
1996, Homicide: Life on the Street: Season 5
1993, Homicide: Life on the Street: Seasons 1 & 2

Cluster 32 (#Filme = 4, Φ = 101.94)
2000, Outlaw Star: Vol. 2
2000, Outlaw Star: Vol. 3
1996, Armitage III: Poly-Matrix
2000, Outlaw Star: Vol. 1

Cluster 33 (#Filme = 15, Φ = 100.68)
1997, Hercules: The Legendary Journeys: Season 4
1998, Hercules: The Legendary Journeys: Season 5
1995, Hercules: Amazon Women/Lost Kingdom
1996, Hercules: The Legendary Journeys: Season 3
1999, Hercules: The Legendary Journeys: Season 6
1996, Xena: Warrior Princess: Season 2
1995, Xena: Warrior Princess: Season 3
1999, Xena: Warrior Princess: Season 5
1995, Hercules: The Legendary Journeys: Season 2
1995, Xena: Warrior Princess: Season 1
2000, Xena: Warrior Princess: Season 6
1995, Hercules: Warrior Princess / Gauntlet / Unchained Heart
1998, Xena: Warrior Princess: Season 4
1994, Hercules: The Legendary Journeys: Season 1
2001, Xena: Warrior Princess: Series Finale

Cluster 34 (#Filme = 12, Φ = 100.43)
1984, George Carlin: Carlin on Campus
1990, George Carlin: Personal Favorites
1996, George Carlin: Back in Town
1999, George Carlin: You are All Diseased
1996, George Carlin: George's Best Stuff
1977, George Carlin: On Location With George Carlin
2001, George Carlin: Complaints and Grievances
1982, George Carlin: Carlin at Carnegie
1978, George Carlin Again!
1988, George Carlin: What Am I Doing in New Jersey?
1986, George Carlin: Playing with Your Head
1992, George Carlin: Jammin' in New York / Doin' It Again

Cluster 35 (#Filme = 11, Φ = 100.07)
1996, Beautiful Thing
2004, Latter Days
2000, The Broken Hearts Club
1997, Love! Valour! Compassion!
1995, Jeffrey
1999, Get Real
2000, Big Eden
1987, Maurice
1999, Trick
1990, Longtime Companion
2001, All Over the Guy

Cluster 36 (#Filme = 7, Φ = 98.35)
2002, The Land Before Time IX: Journey to Big Water
1994, The Land Before Time II: The Great Valley Adventure
1997, The Land Before Time V: The Mysterious Island
2003, The Land Before Time X: The Great Longneck Migration
2001, The Land Before Time VIII: The Big Freeze
1995, The Land Before Time III: The Time of the Great Giving
2000, The Land Before Time VII: Stone of Cold Fire

Cluster 37 (#Filme = 8, Φ = 97.71)
1999, Trigun
2000, Samurai X
2002, Hellsing
1999, Samurai X: Trust and Betrayal, Director's Cut
2000, Outlaw Star: Vol. 1
1997, Rurouni Kenshin
2000, Outlaw Star: Vol. 3
2000, Outlaw Star: Vol. 2

Cluster 38 (#Filme = 5, Φ = 96.44)
1999, Lain #1: Navi
1998, Lain #2: Knights
1999, Lain #4: Reset
1999, Lain #3: Deus
2002, Millennium Actress

Cluster 39 (#Filme = 5, Φ = 96.38)
2001, Dragon Ball Z: World Tournament
2000, Dragon Ball Z: The History of Trunks
2003, Dragon Ball Z: Babidi
2003, Dragon Ball Z: Majin Buu
1999, Tenchi Forever: Tenchi Muyo in Love 2

Cluster 40 (#Filme = 27, Φ = 93.36)
1997, Mystery Science Theater 3000: Prince of Space
1993, Mystery Science Theater 3000: Mitchell
1992, Mystery Science Theater 3000: Hercules Unchained
1997, Mystery Science Theater 3000: Space Mutiny
1993, Mystery Science Theater 3000: Mr. B's Lost Shorts
1997, Mystery Science Theater 3000: Overdrawn at the Memory Bank
1993, Mystery Science Theater 3000: The Brain That Wouldn't Die
1992, Mystery Science Theater 3000: The Killer Shrews
1990, Mystery Science Theater 3000: The Hellcats
1993, Mystery Science Theater 3000: I Accuse My Parents
1993, Mystery Science Theater 3000: Gunslinger
1993, Mystery Science Theater 3000: Beginning of the End
1994, Mystery Science Theater 3000: Red Zone Cuba
1999, Mystery Science Theater 3000: The Girl in Gold

Boots
1992, Mystery Science Theater 3000: Attack of the Giant Leeches
1992, Mystery Science Theater 3000: Teenagers from Outer Space
1999, Mystery Science Theater 3000: Hamlet
1993, Mystery Science Theater 3000: Manos: Hands of Fate
1998, Mystery Science Theater 3000: The Touch of Satan
1997, Mystery Science Theater 3000: Timechasers
1993, Mystery Science Theater 3000: The Wild World of Batwoman
1993, Mystery Science Theater 3000: Eegah!
1999, Mystery Science Theater 3000: Boggy Creek II: And the Legend Continues
1999, Mystery Science Theater 3000: Merlin's Shop of Mystical Wonders
1991, Mystery Science Theater 3000: Santa Claus Conquers the Martians
1992, Mystery Science Theater 3000: Hercules Against the Moon Men
1989, Mystery Science Theater 3000: The Crawling Hand

Cluster 41 (#Filme = 13, Φ = 93.24)
2003, Little House on the Prairie: There's No Place Like Home
1976, Little House on the Prairie: Season 3
1977, Little House on the Prairie: Season 4
1981, Little House on the Prairie: Season 8
1974, Little House on the Prairie: Season 1
2003, Little House on the Prairie: The Pilot
2003, Little House on the Prairie: As Long as We Are Together
2003, Little House on the Prairie: I'll Be Waving as You Drive Away
1978, Little House on the Prairie: Season 5
2003, Little House on the Prairie: Journey in the Spring
1980, Little House on the Prairie: Season 7
1975, Little House on the Prairie: Season 2
1979, Little House on the Prairie: Season 6

Cluster 42 (#Filme = 22, Φ = 92.82)
1938, Three Stooges: Stooges at Work
1960, The Three Stooges: Stop, Look and Laugh
1951, The Three Stooges: Merry Mavericks
1949, The Three Stooges: All Time Favorites
1937, The Three Stooges: Dizzy Doctors
1965, The Three Stooges: The Outlaws Is Coming
1934, The Three Stooges: Goofs on the Loose
1935, The Three Stooges: Three Stooges in History
1936, The Three Stooges: Cops and Robbers
1947, The Three Stooges Double Feature
1940, The Three Stooges: Nutty but Nice
1943, The Three Stooges: Spook Louder
1938, The Three Stooges: Stooged and Confoosed
1938, The Three Stooges: Healthy, Wealthy and Dumb
1945, The Three Stooges: Curly Classics
1947, The Three Stooges: Sing a Song of Six Pants
1932, The Three Stooges: Greatest Hits & Rarities
1938, The Three Stooges: G.I. Stooge
1963, The Three Stooges Go Around the World in a Daze
1942, The Three Stooges: Three Smart Saps
1935, Three Stooges: Stooges and the Law
1941, The Three Stooges: All the World's a Stooge

Cluster 43 (#Filme = 24, Φ = 92.46)
1960, The Alamo
1963, Donovan's Reef
1976, The Shootist
1965, Sons of Katie Elder
1942, The Flying Tigers
1967, El Dorado
1970, Chisum
1972, The Cowboys
1975, Rooster Cogburn
1967, The War Wagon
1970, Rio Lobo
1949, She Wore a Yellow Ribbon
1968, Hellfighters
1959, Rio Bravo
1961, The Comancheros
1973, Cahill: U.S. Marshall
1969, The Undefeated
1960, North to Alaska
1963, McLintock! Collector's Edition
1962, Hatari!
1971, Big Jake
1944, The Fighting Seabees
1965, In Harm's Way
1959, The Horse Soldiers

Cluster 44 (#Filme = 4, Φ = 91.68)
1997, King of the Hill: Season 2
1997, King of the Hill: Season 1
1998, King of the Hill: Season 3
1999, King of the Hill: Season 4

Cluster 45 (#Filme = 4, Φ = 90.42)
1990, Jeeves and Wooster: Season 4
1991, Jeeves and Wooster: Season 2
1990, Jeeves and Wooster: Season 3
1990, Jeeves and Wooster: Season 1

Cluster 46 (#Filme = 6, Φ = 88.79)
1988, Friday the 13th: Part 7: The New Blood
1986, Friday the 13th: Part 6: Jason Lives
1993, Jason Goes to Hell
1989, Friday the 13th: Part 8: Jason Takes Manhattan
2002, Jason X
1985, Friday the 13th: Part 5: A New Beginning

Cluster 47 (#Filme = 5, Φ = 87.33)
1973, Upstairs, Downstairs: Season 3
1972, Upstairs, Downstairs: Season 2
1974, Upstairs, Downstairs: Season 4
1975, Upstairs, Downstairs: Season 5
1971, Upstairs, Downstairs: Season 1

Cluster 48 (#Filme = 4, Φ = 85.64)
2001, The Blue Planet: Seas of Life: Seasonal Seas - Coral Seas
2001, The Blue Planet: Seas of Life: Ocean World - Frozen Seas
2001, The Blue Planet: Seas of Life: Open Ocean - The Deep
2001, The Blue Planet: Seas of Life: Tidal Seas - Coasts

Cluster 49 (#Filme = 5, Φ = 83.4)
1986, Black Adder II
1989, Black Adder IV
1999, Black Adder Back & Forth
1987, Black Adder III
1983, Black Adder

Cluster 50 (#Filme = 6, Φ = 82.76)
1995, Candyman 2: Farewell to the Flesh
1990, Child's Play 2: Chucky's Back

1988, Friday the 13th: Part 7: The New Blood
1993, Jason Goes to Hell
1989, Friday the 13th: Part 8: Jason Takes Manhattan
1985, Friday the 13th: Part 5: A New Beginning

Cluster 51 (#Filme = 4, Φ = 82.5)
2002, Gilmore Girls: Season 3
2000, Gilmore Girls: Season 1
2003, Gilmore Girls: Season 4
2001, Gilmore Girls: Season 2

Cluster 52 (#Filme = 16, Φ = 80.91)
1997, South Park: Passion of the Jew
1993, The Simpsons: Season 5
2001, Family Guy: Vol. 2: Season 3
2004, Simpsons Gone Wild
1994, The Simpsons: Season 6
1991, The Simpsons: Season 3
1990, The Simpsons: Season 2
1998, South Park: Season 2
1999, Family Guy: Vol. 1: Seasons 1-2
1997, South Park: Season 1
1999, South Park: Season 3
1989, The Simpsons: Season 1
1990, The Simpsons: Treehouse of Horror
1992, The Simpsons: Season 4
2000, South Park: Season 4
2001, South Park: Season 5

Cluster 53 (#Filme = 4, Φ = 80.81)
2000, Gundam Wing
2000, Outlaw Star: Vol. 3
2000, Outlaw Star: Vol. 2
2000, Outlaw Star: Vol. 1

Cluster 54 (#Filme = 5, Φ = 79.97)
2000, Aqua Teen Hunger Force: Vol. 1
2003, Aqua Teen Hunger Force: Vol. 3
2000, Aqua Teen Hunger Force: Vol. 2
2001, Sealab 2021: Season 1
2002, Sealab 2021: Season 2

Cluster 55 (#Filme = 5, Φ = 79.13)
1993, Saved by the Bell: The College Years: Season 1
1989, Saved by the Bell: Season 2
1989, Saved by the Bell: Season 1
2005, Saved by the Bell: Season 5
1991, Saved by the Bell: Seasons 3 & 4

Cluster 56 (#Filme = 5, Φ = 79.11)
1999, Farscape: Season 1
2000, Farscape: Season 2
2004, Farscape: The Peacekeeper Wars
2001, Farscape: Season 3
2002, Farscape: Season 4

Cluster 57 (#Filme = 9, Φ = 78.86)
1967, El Dorado
1959, The Horse Soldiers
1962, Hatari!
1948, Red River
1970, Chisum
1960, North to Alaska
1959, Rio Bravo
1970, Rio Lobo
1949, She Wore a Yellow Ribbon

Cluster 58 (#Filme = 8, Φ = 76.56)
1983, Cujo
1987, A Nightmare on Elm Street 3: Dream Warriors
1981, Friday the 13th: Part 2
1981, Halloween II
1984, Children of the Corn
1985, A Nightmare on Elm Street 2: Freddy's Revenge
1988, Hellbound: Hellraiser II
1988, Child's Play

Cluster 59 (#Filme = 9, Φ = 75.75)
1963, From Russia With Love
1981, For Your Eyes Only
1973, Live and Let Die
1965, Thunderball
1987, The Living Daylights
1977, The Spy Who Loved Me
1962, Dr. No
1974, The Man with the Golden Gun
1967, You Only Live Twice

Cluster 60 (#Filme = 7, Φ = 73.68)
1978, Best of The Muppet Show: Peter Sellers / John Cleese / Dudley Moore
1978, Best of The Muppet Show: Harry Belafonte / Linda Ronstadt / John Denver
1977, Best of The Muppet Show: Steve Martin / Carol Burnett / Gilda Radner
1976, Best of The Muppet Show: Diana Ross / Brooke Shields / Rudolph Nuryev
1977, Best of The Muppet Show: George Burns / Bob Hope / Dom DeLuise
1978, Best of The Muppet Show: Mark Hamill / Paul Simon / Raquel Welch
1977, Best of The Muppet Show: Elton John / Julie Andrews / Gene Kelly

Cluster 61 (#Filme = 6, Φ = 73.64)
1953, I Love Lucy: Season 3
1977, Three's Company: Season 2
2001, The I Love Lucy 50th Anniversary Special
1952, I Love Lucy: Season 2
1954, I Love Lucy: Season 4
1955, I Love Lucy: Season 5

Cluster 62 (#Filme = 5, Φ = 73.11)
1997, Hercules: The Legendary Journeys: Season 4
2004, Highlander: Unholy Alliance
1997, Highlander: Season 6
1995, Highlander: Finale
1994, Highlander: Counterfeit

Cluster 63 (#Filme = 21, Φ = 72.62)
2003, Scooby-Doo and the Monster of Mexico
2000, Scooby-Doo's Original Mysteries
1972, Scooby-Doo Meets the Harlem Globetrotters
1998, Scooby-Doo on Zombie Island
2002, Scooby-Doo! Winter Wonder Dog
1989, Scooby-Doo and the Reluctant Werewolf
1972, Scooby-Doo Meets Batman
1988, Scooby-Doo and the Ghoul School

2002, What's New Scooby-Doo?
1999, Scooby-Doo and the Witch's Ghost
1987, Scooby-Doo Meets the Boo Brothers
2000, Scooby-Doo's Spookiest Tales
1969, Scooby-Doo Where Are You?: Seasons 1 & 2
2000, Scooby-Doo's Creepiest Capers
2001, Scooby-Doo and the Cyber Chase
2004, Scooby-Doo and the Loch Ness Monster
1979, Scooby-Doo Goes Hollywood
2005, Aloha Scooby-Doo!
2003, Scooby-Doo and the Legend of the Vampire
2000, Scooby-Doo and the Alien Invaders
1999, Scooby-Doo's Greatest Mysteries

Cluster 64 (#Filme = 4, Φ = 71.71)
1970, Beneath the Planet of the Apes
1975, Battle for the Planet of the Apes
1971, Escape from the Planet of the Apes
1973, Conquest of the Planet of the Apes

Cluster 65 (#Filme = 10, Φ = 70.09)
1972, Play it Again, Sam
1983, Zelig
1975, Love and Death
1971, Bananas
1973, Sleeper
1971, Everything You Always Wanted to Know About Sex But Were Afraid to Ask
1984, Broadway Danny Rose
1985, The Purple Rose of Cairo
1969, Take the Money and Run
1987, Radio Days

Cluster 66 (#Filme = 4, Φ = 70.08)
1998, Mr. Show: Season 4
1995, Mr. Show: Season 1
1996, Mr. Show: Season 2
1997, Mr. Show: Season 3

Cluster 67 (#Filme = 8, Φ = 69.87)
2002, Brown Sugar
1997, Love Jones
2000, Soul Food: Season 1
1999, The Wood
2001, Two Can Play That Game
1991, The Five Heartbeats
2001, The Brothers
1999, The Best Man

Cluster 68 (#Filme = 27, Φ = 69.6)
2001, Justice League: Justice on Trial
1996, Superman the Animated Series: A Little Piece of Home
1992, Adventures of Batman & Robin: Poison Ivy/The Penguin
1992, Adventures of Batman & Robin: The Joker/Fire & Ice
1996, Superman: Last Son of Krypton
2002, Justice League: Paradise Lost
1999, Batman Beyond: The Movie
1992, Batman: The Animated Series: Out of the Shadows
1992, Batman the Animated Series: Secrets of the Caped Crusader
1993, Batman the Animated Series: Vol. 3
2001, Justice League
1998, Batman & Mr. Freeze: Subzero
1992, Batman: The Animated Series: The Legend Begins
1998, The Batman Superman Movie
2005, Justice League Unlimited: Saving the World
1992, Batman the Animated Series: Vol. 1
1992, Batman the Animated Series: Vol. 2
2000, Batman Beyond: Return of the Joker
2004, Justice League: Starcrossed the Movie
1999, Batman Beyond: Tech Wars / Disappearing Inque
1966, Superman the Animated Series: Vol. 1
2004, Justice League Unlimited: Joining Forces
1999, Batman Beyond: School Dayz / Spellbound
1993, Batman: Mask of the Phantasm
1996, Daredevil vs. Spiderman
2004, Justice League: The Brave and the Bold
1992, Batman: The Animated Series: Tales of the Dark Knight

Cluster 69 (#Filme = 5, Φ = 68.14)
1996, The Pretender: Season 1
1999, Profiler: Season 4
1998, Profiler: Season 3
1997, Profiler: Season 2
1996, Profiler: Season 1

Cluster 70 (#Filme = 5, Φ = 65.3)
1941, Shadow of the Thin Man
1947, Song of the Thin Man
1939, Another Thin Man
1936, After the Thin Man
1945, The Thin Man Goes Home

Cluster 71 (#Filme = 4, Φ = 64.83)
2002, The Dead Zone
2002, The Dead Zone: Season 2
2004, The Dead Zone: Season 3
2002, The Dead Zone: Season 1

Cluster 72 (#Filme = 9, Φ = 64.64)
1990, Child's Play 2: Chucky's Back
1985, Friday the 13th: Part 5: A New Beginning
1988, Friday the 13th: Part 7: The New Blood
1993, Jason Goes to Hell
1989, Friday the 13th: Part 8: Jason Takes Manhattan
1982, Friday the 13th: Part 3
1989, Halloween 5: The Revenge of Michael Myers
2000, Hellraiser IV: Bloodline
1994, Wes Craven's New Nightmare

Cluster 73 (#Filme = 31, Φ = 64.05)
1982, Doctor Who: Earthshock (Ep. 122)
1977, Doctor Who: The Talons of Weng-Chiang
1984, Doctor Who: The Caves of Androzani
1989, Doctor Who: The Curse of Fenric
1982, Doctor Who: The Visitation
1984, Doctor Who: Resurrection of the Daleks
1989, Doctor Who: Ghost Light
1979, Doctor Who: The Armageddon Factor
1978, Doctor Who: The Ribos Operation
1978, Doctor Who: The Pirate Planet
1975, Doctor Who: Pyramids of Mars (Ep. 82)
1964, Doctor Who: The Dalek Invasion of Earth
1983, Doctor Who: The Five Doctors
1978, Doctor Who: The Power of Kroll
1975, Doctor Who: Lost in Time: The William Hartnell Years
1973, Doctor Who: Carnival of Monsters
1967, Doctor Who: The Tomb of the Cybermen
1975, Doctor Who: The Ark in Space
1976, Doctor Who: The Robots of Death

1964, Doctor Who: The Aztecs
1985, Doctor Who: The Two Doctors
1968, Doctor Who: Lost in Time: The Patrick Troughton Years
1970, Doctor Who: Spearhead from Space
1980, Doctor Who: The Leisure Hive
1972, Doctor Who: The Three Doctors
1973, Doctor Who: The Green Death
1987, Doctor Who: Remembrance of the Daleks
1978, Doctor Who: The Stones of Blood
1985, Doctor Who: Vengeance on Varos
1978, Doctor Who: The Androids of Tara
1966, Doctor Who: Seeds of Death

Cluster 74 (#Filme = 10, Φ = 63.51)
1951, Operation Pacific
1944, The Fighting Seabees
1967, The War Wagon
1973, Cahill: U.S. Marshall
1942, The Flying Tigers
1969, The Undefeated
1959, The Horse Soldiers
1970, Rio Lobo
1951, Flying Leathernecks
1961, The Comancheros

Cluster 75 (#Filme = 4, Φ = 63.1)
1999, The West Wing: Season 1
2002, The West Wing: Season 4
2001, The West Wing: Season 3
1999, The West Wing: Season 2

Cluster 76 (#Filme = 4, Φ = 63.08)
2004, Smallville: Season 4
2001, Smallville: Season 1
2002, Smallville: Season 2
2003, Smallville: Season 3

Cluster 77 (#Filme = 4, Φ = 63.01)
2001, Andromeda: Season 2
2000, Andromeda: Season 1
2003, Andromeda: Season 4
2002, Andromeda: Season 3

Cluster 78 (#Filme = 7, Φ = 62.12)
1999, Futurama: Vol. 1
2004, Simpsons Gone Wild
1999, Futurama: Vol. 4
1999, Futurama: Vol. 2
1994, The Simpsons: Season 6
1990, The Simpsons: Treehouse of Horror
1999, Futurama: Vol. 3

Cluster 79 (#Filme = 5, Φ = 59.4)
1977, Herbie Goes to Monte Carlo
1979, The Apple Dumpling Gang Rides Again
1974, Herbie Rides Again
1968, The Love Bug
1980, Herbie Goes Bananas

Cluster 80 (#Filme = 4, Φ = 59.19)
2003, Queer as Folk: Season 3
2001, Queer as Folk: Season 1
2002, Queer as Folk: Season 2
2004, Queer as Folk: Season 4

Cluster 81 (#Filme = 4, Φ = 59.1)
2002, I Can Do Bad All By Myself
2002, Madea's Family Reunion (Stage Play)
2003, Madea's Class Reunion
2004, Meet the Browns

Cluster 82 (#Filme = 10, Φ = 58.82)
1983, Cujo
1985, Silver Bullet
1990, Stephen King's It!
1983, Christine: Special Edition
1985, Fright Night
1982, Creepshow
1985, Stephen King's Cat's Eye
1979, The Amityville Horror
1984, Children of the Corn
1980, The Howling

Cluster 83 (#Filme = 5, Φ = 58.51)
1997, Rurouni Kenshin
2000, Outlaw Star: Vol. 3
2000, Outlaw Star: Vol. 2
2000, Outlaw Star: Vol. 1
2000, Gundam Wing: The Movie: Endless Waltz

Cluster 84 (#Filme = 21, Φ = 57.72)
2000, VeggieTales: Esther, the Girl Who Became Queen
2005, VeggieTales: Duke and the Great Pie War
1997, VeggieTales Classics: Larry-Boy & The Fib from Outer Space
1999, VeggieTales: Madame Blueberry
2003, VeggieTales: The Wonderful World of Auto-tainment
1995, VeggieTales Classics: Rack, Shack and Benny
2004, VeggieTales: Dave and the Giant Pickle
1997, VeggieTales Classics: God Wants Me to Forgive Them!?!
1997, VeggieTales Classics: Josh and the Big Wall!
2002, Jonah: A VeggieTales Movie
2003, VeggieTales: Bible Heroes: Stand Up! Stand Tall! Stand Strong!
2003, VeggieTales: The Ballad of Little Joe
2001, VeggieTales: The Ultimate Silly Song Countdown
2000, VeggieTales: King George and the Ducky
1997, VeggieTales: The Star of Christmas
2004, VeggieTales: An Easter Carol
2001, VeggieTales: Lyle the Kindly Viking
2004, VeggieTales Classics: Where's God When I'm Scared?
1996, VeggieTales: The Toy That Saved Christmas
2003, VeggieTales: Bible Heroes: Lions, Shepherds and Queens
1997, VeggieTales: A Snoodles Tale

Cluster 85 (#Filme = 4, Φ = 57.16)
2000, Will & Grace: Season 3
2001, Will & Grace: Season 4
1998, Will & Grace: Season 1
1998, Will & Grace: Season 2

Cluster 86 (#Filme = 16, Φ = 56.37)
1961, The Dick Van Dyke Show: Season 1
1963, The Andy Griffith Show: Vol 1: High Noon at Mayberry / The Loaded Goat
1964, The Dick Van Dyke Show: Season 4

1963, The Andy Griffith Show: Vol 2: The Big House / A Wife for Andy
1962, The Dick Van Dyke Show: Season 2
1962, The Andy Griffith Show: Season 3
1997, The Best of the Andy Griffith Show
1963, The Andy Griffith Show: Vol 4: The Mountain Wedding / Opie and the Spoiled Child
1963, Andy Griffith Show: Classic Favorites
1965, The Dick Van Dyke Show: Season 5
1961, The Andy Griffith Show: Season 2
1960, The Andy Griffith Show
1960, The Andy Griffith Show: Season 1
1963, The Dick Van Dyke Show: Season 3
1963, The Andy Griffith Show: Vol 3: Andy Discovers America / Andy's English Valet
1963, The Andy Griffith Show: Vol 5: Rafe Hollister Sings / Class Reunion

Cluster 87 (#Filme = 8, Φ = 56.32)
2003, The Inspector Lynley Mysteries: Playing for Ashes
2002, The Inspector Lynley Mysteries: Payment in Blood
2002, The Inspector Lynley Mysteries: Well-Schooled in Murder
2003, The Inspector Lynley Mysteries: In the Presence of the Enemy
2001, Second Sight: Series 2
2001, The Inspector Lynley Mysteries: A Great Deliverance
2003, The Inspector Lynley Mysteries: Deception on His Mind
1999, Second Sight: Series 1

Cluster 88 (#Filme = 8, Φ = 56.07)
1985, Transformers: Season 3: Part 2/Season 4
1985, G.I. Joe: Season 1: Part 2
1986, G.I. Joe: Season 2: Part 1
1985, Transformers: Season 2: Part I
1983, G.I. Joe: Season 1: Part 1
1984, Transformers: Season 1
1987, G.I. Joe: The Movie
1986, Transformers: Season 3: Part 1

Cluster 89 (#Filme = 4, Φ = 55.41)
1985, Transformers: Season 2: Part I
1985, Transformers: Season 3: Part 2/Season 4
1984, Transformers: Season 1
1986, Transformers: The Movie

Cluster 90 (#Filme = 4, Φ = 55.17)
1977, George Carlin: On Location With George Carlin
1990, George Carlin: Personal Favorites
1986, George Carlin: Playing with Your Head
1985, Richard Pryor: Live & Smokin'

Cluster 91 (#Filme = 8, Φ = 54.38)
1998, Jackie Chan's Who Am I
1991, Operation Condor
1997, Jackie Chan's First Strike
2002, The Accidental Spy
1999, Twin Dragons
1983, Jackie Chan's Project A
1997, Mr. Nice Guy
1992, Supercop

Cluster 92 (#Filme = 5, Φ = 53.81)
1966, The Avengers '66
1968, The Avengers '68
1963, The Avengers '63
1965, The Avengers '65
1967, The Avengers '67

Cluster 93 (#Filme = 6, Φ = 53.5)
1999, Cirque du Soleil: Quidam
2003, Cirque du Soleil: Varekai
1994, Cirque du Soleil: Saltimbanco
2000, Cirque du Soleil: Journey of Man: IMAX
2003, Cirque du Soleil: Alegria
2000, Cirque du Soleil: Dralion

Cluster 94 (#Filme = 4, Φ = 53.45)
1994, Eddie Izzard: Unrepeatable
1997, Eddie Izzard: Glorious
1996, Eddie Izzard: Definite Article
2000, Eddie Izzard: Circle

Cluster 95 (#Filme = 12, Φ = 53.03)
1982, Cheers: Season 1
1993, Frasier: Season 1
1984, Cheers: Season 3
1994, Frasier: Season 2
1983, Cheers: Season 2
1998, Frasier: Season 6
1985, Cheers: Season 4
2003, Frasier: The Final Season
1996, Frasier: Season 4
1995, Frasier: Season 3
1997, Frasier: Season 5
1986, Cheers: Season 5

Cluster 96 (#Filme = 4, Φ = 52.78)
2002, Inu-Yasha: The Movie 3: Swords of an Honorable Ruler
2002, Inu-Yasha: The Movie 2: The Castle Beyond the Looking Glass
2000, Inu-Yasha
2001, Inu-Yasha: The Movie: Affections Touching Across Time

Cluster 97 (#Filme = 5, Φ = 51.42)
1988, All Creatures Great and Small: Series 4
1983, All Creatures Great and Small: The Specials
1978, All Creatures Great and Small: Series 1
1979, All Creatures Great and Small: Series 2
1980, All Creatures Great and Small: Series 3

Cluster 98 (#Filme = 4, Φ = 51.27)
1999, Tenchi Forever: Tenchi Muyo in Love 2
1996, Tenchi the Movie: Tenchi Muyo! In Love
1997, Tenchi the Movie 2: Daughter of Darkness
2000, Tenchi Universe

Cluster 99 (#Filme = 4, Φ = 50.9)
1977, Sinbad and the Eye of the Tiger
1958, The 7th Voyage of Sinbad
1974, The Golden Voyage of Sinbad
1963, Jason and the Argonauts

Literatur

[Agrawal:1993] Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. Mining Association Rules between Sets of Items in Large Databases. *ACM SIGMOD Conference*, 1993.

[Ahn:2010] Yong-Yeol Ahn, James P. Bagrow, and Sune Lehmann. Link communities reveal multiscale complexity in networks. *nature*, 09182, 2010.

[Brin:1997] Sergey Brin, Rajeev Motwani, Jeffrey D, and Ullman Shalom Tsur. Dynamic Itemset Counting and Implication Rules for Market Basket Data. *ACM*, 0-89791-911-4/97/0005:255–264, 1997.

[Cobb:2003] G. W. Cobb and Y.P. Chen. An application of Markov chain Monte Carlo to community ecology. *American Mathematical Monthly*, 110:264–288, 2003.

[Diestel:2006] Reinhard Diestel. *Graphentheorie*. Springer, 2006.

[Ester:2000] Martin Ester and Jörg Sander. *Knowledge Discovery in Databases*. Springer, 2000.

[Fayyad:1996] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM*, 39, 1996.

[Fortunato:2010] Santo Fortunato. Community detection in graphs. *Complex Networks and Systems Lagrange Laboratory*, 2010.

[Friedman] Eric D. Friedman. Trove - High Performance Collections for Java. `http://trove.starlight-systems.com/home`. [Online; accessed 01-Sept-2011].

[Gionis:2007] Aristides Gionis, Heikki Mannila, Taneli Mielikäinen, and Panayiotis Tsaparas. Assessing Data Mining Results via Swap Randomization. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(3), December 2007.

[Greenhill:2008] Catherine Greenhill and Brendan D. McKay. Asymptotic enumeration of sparse nonnegative integer matrices with specified row and column sums. *Adv. Appl. Math.*, 41(4):459–481, 2008.

[Jaccard:1901] nach [Ahn:2010] zitiert: P. Jaccard. Etude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901.

[Johnson:1967] S. C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 2:241–254, 1967.

[Krengel:2005] Ulrich Krengel. *Einführung in die Wahrscheinlichkeitstheorie und Statistik*. Vieweg, 2005.

[Krueger:2009] Guido Krüger and Thomas Stark. *Handbuch der Java Programmierung*. Addison-Wesley, 2009.

[Liu:2002] Jun S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer, 2002.

[Matteucci] Matteo Matteucci. A tutorial on clustering algorithms. `http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/index.html`. [Online; accessed 01-Sept-2011].

[Milgram:1967] S. Milgram. The small world problem. *Psychology Today*, page 60–67, 1967.

[Piatetsky-Shapiro:1991] Gregory Piatetsky-Shapiro. *Knowledge discovery in databases. Discovery, analysis, and presentation of strong rules*. 1991.

[Raeder:2010] Troy Raeder and Nitesh V. Chawia. Market basket analysis with networks. *SOCNET*, DOI 10.1007/s13278-010-0003-7, 28.August 2010.

[Reinelt] Gerhard Reinelt. *Effiziente Algorithmen 1*. Universität Heidelberg.

[Serfozo:2009] Richard Serfozo. *Basics of Applied Stochastic Processes*. Springer, 2009.

[Tan:2004] Pang-Ning Tan, Vipin Kumar, and Jaideep Srivastava. Selecting the right objective measure for association analysis. *Information Systems*, 29:293–313, 2004.

[Ullenboom:2009] Christian Ullenboom. *Java ist auch eine Insel*. Galileo Press, 2009.

[Watts:1998] D. J. Watts and S. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[Zweig:2010] Katharina Anna Zweig. How to Forget the Second Side of the Story. *Proceedings of the second International Conference on Advances in Social Network Analysis and Mining (ASONAM'10)*, pages 200–207, 2010.

[Zweig:2011] Katharina Anna Zweig and Michael Kaufmann. A Systematic Approach to the One-Mode Projection of Bipartite Graphs. *Social Network Analysis and Mining 1*, 2011.

Eidesstattliche Erklärung

Hiermit versichere ich, dass meine Arbeit selbstständig unter Anleitung verfasst habe, dass ich keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, und dass ich alle Stellen, die dem Wortlaut oder dem Sinne nach anderen Werken entlehnt sind, durch die Angabe der Quellen als Entlehnungen kenntlich gemacht habe.

Datum Unterschrift