INSTITUT FÜR INFORMATIK
Datenbanken und Informationssysteme
Universitätsstr. 1 D–40225 Düsseldorf

HEINRICH HEINE
UNIVERSITÄT
DÜSSELDORF

Graph Mining in Sozialen Netzwerken

Benjamin Treeck

Masterarbeit

Beginn der Arbeit:	23. Mai 2007
Abgabe der Arbeit:	24. September 2007
Gutachter:	Prof. Dr. Stefan Conrad
	Prof. Dr. Martin Mauve

Erklärung

Hiermit versichere ich, dass ich diese Masterarbeit selbstständig verfasst habe. Ich habe dazu keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Düsseldorf, den 24. September 2007 Benjamin Treeck

Inhaltsverzeichnis

1 Motivation

Häufig gilt es Datensätze zu analysieren, in denen Objekte in irgendeiner Beziehung zueinander stehen. Eine Datenrepräsentation, die auch relationale Informationen bewahrt und in der Data Mining effektiv betrieben werden kann, ist ein Graph. Dieser besteht aus einer Menge von Knoten, wobei je zwei Knoten durch eine Kante verbunden sein können. Beispiele für relationale Daten, die eine Graphdarstellung begünstigen, sind das World Wide Web, Computernetzwerke, Nahrungsketten, Interaktionsnetzwerke von Proteinen oder soziale Netzwerke [CF06]. Wenn wir von sozialen Netzwerken sprechen, meinen wir Akteure mit ihren Beziehungen zueinander (z.B. Freundschaft, Vertrauen, Geschäftsbeziehungen, sexuelle Kontakte usw.). Während viele Netzwerke bis vor wenigen Jahren noch überschaubar waren, so werden besonders soziale Netzwerke heutzutage durch das Internet immer größer. Dienste wie MySpace oder YouTube verzeichnen Mitgliederzahlen von vielen Millionen. Mit zunehmender Größe der Graphen kommen auch neue Fragestellungen auf, während konventionelle Aspekte, z.B. die Konnektivität eines Netzwerks nach Entfernen eines Knotens, an Bedeutung verlieren [New03]. Dazu kommt, dass die Methoden der klassischen Graphanalyse meist ineffizient für große Graphen sind und daher neue (unter Umständen approximative) Methoden erforderlich sind. Beim Graph Mining wird also das Augenmerk hauptsächlich darauf gerichtet, effiziente Algorithmen für große Graphen zu entwickeln. Das Ziel dieser Arbeit ist es daher, mehrere soziale Netzwerke im World Wide Web anhand aktueller Graph-Mining-Methoden zu analysieren und zu ergründen, ob die Muster, die laut Literatur große Graphen aus der realen Welt gegenüber zufällig generierten Graphen derselben Größe auszeichnen, auch in sozialen Netzwerken zu finden sind.

Der Aufbau dieser Arbeit gliedert sich wie folgt: Kapitel 2 liefert eine kurze Einführung in das Thema Graph Mining. Es folgen einige Grundlagen der Graphentheorie, die wichtig für den weiteren Verlauf sind. Kapitel 4 widmet sich der Datenbeschaffung, diskutiert wichtige Aspekte beim Extrahieren von riesigen Datenmengen und stellt die sozialen Netzwerke vor, die den experimentellen Gegenstand dieser Arbeit bilden. Im Hauptteil werden dann bekannte Muster für Graphen aus der realen Welt vorgestellt und die sozialen Netzwerke auf diese hin untersucht. Dabei werden verschiedene Methoden diskutiert. Während wir in Kapitel 6 die ermittelten Größen denen zufälliger Graphen gegenüberstellen, untersuchen wir in Kapitel 7, wie sich die Netzwerke in das Small-World-Modell einordnen lassen. Nach einem Ausblick fasst Kapitel 9 die Ergebnisse zusammen und schließt diese Arbeit ab.

2 Einführung Graph Mining

Bei dem Prozess der „Knowledge Discovery in Databases“ (KDD) geht es darum, aus riesigen Datenmengen nützliches Wissen (Muster) zu extrahieren. Während der Begriff KDD auch die Bereinigung der Daten und Interpretation der Ergebnisse beinhaltet, so wird die eigentliche Analyse der Daten auch Data Mining genannt. Bisher wurde dabei hauptsächlich das Augenmerk darauf gelegt, Entitäten anhand ihrer Attribute zu charakterisieren. Oft stehen die Objekte jedoch in irgendeiner Beziehung zueinander. Es gilt daher eine Form der Datenrepräsentation zu finden, in der relationale Informationen vollständig erhalten bleiben und zudem ein effektives Mining möglich ist [CH07]. Graphen besitzen diese Eigenschaften. Beim Data Mining auf Graphen sprechen wir auch von *Graph Mining*. Die Entitäten mit ihren Attributen werden in einem Graphen als Knoten dargestellt, während ihre Beziehungen untereinander als Kanten zwischen den Knoten repräsentiert werden. Ein weiterer Vorteil von Graphen gegenüber relationalen Datenbanken ist, dass durch die Datenorganisation effizient über die Kanten traversiert werden kann, denn jede Entität wird direkt mit ihren Beziehungen zusammen gespeichert. Zuletzt können Graphen leicht in eine verständliche Form visualisiert werden, während relationale Datenbanken meist erst eine Konvertierung in Graphform vornehmen müssen. Im nachfolgenden Kapitel wollen wir uns einer formaleren Definition von Graphen und ihren Eigenschaften widmen.

3 Grundlagen der Graphentheorie

3.1 Definitionen

Ein *Graph* besteht aus einem Zweitupel $G = (V, E)$, wobei V eine endliche Menge von Knoten und E eine Menge von Kanten ist [SS04]. In einem *gerichteten Graphen* zeigt jede Kante $(i, j) \in E$ von Knoten i zu Knoten j. Ein *ungerichteter Graph* ist ein gerichteter Graph, in dem die Kanten in beide Richtungen zeigen, d.h. wenn (i, j) eine Kante ist, dann auch (j, i).

Nach [OW02] bezeichnet ein *Weg* eine Liste von aufeinander folgenden Knoten, die jeweils durch eine Kante verbunden sind. Ein *Kreis* hingegen ist ein Weg, in dem Start- und Endknoten identisch sind. Ein ungerichteter Graph heißt *zusammenhängend*, falls zwischen jedem möglichen Knotenpaar (v, w) ein ungerichteter Weg existiert, mit v als Startknoten und w als Endknoten. Ein gerichteter Graph heißt dann zusammenhängend, wenn der entsprechende ungerichtete Graph zusammenhängend ist. Dieser ergibt sich, indem sämtliche gerichtete Kanten durch ungerichtete ersetzt werden. Abbildung 1 zeigt ein Beispiel für einen gerichteten und einen ungerichteten zusammenhängenden Graphen.

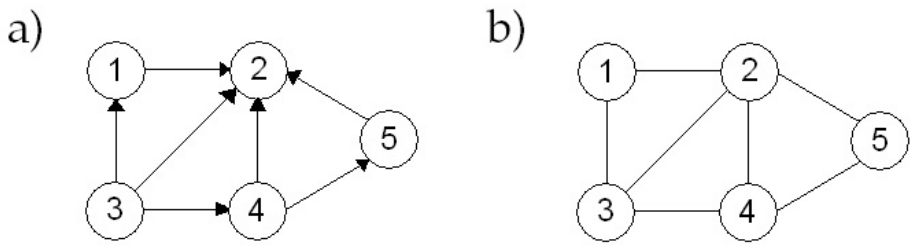

Abbildung 1: a) Beispiel für einen gerichteten Graphen. b) Beispiel für einen ungerichteten Graphen.

Im ungerichteten Fall ist ein Knoten *inzident* mit einer Kante, wenn er von dieser berührt wird. Zwei Knoten sind hingegen *adjazent* zueinander, wenn sie durch eine Kante verbunden sind. Ein Graph lässt sich durch seine *Adjazenzmatrix* repräsentieren. In dieser werden die Verbindungen als $(n \times n)$ -Matrix dargestellt, wobei n der Anzahl der Knoten $|V|$ entspricht. Das bedeutet, eine Kante von i nach j wird im entsprechenden Eintrag der Matrix mit einer 1 versehen, ansonsten ist der Eintrag 0. Für ungerichtete Graphen werden die Kanten als doppelt gerichtete Kanten aufgefasst, also sowohl von i nach j, als auch von j nach i. Die folgenden Adjazenzmatrizen stellen die Graphen G_1 aus Abbildung 1 a) und G_2 aus Abbildung 1 b) dar:

$$G_1 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}, \qquad G_2 = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

Für den gerichteten Fall definieren wir für einen Knoten u den *eingehenden Grad* $d_{in}(u)$ als die Anzahl der Kanten, die zu u zeigen. Analog ergibt sich der *ausgehende Grad* $d_{out}(u)$ aus der Summe der Kanten, die von u ausgehen. Im ungerichteten Fall sprechen wir lediglich vom *Grad* $d(u)$ und meinen damit die Anzahl Kanten, mit denen u inzident ist.

Im Folgenden sollen zwei Algorithmen vorgestellt werden, mit denen alle Knoten eines Graphen systematisch besucht werden können. Sie eignen sich zudem für die Suche von bestimmten Knoten im Graphen. Wir unterscheiden dabei zwischen dem Tiefendurchlauf und dem Breitendurchlauf.

3.2 Traversieren von Graphen

3.2.1 Tiefendurchlauf

Bei der Tiefensuche werden zunächst alle vom Startknoten aus direkt erreichbaren Knoten in einem Kellerspeicher (Stack) gespeichert [OW02]. Nach dem Last-in-First-Out-Prinzip (LIFO) wird nun immer der oberste Knoten aus dem Stack herausgenommen und auf seine Nachbarn hin betrachtet. Sollten diese noch nicht besucht worden sein, werden sie auf dem Stack abgelegt. Wenn der Stack leer ist, wurden alle Knoten des Graphen besucht und die Suche ist zu Ende. Nach diesem Vorgehen wird zunächst nur einem speziellen Pfad durch den Graphen gefolgt und daher erst in die Tiefe anstatt in die Breite gegangen [SS04]. Dies führt zu einer schmalen Abdeckung des Graphen und im Falle einer abgebrochenen Suche im schlechtesten Fall zu einer Kette von Knoten. Die Laufzeit und die Speicherplatzkomplexität der Tiefensuche betragen $O(|V| + |E|)$ [SS04], denn jeder Knoten und jede Kante werden einmal betrachtet. Da wir nur zusammenhängende Graphen betrachten, entspricht dies $O(|E|)$, da für diese gilt: $|E| \geq |V| - 1$.

3.2.2 Breitendurchlauf

Bei der Breitensuche wird nicht wie bei der Tiefensuche ein Stack, sondern eine Warteschlange für die noch zu besuchenden Knoten gepflegt [OW02]. Der Unterschied ist, dass der Zugriff jetzt nach dem First-in-First-Out-Prinzip (FIFO) erfolgt, d.h. wir entnehmen immer das erste Element der Warteschlange und fügen seine Nachbarn dem Ende der Warteschlange hinzu. So werden zunächst alle Nachfolger des Startknotens besucht und danach wird erst die nächsttiefere Ebene nach demselben Prinzip abgearbeitet. Im Gegensatz zur Tiefensuche wird der Graph hier zwar weit, aber dafür flach abgedeckt. Dies kann zur Folge haben, dass bei Abbruch der Suche die tiefen Knoten nicht erreicht werden können. Die Komplexität beträgt offensichtlich auch hier $O(|V| + |E|)$ bzw. $O(|E|)$.

4 Datenbeschaffung

4.1 Funktionsweise eines Webcrawlers

Um Graphen aus sozialen Netzwerken im World Wide Web extrahieren zu können, wurde eine Applikation entwickelt, die wie ein Webcrawler die entsprechenden Seiten automatisch absucht. Ein typischer Webcrawler (auch Spider, Robot oder Bot genannt) beginnt an üblicherweise mehreren Start-Webseiten, sammelt dort Informationen und setzt seine Suche auf den durch Hyperlinks verzweigten Seiten fort [Sto07]. Da für unsere Zwecke lediglich die Vernetzung der Mitglieder (wer ist mit wem befreundet) von

Interesse ist, wird nur Links dieser Art zu den entsprechenden Profilen nachgegangen. Von jedem Profil wird dabei die Mitglieds-ID zusammen mit der Information, welche IDs mit diesem Profil benachbart sind, gespeichert. Für den Graphen bedeutet das, die IDs der Profile bilden die Knoten des Graphen und die Verlinkung eines Profils zu einem anderen Profil bildet eine Kante zwischen den entsprechenden Knoten. Das Programm speichert die extrahierten Kanten dann auf tertiärem Speicher (vorzugsweise einer Datenbank) in Form einer Liste mit Einträgen der Form „Knoten 1 → Knoten 2“ ab, was einer Kante von Knoten 1 zu Knoten 2 entspricht.

Beim Crawling bzw. beim Durchlaufen aller Knoten eines Graphen wird im Allgemeinen zwischen zwei verschiedenen Strategien unterschieden: der Breitensuche und der Tiefensuche (siehe Abschnitt 3.2). Speziell die Tatsache, dass aufgrund der hohen Anzahl eventuell nicht alle Seiten in absehbarer Zeit erreicht werden können (und man den Server nicht überlasten möchte), macht die Wahl der richtigen Strategie zu einer wichtigen Entscheidung. Die Breitensuche erscheint für unsere Zwecke die bessere Wahl, da so schon früh (also bei einer geringen Anzahl extrahierter Knoten) bereits Aussagen über Muster im Graphen, z.B. Transitivät (der Freund meines Freundes ist auch mein Freund), gemacht werden können. Abbildung 2 veranschaulicht den Ablauf der Informationsbeschaffung im Sinne der Breitensuche. Bei sehr großen Graphen wird in der Praxis auch häufig eine so genannte „Best-First“-Suche verwendet, bei der die Seiten in Reihenfolge ihrer Relevanz aufgesucht werden [Sto07]. Eventuell wird dabei *nur* über die vielversprechenden Knoten (mit höherer Bewertung) expandiert um zu einer Schätzung des Ergebnisses zu gelangen. Da die Knoten in unserer Analyse jedoch als gleichgestellt betrachtet werden sollen, erscheint dieser Ansatz ungeeignet.

4.2 Besonderheiten bei der Implementierung

Bei einer sehr großen Anzahl von Knoten lassen sich die Listen der besuchten Knoten sowie die Warteschlange nicht mehr im Hauptspeicher unterbringen. Daher müssen solche temporären Informationen in tertiärem Speicher gehalten werden. Das bringt den Vorteil mit sich, dass man einen abgebrochenen Crawling-Durchlauf später wieder fortführen kann. Ein weiterer Aspekt ist die Effizienz beim Suchen von Knoten. Es gilt z.B. jedes Mal, wenn eine Kante gefunden wurde, zu überprüfen, ob der Knoten, zu dem die Kante führt, schon zuvor besucht wurde. Wenn dies nämlich der Fall ist, brauchen wir ihn nicht in die Warteschlange der zu bearbeitenden Knoten aufnehmen. Diese Suche nach Duplikaten umfasst die Suche in der Warteschlange sowie die Liste der bereits besuchten Knoten. Bei sequenziellem Vorgehen $(O(|V|))$ erscheint die Suche ineffizient und kann bei 10^5 und mehr Knoten zu erheblichen Performanceeinbüßungen führen. Andere Möglichkeiten sind daher, Hashverfahren oder Tries zu verwen-

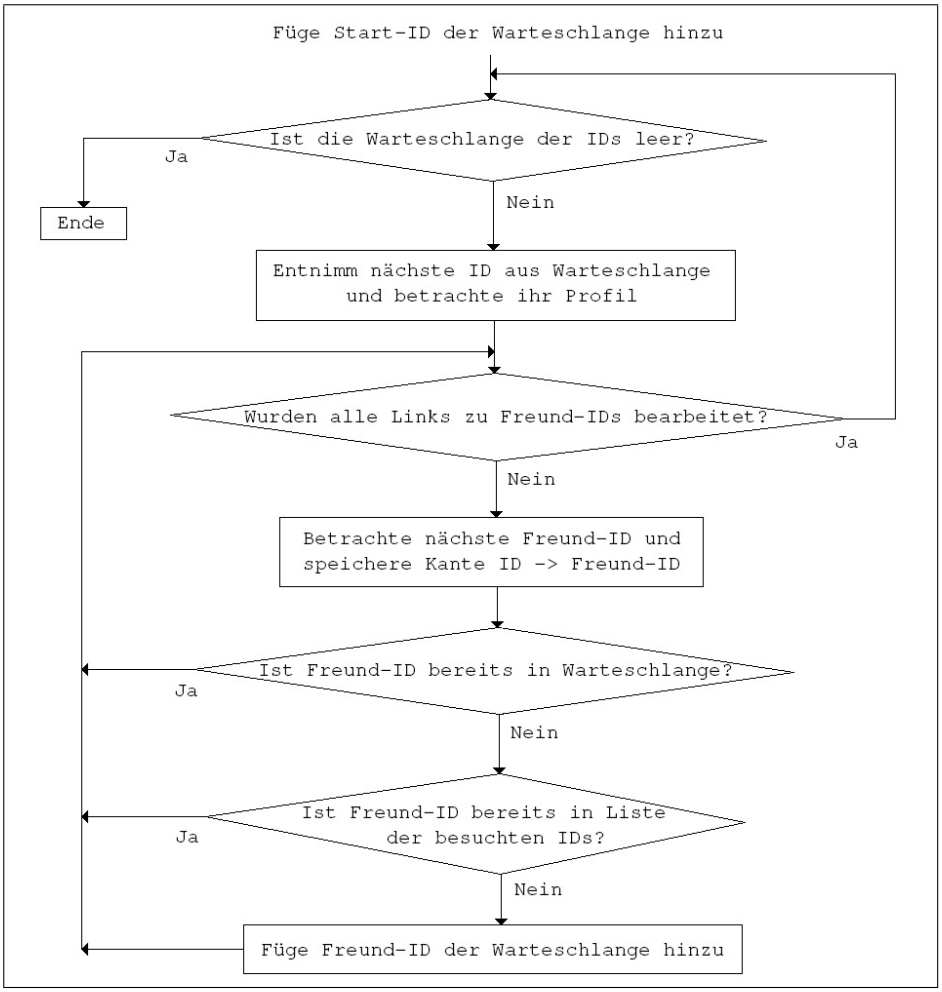

Abbildung 2: Algorithmus zur Extraktion der Freundesstruktur sozialer Netzwerke im World Wide Web.

den. Diese Formen der Datenspeicherung sowie ihre Anwendung in der Implementierung sollen im Folgenden skizziert werden.

4.2.1 Tries

Tries sind spezielle Suchbäume, in denen Zeichenketten effizient gespeichert werden können [Sto07]. Abbildung 3 zeigt dazu ein Beispiel. Jeder innere Knoten eines Tries enthält maximal $|\Sigma|$ viele Einträge, wobei Σ das Alphabet ist, aus dem die Schlüssel (also die Knoten-IDs) sich zusammensetzen. Jeder Eintrag verweist zudem auf einen Zielknoten. Die entstehende Verkettung von Symbolen bilden die Präfixe der Schlüssel. Sobald ein Präfix für einen Schlüssel eindeutig ist, wird auf das Blatt gezeigt, welches

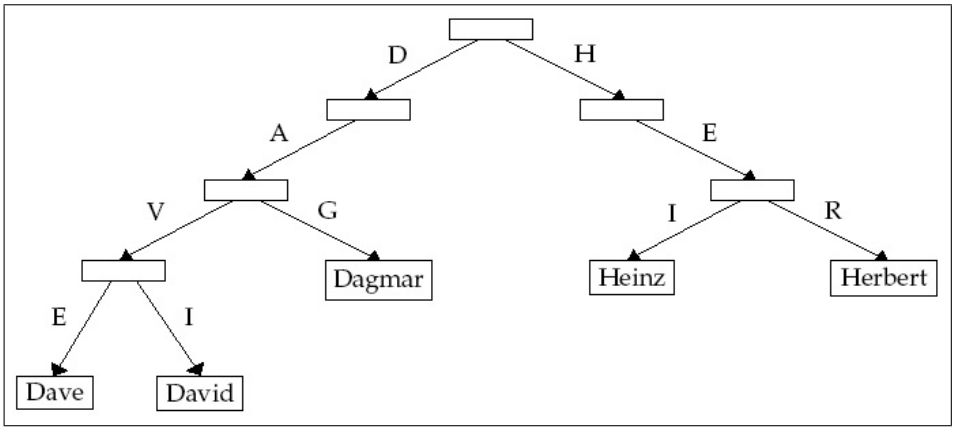

Abbildung 3: Beispiel eines Tries.

den Schlüssel enthält. Möchte man also einen Schlüssel im Trie suchen, so beginnt man an der Wurzel, sucht dort das Anfangssymbol des Schlüssels, folgt der entsprechenden Verzweigung und fährt so für die restlichen Symbole des Schlüssels fort. Erreicht man dabei das entsprechende Blatt, so ist der Schlüssel im Trie enthalten. Da hierbei im schlechtesten Fall in jedem betrachteten Knoten $|\Sigma|$ viele Einträge durchgegangen werden müssen und die Anzahl der zu besuchenden Knoten höchstens der Schlüssellänge entspricht, beträgt der Suchaufwand $O(m \cdot |\Sigma|)$ bei fester Schlüssellänge m. Der große Vorteil ist dabei, dass in Abhängigkeit der Anzahl gespeicherter Schlüssel der Suchaufwand konstant bleibt. Ein Nachteil der Tries zeigt sich jedoch bei einer großen Anzahl von Schlüsseln, denn dann wird die Knotenanzahl des zugehörigen Baums beträchtlich hoch (mindestens schon einmal genausoviele Blätter, zuzüglich der inneren Knoten). Gerade dann, wenn die Tries auf der Festplatte gespeichert sind, führt das Hin- und Herspringen des Lesekopfes zu erheblichem Geschwindigkeitsverlust. Die Tries haben sich daher in dieser Form für unsere Zwecke als unpraktikabel erwiesen. Eigene experimentelle Untersuchungen haben ergeben, dass sich eine Mischform aus Tries und sequenzieller Speicherung geeigneter zeigt. Bei dieser werden die Knoten-IDs bei Einfügen anhand ihres Präfixes sortiert und beim Suchen dann sequenziell unter allen IDs mit demselben Präfix nachgeschaut. Die Präfixlänge wird dabei so gewählt, dass die Schritte beim Suchen des Präfixes zuzüglich der Suche der ID unter den sortierten IDs möglichst gering gehalten werden. Unter der Annahme, dass jedes Präfix gleich häufig unter den IDs vorkommt, gilt es also $|\Sigma|^l + \frac{n}{|\Sigma|^l}$ bei n Schlüsseln mit Präfixlänge l zu minimieren. Wenn wir $n = 10^6$ annehmen für $|\Sigma| = 40$ (Buchstaben zuzüglich Zahlen und Sonderzeichen), wird folglich nach den ersten zwei Buchstaben sortiert. Für $|\Sigma| = 10$ (nur Zahlen) ist die Präfixlänge 3. Diese Methode ändert aller-

dings nichts an der Worst-Case-Laufzeit von $O(n)$, da im schlimmsten Fall alle IDs das gleiche Präfix haben.

4.2.2 Hashverfahren

Während die oben beschriebenen Tries für die ID-Suche beim Crawling zum Einsatz kommen, so werden für kurzzeitigere Berechnungen die Graphen anhand eines Hashverfahrens im Hauptspeicher gehalten. Das bedeutet, die Zeichenketten der Knoten-IDs werden über eine Hashfunktion auf einen Feldeintrag mit direktem Zugriff abgebildet [SS04]. Dabei sollte die Hashfunktion die Eigenschaft haben, dass sie die Werte möglichst breit auf das zur Verfügung stehende Feld streut, um Kollisionen zu vermeiden. Haben zwei oder mehr IDs trotzdem denselben Hashwert, so werden diese im entsprechenden Feldeintrag als verkettete Liste abgespeichert. Im schlimmsten Fall entartet die Speicherung aller IDs des Graphen somit zu einer linearen Verkettung. Für die Implementierung wurde auf die folgende Hashfunktion zurückgegriffen:

$$h(s) = \left(\sum_{i=0}^{b-1} s(i) \cdot 31^{b-i-1} \right) \mod N,$$

wobei $s(i)$ den ASCII-Wert des Symbols an Position i in der Knoten-ID darstellt, b die Länge der ID und N die Größe des Feldes. Möchte man nun einen Knoten im Graphen suchen, so berechnet man wieder die Hashfunktion der ID und findet den Knoten an der entsprechenden Stelle im Feld. Des Weiteren lässt sich dort eine verkette Liste von Knoten-IDs vorfinden, welche die Kanten darstellen, die von dem entsprechenden Knoten ausgehen.

Hinsichtlich des Aufwandes beim Hashen kann auch hier das Suchen eines Elements zur Suche in einer sequenziellen Liste entarten, nämlich dann, wenn alle IDs denselben Hashwert aufweisen. Auf der anderen Seite wäre der Aufwand konstant, wenn überhaupt keine Kollision auftreten würde. Er hängt daher stark von der verwendeten Hashfunktion ab.

4.3 Bereinigung des Datensatzes

Um nach Extraktionsende einen konsistenten Graphen konstruieren zu können, müssen nun all diejenigen Kanten wieder entfernt werden, die zu ungültigen Knoten führen. Ungültige Knoten entsprechen dabei Profilen, die privat oder aus sonstigen Gründen nicht einsehbar sind sowie solchen, die im Falle einer unvollständigen Netzextraktion nicht mehr besucht werden konnten und sich noch in der Warteschlange befanden. Wenn der Graph daher lediglich auf Basis der Kantenliste aufgebaut werden soll, würden ohne diese Bereinigung solche Profile im Graphen fälschlicherweise als Knoten mit ausgehendem Grad 0 repräsentiert werden und sich

somit nicht von Profilen unterscheiden, die wirklich keine ausgehenden Links haben. Wenn daher die vom Programm extrahierte Kantenliste des Graphen einfach um solche Kanten bereinigt werden würde, deren Knoten-ID auf der rechten Seite niemals auf der linken Seite vorkommt, so würden auch alle Kanten zu gültigen Knoten mit Grad 0 entfernt werden. Es ist daher unverzichtbar, bei der Extraktion eine Unterscheidung zwischen gültigen und ungültigen Knoten vorzunehmen, also zwischen Profilen, deren Links wir einsehen können und solchen, deren Links uns verborgen bleiben. Um dies bewerkstelligen zu können, genügt das Mitführen einer Liste von lediglich einer Sorte Knoten (gültig oder ungültig). Nach der Extraktion können dann alle Links zu ungültigen Knoten entfernt werden.

4.4 Vorstellung der extrahierten Netzwerke

Anhand des implementierten Webcrawlers wurden nun die Graphen von vier sozialen Online-Netzwerken verschiedener Größenordnungen extrahiert. Tabelle 1 gibt eine Übersicht über die Graphen und informiert über wesentliche Eigenschaften wie die Anzahl aller gefundenen Knoten und die Anzahl Knoten nach Bereinigung des Datensatzes (siehe Abschnitt 4.3). Betrachtet man die Anzahl Knoten, so fällt auf, dass Netzwerke A und

Netzwerk	n	n_v	m	m_v	z	n_q
A	24,142	24,142	223,905	223,905	9.27	0
B	45,854	41,005	206,259	180,558	4.40	0
C	48,602	48,602	304,488	304,488	6.27	0
D	219,338	200,000	3,427,127	1,756,492	8.78	720,900

Tabelle 1: Eigenschaften der extrahierten Graphen: Anzahl Knoten n, Anzahl gültiger Knoten n_v, Anzahl Kanten m, Anzahl gültiger Kanten m_v, durchschnittlicher Knotengrad z und Anzahl verbleibender Knoten in der Warteschlange bei Extraktionsende n_q.

B ungefähr in derselben Größenordnung liegen. Dies ermöglicht es uns auch Vergleiche zwischen gleich großen Netzwerken anzustellen. Für alle betrachteten Netzwerke gilt, dass die Kanten zwischen den Knoten eine Freundschaftsbeziehung darstellt. Die Funktion, Freundschaften zu schließen, ist jedoch in allen Netzwerken optional, so dass wir eigentlich immer nur einen Ausschnitt des Netzwerks betrachten. Im Folgenden sollen die Netzwerke kurz charakterisiert werden:

- **Soziales Netzwerk A:** Die Mitglieder können hier Videoclips zur Verfügung stellen oder sich selbst via Webcam zur Schau stellen. Vorteil-

haft für unsere Zwecke ist, dass es keine privaten Profile gibt und daher keine ungültigen Knoten.

- **Soziales Netzwerk B:** Dieses Netzwerk gibt Reisenden die Möglichkeit, kostenlose Unterkünfte bei anderen Mitgliedern zu finden. Laut den Betreibern sind dort über 200,000 Menschen angemeldet und über 44,000 davon weisen Freundschaftsbeziehungen auf, was mit der gemessenen Größe von 45,854 konform ist.

- **Soziales Netzwerk C:** Eine Möglichkeit für Leute, ihre Ideale nach außen zu tragen und mit anderen zu teilen. Kommunikation unter den Mitgliedern sollte demnach sehr stark vorhanden sein. Diese und die Tatsache, dass es keine privaten Profile gibt, begünstigen das Messergebnis. Laut Homepage der Betreiber gab es knapp 55,000 Mitglieder zur Zeit der Erhebung, wovon der Großteil erfasst werden konnte.

- **Soziales Netzwerk D:** Ein bekannter Video-Stream-Service, bei dem Mitglieder Videos hochladen können, welche wiederum von allen betrachtet werden können. Es ist das einzige Netzwerk, welches aufgrund seiner Größe nicht vollständig extrahiert werden konnte. Stattdessen beschränken wir uns für die Analyse auf die ersten 200,000 gültigen Knoten.

5 Mustersuche

Wie lassen sich große Netzwerke charakterisieren? Welche Muster zeichnen sie gegenüber zufälligen Graphen aus? In der Literatur wurden diese Fragen schon an verschiedenen Stellen aufgegriffen. Zum einen wird behauptet, dass in vielen komplexen Netzwerken der realen Welt eine so genannte „Power-Law"-Verteilung zu beobachten ist [CF06]. Betrachtet man die Graphen, so ist es gerade der Knotengrad, der einer solchen Verteilung unterliegt. Zum anderen haben viele komplexe Netzwerke, und im Speziellen soziale Netzwerke, geringe Durchmesser sowie einen hohen Grad des Clusterings [New03]. Im Folgenden wollen wir die genannten Muster näher erläutern sowie diverse Methoden zur Bestimmung dieser vorstellen. Schließlich sollen die beschriebenen Verfahren praktisch auf die im vorangegangenen Kapitel vorgestellten sozialen Netzwerke angewendet und die Ergebnisse diskutiert werden.

5.1 Power-Law-Verteilung

5.1.1 Definition

Häufig lassen sich in der Realität Verteilungen finden, bei denen Ereignisse mit einem viel größeren Wert als dem Erwartungswert wahrscheinlicher auftreten als in der Normalverteilung [CF06]. Ein Beispiel hierfür sind die Router im Internet: Viele sind an wenige Leitungen angeschlossen (wahrscheinlich die Router bei den Kunden), während nur wenige eine hohe Konnektivität aufweisen (Router des Backbone-Netzwerks). Auch die Anzahl der Bücherverkäufe weist eine solche Verteilung auf [New04]: Während viele Bücher sich nur spärlich verkaufen, so gibt es einige wenige mit hohen Verkaufszahlen. Dabei handelt es sich offensichtlich um die oberen Ränge der Bestsellerliste. Dies führt wiederum dazu, dass der Bekanntheitsgrad noch weiter steigt und noch viele weitere Käufe dazukommen. Solch eine Beziehung wird durch ein „Power-Law“ (im Deutschen auch Potenzgesetz genannt) beschrieben. Zwei Variablen x und y stehen in einer Power-Law-Relation zueinander, wenn gilt:

$$y = c \cdot x^{-\gamma},$$

wobei c und γ positive Konstanten sind [CF06]. Wir sprechen hingegen genau dann von einer *Power-Law-Verteilung*, wenn für eine Zufallsvariable X die Dichtefunktion

$$p(x) = c \cdot x^{-\gamma}$$

für $\gamma > 1$ und $x \geq x_{min}$ gegeben ist, wobei x_{min} der Datenpunkt ist, ab dem die Power-Law-Beziehung zu beobachten ist. Wir sehen an der Gleichung also, dass die Wahrscheinlichkeit mit $x \to \infty$ nicht exponentiell wie bei der Normalverteilung, sondern polynomiell abnimmt [CF06]. Abbildung 4 a) zeigt die Kurve der Dichtefunktion für verschiedene Werte des Exponenten γ. Dieser wird häufig auch als Power-Law-Exponent bezeichnet. Die Bedingung $\gamma > 1$ ist deswegen gegeben, damit die Dichtefunktion normalisiert werden kann, also damit die Wahrscheinlichkeiten der Verteilung zusammen 1 ergeben [New04]:

$$1 = \int_{x_{min}}^{\infty} p(x)\,\mathrm{d}x = c \cdot \int_{x_{min}}^{\infty} x^{-\gamma}\,\mathrm{d}x = \frac{c}{1-\gamma}\left[x^{-\gamma+1}\right]_{x_{min}}^{\infty}.$$

Es lässt sich leicht einsehen, dass der rechte Term der Gleichung bei $\gamma < 1$ gegen unendlich strebt und für $\gamma = 1$ nicht definiert ist. Er kann somit nicht 1 ergeben. Durch die Normalisierung wird zudem der Wert des Faktors c festgelegt. Dazu löst man die Gleichung nach c auf und erhält:

$$c = (\gamma - 1) \cdot x_{min}^{\gamma-1}.$$

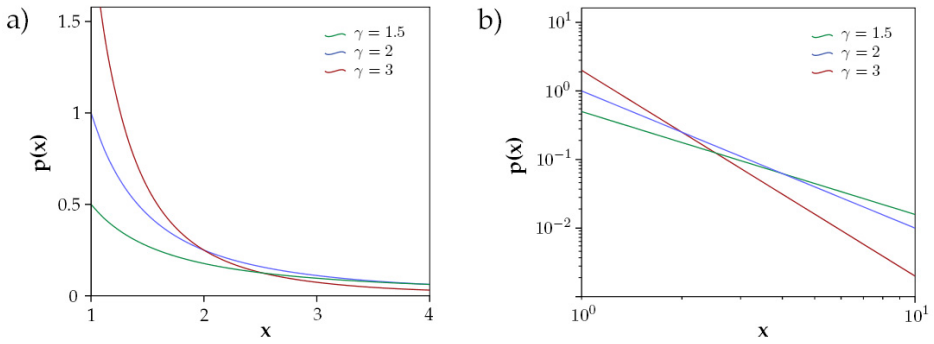

Abbildung 4: a) Dichtefunktion einer Power-Law-verteilten Zufallsvariable X für $x_{min} = 1$ und $\gamma = 1.5, 2$ und 3. b) Dieselbe Funktion auf logarithmischen Skalen zur Basis 10.

Wenn man dies nun für c in die Dichtefunktion einsetzt, so ergibt dies das Power-Law in normalisierter Form:

$$p(x) = \frac{\gamma - 1}{x_{min}} \cdot \left(\frac{x}{x_{min}} \right)^{-\gamma}.$$

Die Größe von c ist für die Form der Verteilungskurve jedoch irrelevant. Das bedeutet, egal welche Skalen wir betrachten, der funktionelle Zusammenhang ändert sich nicht. Wenn z.B. Dateien der Größe 2kB 0.25 mal so häufig wie Dateien der Größe 1kB vorkommen, so ist dies äquivalent dazu, dass 2MB-große Dateien 0.25 mal so häufig vorkommen wie Dateien der Größe 1MB [New04]. Aus diesem Grund nennt man diese Verteilung auch skaleninvariant. Weitere Beispiele für Power-Laws aus der Realität sind die Häufigkeit von Wörtern der englischen Sprache, das jährliche Einkommen der Menschen oder die Größe von Erdbeben, Mondkratern sowie Kriegen. Eine ausführlichere Liste ist in [New04] zu finden.

Beim Graph Mining werden üblicherweise die beiden Variablen Anzahl Knoten mit Grad k und k auf eine Power-Law-Beziehung hin untersucht. Da wir mit gerichteten Graphen arbeiten, wollen wir uns hier auf ausgehende Knotengrade beschränken. Die Fragestellung lautet dann also: Wie groß ist die Wahrscheinlichkeit für einen Knoten, k ausgehende Kanten zu haben? Im Fall der sozialen Netzwerke lässt sich die Frage umformulieren in: Wie wahrscheinlich ist es, dass ein Akteur in einer Beziehung zu k anderen Akteuren steht? Die Anzahl der Bekanntschaften k pro Individuum ist also die Größe, die es auf eine Power-Law-Verteilung hin zu untersuchen gilt.

5.1.2 Bestimmung des Power-Law-Exponenten

Möchte man eine Power-Law-Verteilung nachweisen, so muss der Power-Law-Exponent γ bestimmt werden, während c aufgrund der Skaleninvarianz (siehe Abschnitt 5.1.1) außer Acht gelassen werden kann. γ gibt Auskunft darüber, wie stark die Wahrscheinlichkeit für den zunehmenden Wert der zu untersuchenden Größe abnimmt. Klassische Ansätze basieren dabei auf graphischen Methoden. Ein alternativer Ansatz beruht auf dem Maximum-Likelihood-Prinzip, bei dem das Maximum der Wahrscheinlichkeit gesucht wird, dass die betrachteten Punkte aus einer Power-Law-Verteilung stammen.

5.1.2.1 Graphische Methoden Eine Power-Law-Relation hat die Eigenschaft, dass sie eine Gerade in einem Log-Log-Plot darstellt, also einem Koordinatensystem, in dem beide Skalen logarithmisch sind (siehe Abbildung 4 b) [Ada00]. Dies ist leicht einzusehen, wenn wir beide Seiten der Gleichung logarithmieren:

$$y = c \cdot x^{-\gamma}$$

$$\log(y) = \log(c) - \gamma \cdot \log(x).$$

Der Power-Law-Exponent γ entspricht also der Steigung der Geraden im Log-Log-Plot. Um diese zu bestimmen, bietet es sich an, eine lineare Regression auf dem Datensatz durchzuführen. Dabei wird ein linearer Zusammenhang zwischen den Größen angenommen und eine Gerade so konstruiert, dass sie die gemessenen Punkte so gut wie möglich abbildet. Meistens wird zur Bestimmung der Geradengleichung die Methode der kleinsten Quadrate verwendet: Die Summe der quadrierten Abstände der Punkte zur Gerade soll dabei minimal werden. Viele Autoren verwenden diese Methode, um eine Power-Law-Verteilung nachzuweisen. Ein Problem bei diesem Ansatz ist jedoch, dass die Dichte der Punkte mit hohen Graden (rechts im Plot) stark abnimmt. Man spricht dabei auch von Rauschen in den Abschlusspunkten. Diese Punkte, welche nur einen sehr geringen Anteil des Datensatzes repräsentieren, fallen bei der Methode der kleinsten Quadrate besonders ins Gewicht und beeinflussen die Schätzung von γ folglich signifikant. Man versucht daher auf verschiedene Art und Weise diesem Problem entgegenzuwirken. Die folgenden Vorgehensweisen, welche sich alle auf die entsprechenden Log-Log-Plots beziehen, finden in der Praxis Verwendung:

- Einfache lineare Regression auf allen Datenpunkten. Durch das Rauschen in den letzten Datenpunkten erhält man hierbei meist nur eine ungenaue Schätzung für γ.
- Lineare Regression auf den ersten fünf Punkten (ab x_{min}), da allein diese schon den größten Teil der Daten ausmachen [GMY04]. Dieser

Ansatz versucht dem Rauschen in den letzten Punkten entgegenzuwirken, indem die Punkte einfach ignoriert werden.

- Lineare Regression nach Durchschnittsbildung. Hierbei werden Datenpunkte in Intervalle unterteilt und dem Intervall dann der Durchschnittswert seiner Punkte zugeteilt [New04]. Wichtig hierbei ist die Wahl der Intervallgröße. Am besten eignen sich exponentiell wachsende Intervalle, da diese im Log-Log-Plot gleich breit erscheinen und gerade bei den letzten Punkten die Intervalle dann groß genug sind, um einzelne Ausreißer durch andere Punkte im Intervall auszugleichen. Wählt man die Intervallgröße jedoch zu groß, wird die Schätzung des Exponenten zu ungenau aufgrund des hohen Informationsverlustes. Es bleibt zudem mit dieser Methode meistens immer noch ein vermindertes Rauschen in den letzten Datenpunkten bestehen.

- Lineare Regression auf der kumulativen Verteilung. Geplottet wird hier die Wahrscheinlichkeit $P(x)$, dass X einen Wert größer oder gleich x annimmt:

$$P(X \geq x) = \int_x^\infty p(x')\,\mathrm{d}x'.$$

  Wenn nun $p(x)$ einer Power-Law-Verteilung zu Grunde liegt, lässt sich nach [New04] zeigen, dass $P(X \geq x)$ ebenfalls einem Power-Law folgt:

$$P(X \geq x) = c \cdot \int_x^\infty x'^{-\gamma}\,\mathrm{d}x' = \frac{c}{\gamma - 1} \cdot x^{-(\gamma-1)}.$$

  Der Power-Law-Exponent der kumulativen Verteilung ist folglich $\gamma - 1$ und führt im Log-Log-Plot zu einer etwas flacheren Geradensteigung als die der Dichtefunktion. Ein Vorteil bei dieser Methode ist, dass wir keinen Verlust an Informationen durch Durchschnittsbildung haben und uns keine Gedanken um Intervallgrößen machen zu brauchen.

Goldstein, Morris und Yen stellen in [GMY04] empirische Vergleiche der Verfahren an und gelangen zu dem Ergebnis, dass die 5-Punkte-Methode den Exponenten am genauesten abzuschätzen vermag, während die einfache Regression auf dem gesamten Datenbestand die größte Abweichung verursacht. Sie behaupten jedoch, graphische Methoden zur Bestimmung des Power-Law-Exponenten seien generell viel zu varianzbehaftet. Eine weitere, nicht-graphische Möglichkeit stellt z.B. die Abschätzung des Exponenten anhand des Maximum-Likelihood-Prinzips dar.

5.1.2.2 Maximum-Likelihood-Schätzung Gesucht wird hierbei der Wert des Exponenten, der die Wahrscheinlichkeit maximiert, dass die zu untersuchende Größe einem solchen Power-Law zu Grunde liegt. Die Wahrscheinlichkeit, dass n unabhängige Stichprobenwerte x_i aus einer solchen Verteilung stammen, entspricht ([New04])

$$P(x|\gamma) = \prod_{i=1}^{n} p(x_i).$$

Betrachtet wird nun umgekehrt die Wahrscheinlichkeit $P(\gamma|x)$, dass unter den gegebenen Stichprobenwerten der Exponent γ einen konkreten Wert annimmt. Nach dem Satz von Bayes hängen diese beiden Wahrscheinlichkeiten wie folgt zusammen:

$$P(\gamma|x) = P(x|\gamma) \cdot \frac{P(\gamma)}{P(x)}.$$

Da die Stichprobe x konstant bleibt, ist $P(x)$ auch immer gleich. Ebenso nimmt man an, dass $P(\gamma)$ einem konstanten Wert entspricht. Betrachtet man daher den natürlichen Logarithmus auf beiden Seiten der Gleichung (durch Logarithmusbildung bleibt das Maximum einer Funktion an derselben Stelle), so entspricht der Quotient $\frac{P(\gamma)}{P(x)}$ nur noch einer additiven Konstante und ist somit irrelevant für die Maximumsfindung. Anstatt $\ln P(\gamma|x)$ können wir daher auch $\ln P(x|\gamma)$ in Abhängigkeit von γ maximieren. Wenn wir unsere Likelihood-Funktion

$$L = \ln P(x|\gamma)$$

nun nach γ ableiten und mit Null gleichsetzen, so können wir die resultierende Gleichung wie in [New04] nach γ auflösen und erhalten das Maximum bei

$$\gamma = 1 + \frac{n}{\sum_{i=1}^{n} \ln \frac{x_i}{x_{min}}}.$$

Um den Power-Law-Exponenten γ anhand des Maximum-Likelihood-Verfahrens abzuschätzen, genügt es also, die beobachteten Werte x_i in die Formel einzusetzen und diese auszurechnen. Newman gibt in [New04] zudem eine Abschätzung für die Standardabweichung σ von γ an, sobald γ mit dem oben beschriebenen Verfahren bestimmt wurde:

$$\sigma = \frac{\gamma - 1}{\sqrt{n}}.$$

5.1.3 Eigene Ergebnisse und Bewertung

Es soll nun für unsere vier sozialen Netzwerke untersucht werden, ob die Verteilung der Anzahl Freunde pro Individuum einem Power-Law

folgt. Wenn dies der Fall ist, soll zudem der Power-Law-Exponent ermittelt werden. Für die Messung wurde der unbereinigte Datensatz verwendet, da es für die Verteilung des Knotengrads keine Rolle spielt, ob ausgehende Kanten zu nicht eingesehenen Profilen führen. Abbildung 5 zeigt einen Log-Log-Plot, in dem wie in Abschnitt 5.1.2.1 beschrieben die Anzahl Knoten mit ausgehendem Grad k gegen k aufgetragen wurden. Die vertikale Trennlinie bei den Netzwerken B und D markiert den Wert

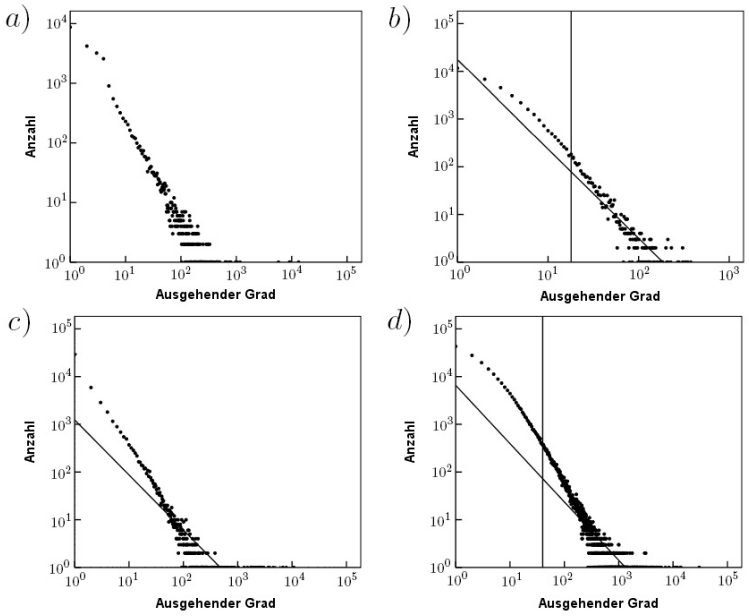

Abbildung 5: Verteilung der ausgehenden Knotengrade auf logarithmischen Skalen für die sozialen Netzwerke A, B, C und D.

x_{min}. Es gehen also nur solche Werte in die Berechnung der Regressionsgeraden mit ein, die größer oder gleich x_{min} sind. Abbildung 6 zeigt zudem die entsprechenden kumulativen Verteilungen, während für Abbildung 7 die Punkte in exponentiell wachsende Intervalle mit anschließender Durchschnittsbildung unterteilt wurden. Es wurde zudem anhand der Maximum-Likelihood-Methode ein Schätzer für γ berechnet. Die geschätzten Power-Law-Exponenten der verschiedenen Methoden sind in Tabelle 2 abzulesen. Sie gibt außerdem jeweils die Standardabweichung σ sowie für die graphischen Methoden den Korrelationskoeffizienten r an, welcher ein Maß für die Stärke des linearen Zusammenhangs darstellt und damit angibt, wie gut die Regressionsgerade die Punkte abbildet (und somit wie gut die Verteilung durch ein Power-Law beschrieben wird). Dieser kann Werte zwischen 0 (kein Zusammenhang) und 1 (maximaler Zusammenhang) annehmen. Nach Abbildung 6 ist deutlich zu erkennen, dass die Verteilung des Knotengrads in Netzwerk A keine Gerade darstellt, sondern eher ex-

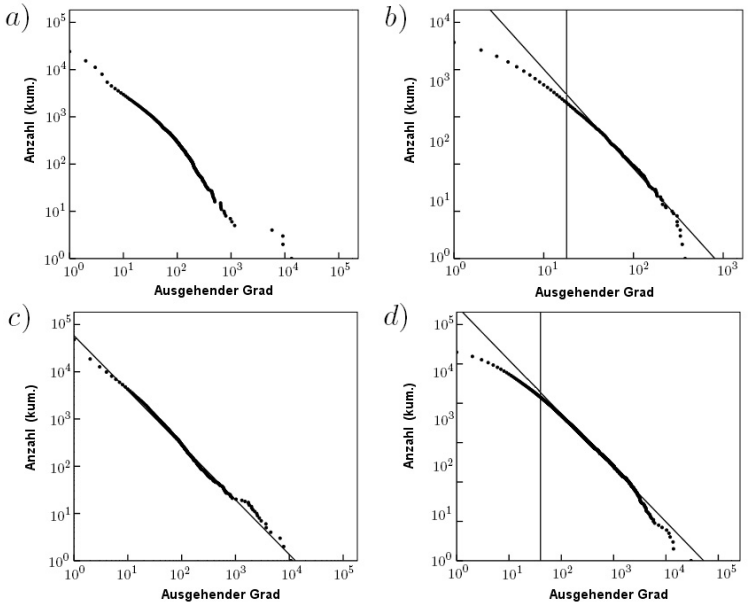

Abbildung 6: Kumulative Verteilung der ausgehenden Knotengrade für die sozialen Netzwerke A, B, C und D.

ponentiell abfällt. Es ergibt daher keinen Sinn, eine lineare Regression für dieses Netzwerk durchzuführen. Ansonsten lässt sich jedoch in allen anderen Netzwerken Linearität im Log-Log-Plot vermuten. Die Anzahl Freunde in diesen Netzwerken scheint somit einem Power-Law zu folgen. Bis auf Netzwerk C ist dies allerdings erst ab den höheren Graden zu beobachten ($x_{min} = 18$ für Netzwerk B und $x_{min} = 40$ für Netzwerk D). Vorher lässt sich für Netzwerke B und D nach Abbildung 6 eine Logarithmische Normalverteilung vermuten (eine Verteilung, bei der der Logarithmus einer Zufallsvariable normalverteilt ist). Eine Abweichung von der Power-Law-Verteilung ist bei diesen Netzwerken zudem für die Abschlusspunkte in der kumulativen Verteilung zu beobachten, welche ebenfalls exponentiell abfallen („exponential cutoff“).

Was die Schätzung des Exponenten betrifft, so scheinen die einfache Regression auf allen Punkten sowie die Intervall-Methode der Vielzahl an Punkten mit hohen Graden zum Opfer zu fallen. Wie in Abschnitt 5.1.2.1 bereits erwähnt, haben diese den unerwünschten Effekt, dass sie die Steigung der Regressionsgeraden stark beeinflussen, obwohl sie nur einen geringen Anteil der Stichprobe ausmachen. Dies hat für die oben genannten Methoden zur Folge, dass die Geradensteigung zu niedrig ausfällt. Das Problem bei der Intervallbildung ist des Weiteren, dass die Intervalle nicht groß genug sind, um die breite Spanne der nur einmal auftretenden Grade zusammenzufassen. Bei größeren Intervallen würde jedoch

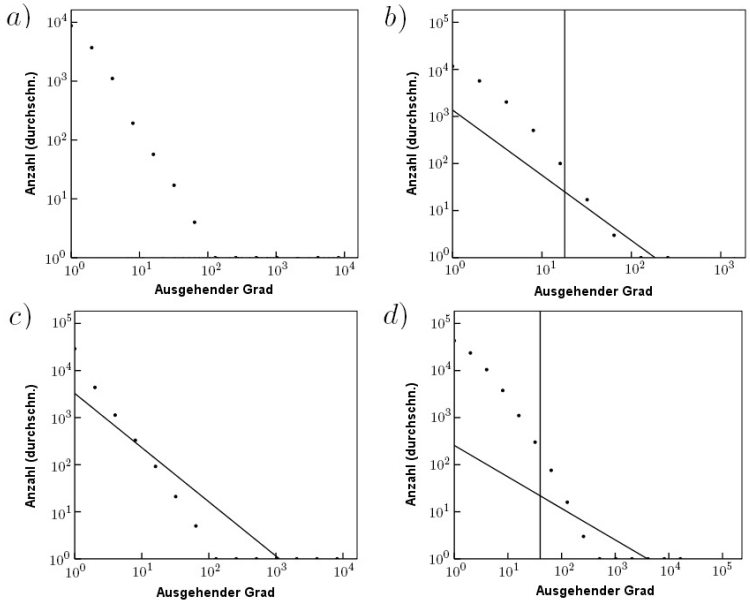

Abbildung 7: Verteilung der ausgehenden Knotengrade anhand exponentiell wachsender $\log_2$-Intervalle für die sozialen Netzwerke A, B, C und D.

die Anzahl Punkte so stark reduziert werden, dass keine genaue Schätzung mehr möglich wäre. Bei der kumulativen Verteilungskurve hingegen fällt die Schätzung eher zu hoch aus (außer für Netzwerk C), da die Punkte weit rechts im Plot zu stark abfallen. Dies tritt dann auf, wenn in der nicht-kumulativen logarithmischen Verteilung die Punkte mit hohen Graden sehr dicht beieinander liegen. Bei der Maximum-Likelihood-Schätzung hingegen fallen die Punkte mit hohen Graden weniger ins Gewicht, da hier alle beobachteten Knotengrade mit derselben Gewichtung in die Berechnung eingehen (siehe Formel für γ in Abschnitt 5.1.2.2) und hohe Grade nur einen geringen Anteil der Stichprobe ausmachen.

Diesen Überlegungen vorausgehend, lässt sich nach Tabelle 2 für das Power-Law in Netzwerk B ein Exponent von ca. 2.8 abschätzen. Für den Exponenten von Netzwerk C hingegen kann keine genaue Aussage gemacht werden. Er liegt wahrscheinlich zwischen 2 (5-Punkte) und 2.6 (MLE). Eindeutiger dagegen sind die Schätzungen für das große Netzwerk D. Hier beträgt der Exponent ca. 2.2.

Betrachtet man den Plot in Abbildung 5, so zeigt sich, dass Netzwerk A vier Datenpunkte aufweist, deren Werte stark abweichend zwischen 5,000 und 10,000 liegen. Auch Netzwerk D weist einen solchen Ausreißer mit einem Grad von über 30,000 auf. Eine genauere Betrachtung dieser Punkte konnte jedoch keinen Aufschluss über den Grund ihres hohen Knotengrads geben: Bei allen handelt es sich scheinbar um gewöhnliche Profile

Netzwerk	Methode	γ	σ	r
B	Linear	1.870	0.075	0.905
	Linear (5 Punkte)	2.314	0.560	0.922
	Linear (Kumulativ)	3.102	0.027	0.988
	$\log_2$-Intervalle	1.385	0.398	0.926
	MLE	2.766	0.040	/
C	Linear	1.154	0.042	0.856
	Linear (5 Punkte)	1.979	0.083	0.997
	Linear (Kumulativ)	2.158	0.006	0.996
	$\log_2$-Intervalle	1.151	0.147	0.915
	MLE	2.567	0.007	/
D	Linear	1.222	0.024	0.853
	Linear (5 Punkte)	2.116	0.406	0.949
	Linear (Kumulativ)	2.370	0.004	0.996
	$\log_2$-Intervalle	0.669	0.190	0.800
	MLE	2.209	0.010	/

Tabelle 2: Geschätzter Power-Law-Exponent γ sowie Standardabweichung σ und Korrelationskoeffizient r für die sozialen Netzwerke B, C und D anhand verschiedener Methoden ermittelt.

von Privatpersonen. Es hat sich jedoch gezeigt, dass eine Analyse ohne jene Punkte sich kaum auf das Ergebnis auswirkt: Während für Netzwerk D die MLE-Schätzung keine merkliche Änderung aufweist, so sind die Schätzer für die einfache und die kumulative Verteilung um ca. 0.02 bzw. 0.01 höher. Stärker hingegen wirkt sich der Ausschluss auf die Intervall-Methode aus, denn dort nimmt der Schätzer um ca. 0.14 zu, resultierend aus einem zusätzlichen Intervall, welches lediglich aus dem ausreißenden Punkt besteht.

5.1.4 Vergleich mit anderen Ergebnissen

Um einen Vergleich mit anderen sozialen Netzwerken herzustellen, so wurde z.B. in einem E-Mail-Netzwerk ($|V| = 59,912$, $|E| = 86,300$) ein Power-Law in der Anzahl Bekanntschaften mit einem Exponenten von 1.49 für eingehende Mails und 2.03 für ausgehende beobachtet [EMB02]. Auch die Anzahl verschiedener Personen, zu denen Telefonanrufe innerhalb eines Tages getätigt wurden ($|V| = 47$ Mio, $|E| = 80$ Mio), weist ein Power-Law

mit $\gamma = 2.1$ auf [ACL00]. Liljeros et al. zeigen in [LEA+01], dass selbiges für die Anzahl sexueller Kontakte in einem Netzwerk mit 2,810 Individuen gilt. Es wurde dabei über einen Zeitraum von einem Jahr Protokoll geführt und ein Power-Law-Exponent von 3.54 für Frauen und $\gamma = 3.31$ für Männer herausgearbeitet. Power-Law-Verteilungen, deren Exponenten außerhalb $1 < \gamma < 4$ liegen, kommen nach [CF06] in der Realität nur selten vor.

5.1.5 Ausblick

Bei der Untersuchung haben wir uns auf die Verteilung des ausgehenden Knotengrads beschränkt. Man könnte jedoch weiterhin die eingehenden Kanten betrachten und prüfen, ob diese demselben Muster folgen oder zumindest Zusammenhänge bestehen. Was die eigentlichen Methoden zur Bestimmung des Power-Law-Exponenten betrifft, so haben alle vorgestellten Verfahren gemeinsam, dass mit dem Auge abgeschätzt werden muss, ab welchem Wert für x die Power-Law-Beziehung gilt. Da eine solche Abschätzung zwangsläufig ungenau ist, wirkt sich dies negativ auf die Schätzung des Exponenten aus. Crovella und Taqqu stellen daher in [CT99] einen nicht-parametrischen Algorithmus vor, der γ ohne Angabe von x_{min} abzuschätzen vermag. Nach eigenen Angaben nimmt die Genauigkeit ihres Schätzers mit zunehmender Größe des betrachteten Netzwerks zu. Es wird jedoch keine Angabe über die Varianz des Schätzers gemacht.

Als Gütemaß für die Angleichung der Verteilung des Knotengrads an ein Power-Law, wurde für die graphischen Methoden der Korrelationskoeffizient bestimmt. Um jedoch eine formalere Untersuchung durchzuführen, werden inzwischen immer häufiger statistische Hypothesentests angewendet (siehe [GMY04] und [BWG06]).

5.2 Durchmesser

1967 führte der Forscher Stanley Milgram ein Experiment durch: Er besorgte sich Namen von zufälligen Personen, die in Nebraska und Boston wohnten und schrieb diese an. Inhalt des Schreibens war, dass die Briefe eine Zielperson in Massachussetts erreichen sollten mit der Aufforderung, den Brief an diejenige Person weiterzuschicken, die dem Ziel wohl am nächsten sei ([Mil67],[TM69]). Die betroffene Person sollte genauso vorgehen usw., bis der Brief sein Ziel erreicht. Zudem sollte jeder seinen Namen auf dem Brief hinterlassen, um am Ende die Länge der Kette rekonstruieren zu können. Das Ergebnis des Experiments war, dass die Mehrheit der Briefe über eine Kette von nur fünf bis sechs Personen lief. Das legt die Theorie nahe, dass jeder mit jedem (zumindest in den USA) über eine Kette von maximal sechs Personen in Beziehung steht. Dieses Phänomen ist auch unter dem Namen „Sechs Grade der Trennung“ oder „Small-World-Effekt“ bekannt.

Auch wenn Milgrams Theorie aufgrund lückenhafter Belege in der Kritik steht, so lässt sich die Idee dennoch gut auf die Analyse von großen Graphen übertragen. Es gilt dabei, den Durchmesser zu finden, denn dieser untersucht genau das für einen Graphen, was Milgram in seinem Experiment für die Bevölkerung der USA herauszubekommen versuchte. Es haben sich dazu zwei Ansätze in der Literatur herausgebildet, die den Durchmesser eines Graphen zu erfassen versuchen und somit ein Maß für die Stärke des Small-World-Effekts darstellen: der *mittlere Durchmesser* und der *effektive Durchmesser*.

5.2.1 Mittlerer Durchmesser

Der *mittlere* Durchmesser ist einfach definiert als die durchschnittliche Länge aller kürzesten Wege im Graphen [CF06]. Wir schließen dabei Wege der Länge 0 mit ein. Für sämtliche Knotenpaare muss also der kürzeste Pfad bestimmt werden mit anschließender Durchschnittsbildung. Eine Möglichkeit, dies zu bewerkstelligen, ist für jeden Knoten einen Breitendurchlauf durchzuführen [New03]. Immer wenn ein neuer Knoten gefunden wurde, wird die Distanz zu diesem Knoten, welche der Tiefe der aktuellen Ebene entspricht, zur Summe aller gefundenen kürzesten Wege hinzuaddiert. Die Anzahl der vom aktuellen Knoten aus erreichbaren Knoten wird zudem zur Summe aller gefundenen Knotenpaare hinzugezählt. Dieser Vorgang wird dann für jeden Knoten des Graphen wiederholt und am Ende die Summe der Länge sämtlicher kürzester Pfade durch die Anzahl aller gefundenen Knoten dividiert, um den Mittelwert zu bestimmen. Dies benötigt $O(|E| \cdot |V|)$ Zeit ($|V|$ mal die Laufzeit für den Breitendurchlauf aus Abschnitt 3.2.2) und zeigt sich äußerst zeitintensiv für große Graphen. Der mittlere Durchmesser kann außerdem nur für zusammenhängende Graphen bestimmt werden, da in unzusammenhängenden Graphen Knotenpaare existieren, die eine Distanz von unendlich aufweisen und somit der mittlere Durchmesser ebenfalls unendlich wird. Man betrachtet dann üblicherweise die größte zusammenhängende Komponente des Graphen.

Ein dem mittleren Durchmesser ähnliches Maß aus der Literatur ist die *charakteristische Pfadlänge*. Bei dieser wird anstatt des Mittelwerts einfach der Median der gemittelten Pfadlängen aller Knoten betrachtet [BT02].

5.2.2 Effektiver Durchmesser

5.2.2.1 Definitionen Zunächst führen wir ein Maß für die „Verbundenheit“ eines Graphen ein, welches auf der Nachbarschaftsgröße seiner Knoten basiert. Palmer et al. betrachten in [PGF02] einen gerichteten Graphen und definieren $dist(u, v)$ als die Anzahl Kanten auf dem kürzesten Pfad von u nach v. Die *individuelle Nachbarschaftsfunktion* eines Knotens u in h ist

dann die Anzahl der Knoten mit höchstens Abstand h zu u:

$$IN(u,h) = |\{v : v \in V, dist(u,v) \leq h\}|.$$

Sie kann demnach als Maß für die Verbundenheit eines Knotens aufgefasst werden. Die Konnektivität des gesamten Graphen hingegen wird durch die *Nachbarschaftsfunktion* in h ausgedrückt und stellt die Anzahl Knotenpaare, zwischen denen ein Weg der Länge h oder kleiner besteht, dar:

$$N(h) = |\{(u,v) : u \in V, v \in V, dist(u,v) \leq h\}|.$$

Sie ist also die Summe der individuellen Nachbarschaftsfunktionen in h von sämtlichen Knoten des Graphen:

$$N(h) = \sum_{u \in V} IN(u,h).$$

Für $h = 0$ ist die Nachbarschaftsfunktion für einen Graphen auch ohne aufwendige Berechnung bereits bekannt, denn es gilt

$$N(0) = |V|,$$

was leicht einzusehen ist, da bei einem maximalen Abstand von 0 Hops jeder Knoten nur sich selbst erreichen kann. Ist der Graph frei von Kanten, die an beiden Enden denselben Knoten berühren, so gilt für $h = 1$ außerdem

$$N(1) = |V| + |E|,$$

denn zuzüglich zu den Paaren aus $N(0)$ kommen jetzt noch sämtliche Paare im Abstand von einem Hop, was offensichtlich der Anzahl Kanten im Graphen entspricht.

Wird nun die Nachbarschaftsfunktion eines Graphen für verschiedene Werte von h in einem Plot (auch „Hop-Plot“ genannt) aufgetragen, lässt sich leicht ablesen, wie groß der *effektive Durchmesser* d_{eff} eines Graphen ist. Dieser, welcher auch als „Exzentrizität“ bekannt ist, ist definiert als die minimale Anzahl Hops, über die sich der Großteil der verbundenen Knotenpaare erreichen kann [PSF+01]. In der Literatur wird häufig ein Mindestanteil von 90% angeführt. Wir wollen uns daher für den weiteren Verlauf ebenfalls auf diesen Schwellwert festlegen. Der effektive Durchmesser ist also dann erreicht, wenn die Nachbarschaftsfunktion für den zunehmenden Wert von h nur noch entsprechend gering ansteigt. Fordert man hingegen 100% verbundener Knotenpaare, so zeigt sich der effektive Durchmesser im Gegensatz zum mittleren Durchmesser und der charakteristischen Pfadlänge empfindlich gegenüber Ausreißern und eignet sich somit zur Worst-Case-Analyse. Ein weiterer Vorteil der Exzentrizität ist, dass ihre Definition auch für unzusammenhängende Graphen gilt. Für unsere

Netzwerke ist dies jedoch irrelevant, da es sich, bedingt durch die Art und Weise der Extraktion, immer um zusammenhängende Graphen handelt. Im Folgenden sollen nun Verfahren vorgestellt werden, wie die Nachbarschaftsfunktion eines Graphen (und somit auch der effektive Durchmesser) berechnet werden kann. Wir unterscheiden dabei zwischen exakten und approximativen Methoden.

5.2.2.2 Exakte Berechnung

Matrixmultiplikation Eine Möglichkeit, Wege der Länge h in einem Graphen zu suchen, ist die Adjazenzmatrix des Graphen mit h zu potenzieren [PGF02]. Zur Erinnerung: Das Produkt einer $(v \times w)$ -Matrix $A = (a_{kl})$ $(1 \leq k \leq v, 1 \leq l \leq w)$ und einer $(r \times s)$ -Matrix $B = (b_{fg})$ $(1 \leq f \leq r, 1 \leq g \leq s)$ ist definiert, wenn gilt $w = r$ [Wit01]. Das Resultat ist eine $(v \times s)$ -Matrix C mit den Elementen

$$c_{ij} = \sum_{t=1}^{w} a_{it} b_{tj}, 1 \leq i \leq v, 1 \leq j \leq s.$$

Ein Eintrag in der potenzierten Matrix gibt nun die Anzahl der Wege mit Länge h zwischen dem entsprechenden Knotenpaar an. Sind alle Einträge 0, so gibt es keine zwei Knoten, die sich über h Kanten erreichen können. Uns interessiert hierbei allerdings nicht, wie viele Wege der Länge h zwischen einem Knotenpaar existieren, sondern lediglich dass einer existiert. $N(0)$ entspricht dann also der Anzahl Einträgen in der Matrix A^0, die ungleich 0 sind. Für zunehmenden Wert h ergibt sich $N(h)$ dann aus $N(h-1)$ zuzüglich der Anzahl der neu hinzugekommenen Knotenpaare in A^h. Es ist also darauf zu achten, dass ein Knotenpaar immer nur beim ersten Mal gezählt wird, sobald ein Weg zwischen ihm gefunden wurde, da dieser der kürzeste Weg ist. Dies hat zur Folge, dass es zu merken gilt, welche Paare bereits gezählt wurden. Für ein Beispiel sei an den Graphen aus Abbildung 1 a) erinnert, dessen Adjazenzmatrix wie folgt gegeben war:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Wir potenzieren A nun so lange, bis keine neuen Wege mehr gefunden werden:

$$A^0 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \qquad A^1 = A, \qquad A^2 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$A^3 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \qquad A^4 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Auch für $h > 4$ resultiert das Produkt in der Nullmatrix. Es zeigt sich, dass für $h = 2$, also bei einer Pfadlänge von zwei Hops, der letzte kürzeste Weg gefunden wurde. Der effektive Durchmesser des Graphen ist aber 1, denn über maximal so viele Hops können sich mindestens 90%, d.h. 12 der 13 verbundenen Knotenpaare, gegenseitig erreichen. Die Laufzeit für diese Methode beträgt mit dem naiven Algorithmus der Matrixmultiplikation $O(|V|^3)$. Der derzeit beste bekannte Algorithmus benötigt $O(|V|^{2.38})$ Zeit (Coppersmith und Winograd, 1987). Beide kommen jedoch nicht ohne $O(|V|^2)$ Speicher aus, was für große Graphen inpraktikabel ist.

Breitensuche Führt man eine Breitensuche von einem Knoten u für h Ebenen aus und zählt alle gefundenen Knoten, so erhält man die individuelle Nachbarschaftsfunktion $IN(u, h)$ für diesen Knoten [PGF02]. Um die Nachbarschaftsfunktion $N(h)$ eines Graphen zu bestimmen, muss folglich eine Breitensuche für sämtliche Knoten des Graphen in h Ebenen durchgeführt und die Summe aller gefundenen Knoten gebildet werden. Für den effektiven Durchmesser gilt es auch hier den Vorgang für größere h so lange zu wiederholen, bis die Anzahl der insgesamt gefundenen Knoten kaum weiter zunimmt. Diese Methode benötigt nur $O(|V|+|E|)$ Platz, dafür aber $O((|V| \cdot |E|) \cdot d_{eff})$ Zeit. Ein weiterer Nachteil zeigt sich dadurch, dass auf die Kanten bei der Suche mehr oder weniger zufällig zugegriffen wird. Da die Kantendatei sich aufgrund ihrer Größe meist auf der Festplatte befindet, resultiert dies oft in schlechter Performance.

5.2.2.3 Approximative Berechnung Palmer et al. stellen in [PGF02] ein Verfahren namens ANF (Approximate Neighborhood Function) vor, welches aufgrund seiner guten Laufzeit auch für große Graphen geeignet ist. Der Nachteil dieser Methode ist jedoch, dass sie nur ein approximatives Ergebnis für die Nachbarschaftsfunktion eines Graphen liefert. Das Verfahren basiert auf der Überlegung, dass die Menge der von einem Knoten x über

maximal h Hops erreichbaren Knoten dieselbe ist wie die Menge der von x aus erreichbaren Knoten bei höchstens $h-1$ Hops, vereinigt mit den Knoten, die über maximal $h-1$ Hops von den Nachbarn von x aus erreichbar sind. Das heißt, es gilt

$$IN(x,h) = IN(x,h-1) \cup \bigcup_{y \in Y} IN(y,h-1),$$

wobei Y die Menge der Knoten ist, auf die eine gerichtete Kante von x zeigt. Für die Vereinigung müssen nun die Elemente der Mengen mit denen der Ergebnismenge verglichen werden, um zu überprüfen, ob ein Element eventuell nicht schon hinzugefügt wurde. Da dieser Vorgang sehr zeitaufwendig ist, wird eine Methode vorgeschlagen, wie die Anzahl verschiedener Elemente in den zu vereinigenden Mengen möglichst effizient bestimmt werden kann: Dazu führt man für jeden der $|V|$ Knoten einen eigenen Bitstring der Länge $|V|$ mit. Jeder Knoten wird hierbei durch ein Bit repräsentiert. Ist ein Bit auf 1 gesetzt, so ist der entsprechende Knoten für den aktuellen Wert von h erreichbar für den Knoten, dem der Bitstring zugeordnet ist. Mit diesem Ansatz lassen sich zwei Knotenmengen einfach vereinigen, indem ein bitweises ODER zwischen ihren Bitfolgen durchgeführt wird. Allerdings benötigt dieses Vorgehen $O(|V|^2)$ Speicher und ist somit ungeeignet für Graphen mit vielen Knoten. Daher wird die Anzahl der erreichbaren Knoten stattdessen nur noch approximiert. Es wird dazu eine probabilistische Zählmethode verwendet, das heißt, jedem Knoten wird nicht mehr ein Bitstring der Länge $|V|$, sondern der Länge $\lceil \log_2 |V| + r \rceil$ zugeordnet, wobei r eine kleine Konstante zur Erhöhung der Genauigkeit ist. Es wird hierbei also nicht mehr jeder Knoten durch ein Bit repräsentiert, sondern das erste Bit repräsentiert jetzt 50% der Knoten, das zweite 25%, usw. Da eine einzige Approximation in der Regel noch nicht sehr zuverlässig ist, wird für jeden Knoten nun nicht mehr ein Bitstring, sondern eine Konkatenation von k Bitfolgen der Länge $\lceil \log_2 |V| + r \rceil$ gespeichert, wobei k eine kleine Konstante ist. Wir bezeichnen diese Verkettung von Bitmasken für einen Knoten x und maximaler Hopanzahl h im Folgenden mit $M(x,h)$. Abbildung 8 stellt nun den Algorithmus dar. Zu Beginn ($h = 0$) wird in jedem der $|V|$ Bitstrings ein Bit auf 1 gesetzt und zwar mit Wahrscheinlichkeit 0.5 das Bit an Position 0, mit Wahrscheinlichkeit 0.25 das Bit an Position 1 und mit Wahrscheinlichkeit 0.5^{i+1} das Bit an Position i. Die Idee dahinter ist: Je mehr Knoten betrachtet wurden (also je mehr bitweise-ODER-Operationen durchgeführt wurden), desto wahrscheinlicher ist es, dass die resultierende Bitfolge weiter rechts Einsen aufweist. Oder umgekehrt: Wenn das Bit an Position 1 (25%) nicht gesetzt ist, so wurden wahrscheinlich erst weniger als 4 Knoten betrachtet.

Für alle $h \geq 1$ ergibt sich $M(x,h)$ nun, indem $M(x,h-1)$ durch bitweises ODER mit den Bitfolgen $M(y,h-1)$ aller Knoten y verknüpft wird, zu denen eine Kante von x aus zeigt. Tabelle 3 zeigt die entsprechenden

ANF-Algorithmus(Graph G)

for each $x \in V$ **do**
$M(x,0)$ = Konkatenation von k Bitmasken, in denen jeweils ein Bit an Position i gesetzt ist (P (Bit i) $= 0.5^{i+1}$)

for each h starting with $h = 1$ **do**
for each $x \in V$ **do**
$M(x,h) = M(x,h-1)$
for each $(x,y) \in E$ **do**
$M(x,h) = M(x,h)$ BITWISE-OR $M(y,h-1)$
for each $x \in V$ **do**
$I\hat{N}(x,h) = \frac{2^b}{0.77351}$, wobei b die durchschnittliche Position des linkesten 0-Bits der k Bitmasken ist
$\hat{N}(h) = \sum_{x \in V} I\hat{N}(x,h)$
return $\hat{N}(h)$

Abbildung 8: Der ANF-Algorithmus.

Bitmasken bezogen auf den Beispielgraphen in Abbildung 1 a). Für $h \geq 2$ treten dabei keine Änderungen in den Bits mehr auf. Wurden also die entsprechenden Vereinigungen für einen Knoten x und einem Wert für h gebildet, so ergibt sich nun die approximierte individuelle Nachbarschaftsfunktion für x aus $\frac{2^b}{\varphi}$, wobei φ der Korrelationsfaktor und b die durchschnittliche Position des linkesten Bits der k Bitstrings von x ist, welches nicht gesetzt wurde (bzw. die Anzahl der Bits, falls alle Bits gesetzt wurden). Es gilt $\varphi \approx 0.77351$, denn Flajolet et al. zeigen in [FM85], dass die Anzahl verschiedener Elemente a mit der Position des linkesten 0-Bits wie folgt zusammenhängt:

$$b \approx \log_2 \ \varphi a.$$

Folglich lässt sich die Anzahl verschiedener Elemente abschätzen, indem nach a aufgelöst wird:

$$a \approx \frac{2^b}{\varphi}.$$

Tabelle 4 demonstriert die Berechnung der approximierten individuellen Nachbarschaftsfunktionen für den Graphen in Abbildung 1 a) auf Basis von Tabelle 3. Hat man diese für den aktuellen Wert von h bestimmt, so ergeben sie aufsummiert die approximierte Nachbarschaftsfunktion $\hat{N}(h)$. Der ganze Vorgang wird dann solange für größere h wiederholt, bis die

Knoten x	$M(x,0)$	$M(x,1)$	$M(x,2)$
1	100 100 001	110 100 101	110 100 101
2	010 100 100	010 100 100	010 100 100
3	100 001 100	110 101 101	110 111 101
4	100 100 100	110 110 100	110 110 100
5	100 010 100	110 110 100	110 110 100

Tabelle 3: Bitmasken für den Graphen aus Abbildung 1 a) mit $r = 0$ und $k = 3$.

Knoten x	$I\hat{N}(x,1)$	$I\hat{N}(x,2)$
1	$b = \frac{2+1+1}{3} = \frac{4}{3}, \frac{2^b}{\varphi} \approx 3.26$	$b = \frac{2+1+1}{3} = \frac{4}{3}, \frac{2^b}{\varphi} \approx 3.26$
2	$b = \frac{0+1+1}{3} = \frac{2}{3}, \frac{2^b}{\varphi} \approx 2.05$	$b = \frac{0+1+1}{3} = \frac{4}{3}, \frac{2^b}{\varphi} \approx 2.05$
3	$b = \frac{2+1+1}{3} = \frac{4}{3}, \frac{2^b}{\varphi} \approx 3.26$	$b = \frac{2+3+1}{3} = 2, \frac{2^b}{\varphi} \approx 5.17$
4	$b = \frac{2+2+1}{3} = \frac{5}{3}, \frac{2^b}{\varphi} \approx 4.1$	$b = \frac{2+2+1}{3} = \frac{5}{3}, \frac{2^b}{\varphi} \approx 4.1$
5	$b = \frac{2+2+1}{3} = \frac{5}{3}, \frac{2^b}{\varphi} \approx 4.1$	$b = \frac{2+2+1}{3} = \frac{5}{3}, \frac{2^b}{\varphi} \approx 4.1$

Tabelle 4: Die approximierten individuellen Nachbarschaftsfunktionen für die Knoten des Graphen aus Abbildung 1 a).

Approximation sich nicht weiter ändert und der effektive Durchmesser des Graphen bestimmbar ist. Bei jeder Iteration wird dabei die Kantenliste einmal sequenziell durchlaufen, da für jeden Knoten alle Kanten für die entsprechenden Vereinigungen betrachtet werden müssen. Es wird also nicht zufällig auf die Kantenliste zugegriffen, wie es beim Traversieren des Graphen der Fall wäre. Für den Beispielgraphen ergibt sich somit ein geschätzter effektiver Durchmesser von 2, denn für $h \geq 2$ tritt keine Änderung mehr auf ($\hat{N}(h \geq 2) \approx 18.68$) und für $\hat{N}(1)$ sind noch keine 90% aller verbundenen Knotenpaare erreicht. Es sei betont, dass mit zunehmender Größe des Graphen die Genauigkeit der Approximation aufgrund ihrer probabilistischen Natur zunimmt.

Die Laufzeit des Algorithmus beträgt $O((|V| + |E|) \cdot d_{eff})$, wobei d_{eff} den effektiven Durchmesser darstellt, denn in jedem der d_{eff} Durchläufe wird zunächst sequenziell die Kantenliste betrachtet und anschließend für jeden Knoten die approximierte Nachbarschaftsfunktion berechnet. Da d_{eff} in der Regel recht klein ist, ist das Verfahren auch für große Graphen geeignet. Der benötigte Speicher ist $O(|V| \cdot \log_2 |V|)$, denn es müssen in jeder Iteration immer nur die aktuellen Bitstrings sowie die des vorangegangenen Durchlaufs gemerkt werden. Um den benötigten Speicher zu re-

duzieren, weisen die Autoren auf eine Modifikation des Algorithmus hin, in der die beiden Bitstringtabellen in je gleichgroße Teile zerlegt und diese zur Laufzeit dann nacheinander in den Speicher geladen werden. Eine weitere Optimierung wird dadurch erreicht, dass für jede Bitfolge ein Zähler für die Anzahl führender Einsen verwendet wird, da viele Bitmasken bereits früh einen großen Satz solcher Einsen aufweisen. Es bietet sich jedoch an, vorher die k parallelen Approximationen ineinander zu verschachteln (also zuerst die Bits an nullter Stelle, dann die Bits an erster Stelle usw.), da so die Anzahl führender Einsen noch größer ist und lediglich ein Zähler benötigt wird.

5.2.3 Eigene Ergebnisse und Bewertung

Es wurden nun die mittleren und die effektiven Durchmesser für unsere sozialen Netzwerke berechnet. Bei allen Netzwerken, in denen ungültige Knoten vorliegen (B und D), wurde dazu der bereinigte Datensatz verwendet. Andernfalls würden ungültige Knoten im Graphen als Knoten mit ausgehendem Grad 0 repräsentiert werden, was eine Verfälschung der Nachbarschaftsfunktion zur Folge hätte. Abbildung 9 zeigt die entsprechenden Hop-Plots. Es wurde dazu das Verfahren auf Basis der Breitensuche sowie der ANF-Algorithmus implementiert, damit die Nachbarschaftsfunktionen exakt berechnet und außerdem approximiert werden konnten. Für die Approximation wurden die Parameter $k = 64$ und $r = 5$ verwendet, da Palmer et al. behaupten, dass anhand dieser Werte eine ausreichend genaue Abschätzung möglich sei [PGF02]. Dazu kommt, dass für größere Werte der benötigte Hauptspeicher eine Berechnung für Netzwerk D auf dem verwendeten System unmöglich machte. Betrachtet man nun die Abbildung, so fällt ins Auge, dass die Approximation der Nachbarschaftsfunktion in allen Fällen zu hoch ausfällt. Die vertikalen Linien in den Plots markieren den effektiven Durchmesser der Graphen auf Basis der exakten Berechnung. Netzwerk D bildet hiervon eine Ausnahme, denn dort konnte aufgrund der Größe keine exakte Berechnung durchgeführt werden. Tabelle 5 führt die berechneten Durchmesser für die sozialen Netzwerke auf, insbesondere gibt sie Aufschluss über die durch ANF approximierten effektiven Durchmesser. Wie der starke Anstieg der Approximation in Abbildung 9 bereits vermuten lässt, schätzt ANF den Durchmesser in allen Fällen zu niedrig ab. Für Netzwerk D lässt sich demnach vermuten, dass der wahre effektive Durchmesser auch 4 beträgt (und der mittlere Durchmesser ca. 3). Die Approximation zeigt sich bei Netzwerk B am ungenauesten. Es bleibt zu untersuchen, ob der Fehler der Approximation abhängig von der Größe des Durchmessers ist. Wir können jedoch festhalten, dass in allen Netzwerken der Small-World-Effekt stark beobachtbar ist, und zwar unabhängig von ihrer Größe. Dies bedeutet also, dass die meisten Leute (mindestens 90%) über eine Kette von nur maximal 4 Hops miteinander bekannt sind.

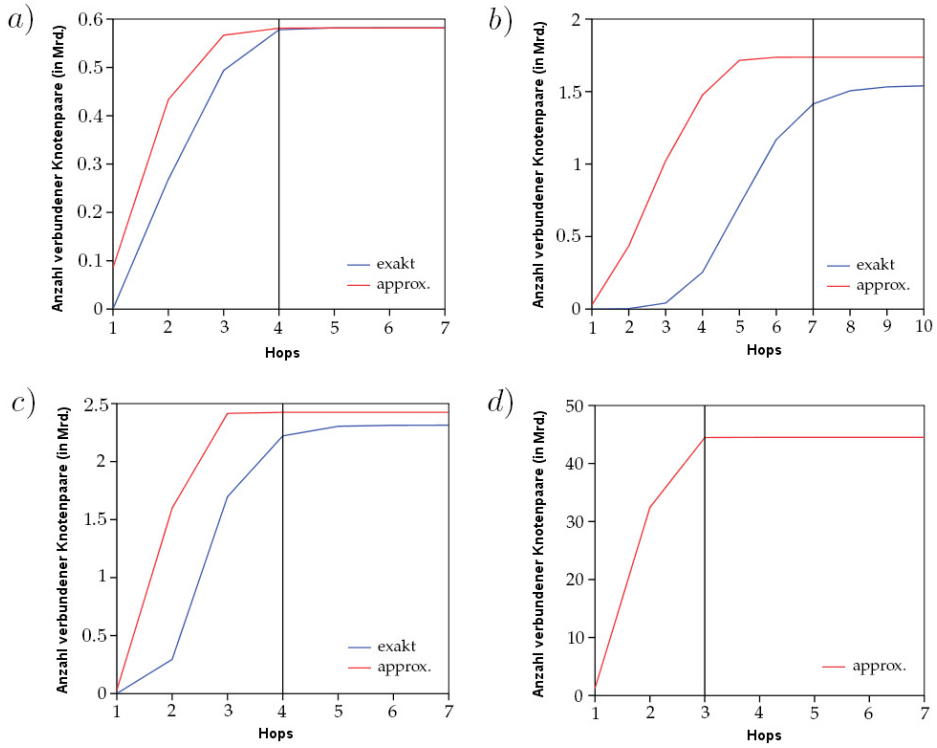

Abbildung 9: Hop-Plots für die sozialen Netzwerke A, B, C und D per exakter sowie mit ANF approximierter Berechnung der Nachbarschaftsfunktion.

Netzwerk B bildet hiervon eine Ausnahme: Mit einem effektiven Durchmesser von 7 bei über 40,000 Knoten lässt es sich allerdings immer noch in die Kategorie Small-World-Netzwerk einordnen. Dass die Konnektivität in diesem Netzwerk nicht so hoch ist wie in den anderen, wird auch dadurch deutlich, dass zwischen mehr als 8% aller möglichen Knotenpaare im Graphen kein Weg besteht (siehe n_{nc} in Tabelle 5). Betrachten wir abschließend noch die mittleren Durchmesser, so fallen diese ähnlich wie die effektiven aus, nur entsprechend kleiner. Bei Netzwerken A und B beträgt die Differenz der beiden Größen 1.3 Hops, bei Netzwerk C hingegen nur 0.72.

Wir haben also gezeigt, dass die hier untersuchten Netzwerke alle den Small-World-Effekt aufweisen. Doch was bedeutet nun ein geringer Durchmesser für soziale Netzwerke im WWW? Er ist z.B. ein Maß dafür, wie stark sich Informationen im Netzwerk verbreiten [New03]. Je kleiner er ausfällt, desto schneller wird die Information durch das komplette Netzwerk propagiert. Anhand des Durchmessers können z.B. Organisationen oder Parteien abschätzen, wie viele Individuen sie kontaktieren müssen, um das gesamte Netzwerk auf sich aufmerksam zu machen. Auf der anderen Seite

Netzwerk	d_{avg}	d_{eff}	d_{apr}	e_{apr}	n_{nc}
A	2.70	4	3	25%	0.13%
B	5.70	7	5	29%	8.31%
C	3.18	4	3	25%	2.04%
D	/	/	3	/	0.00%

Tabelle 5: Durchmesser für die extrahierten Netzwerke: mittlerer Durchmesser d_{avg}, effektiver Durchmesser d_{eff}, Approximation des effektiven Durchmessers d_{apr} durch ANF, Fehler der Approximation e_{apr} und Anteil nicht verbundener Knotenpaare n_{nc}.

ist der Durchmesser für klassische soziale Netzwerke nicht nur ein Maß für die Verbreitung von Informationen, sondern auch für z.B. virale Infektionen. Diese könnten für Online-Netzwerke Computerviren oder Würmer in den Systemen der Mitglieder darstellen. Die hier untersuchten Netzwerke zeigen sich also alle äußerst anfällig.

5.2.4 Vergleich mit anderen Ergebnissen

Es sollen nun weitere Durchmesser anderer Netzwerke zur Gegenüberstellung herangezogen werden: Für das E-Mail-Netzwerk der Universität Kiel z.B., dem wir bereits in Abschnitt 5.1.4 begegnet sind, wurde jeglicher Mailverkehr über einen Zeitraum von über 3 Monaten aufgezeichnet. Im Graphen besteht dann eine Kante zwischen zwei Knoten, wenn E-Mails zwischen diesen ausgetauscht wurden. Ebel et al. konnten hierfür einen mittleren Durchmesser von 4.95 bestimmen [EMB02]. Ein anderes Beispiel stellt ein Schauspieler-Netzwerk dar ($|V| = 225,226$), in dem eine Verbindung dann besteht, wenn zwei Schauspieler im selben Film mitgespielt haben (http://www.imdb.com). Der mittlere Durchmesser dieses Netzwerks beträgt 3.65 [WS98]. In einem wissenschaftlichen Kollaborationsnetzwerk sind zwei Forscher dann miteinander verbunden, wenn sie bereits ein Paper zusammen geschrieben haben. Newman weist in [New01] einen mittleren Durchmesser von 9.7 und einen effektiven Durchmesser von 31 (allerdings für einen Mindestanteil verbundener Knotenpaare von 100%) für ein solches Netzwerk aus dem Bereich Informatik nach ($|V| = 11,994$). Für die Physik ($|V| = 52,909$) werden entsprechend Werte von 5.9 und 20 angeführt.

5.2.5 Ausblick

Für den effektiven Durchmesser konnten wir durch ein approximatives Verfahren auch eine Schätzung für den großen Graphen des Netzwerks D

bestimmen. Für den mittleren Durchmesser hingegen bestünde eine einfache Möglichkeit darin, Breitendurchläufe für eine zufällige Stichprobe der Punkte durchzuführen und den Mittelwert aller kürzesten Wege dann auf den gesamten Graphen hochzurechnen. Es bleibt jedoch Gegenstand zukünftiger Arbeit, einen geeigneteren Näherungsalgorithmus zu finden.

Faloutsos et al. zeigen in [FFF99] für einen Internetgraphen, dass die individuelle Nachbarschaftsfunktion $IN(u, h)$ eines zentralen Knotens u exponentiell mit geringen Werten für h (viel kleiner als der Durchmesser des Graphen) wächst, also $IN(u, h) = ch^{\mathcal{H}}$, wobei c und $\mathcal{H}$ Konstanten sind. Nach [New03] gilt dies für viele verschiedene Netzwerktypen, insbesondere zufällig generierten Netzwerken. Zudem wird gefolgert, dass in solchen Netzwerken der mittlere Durchmesser logarithmisch mit der Netzwerkgröße wächst, also $d_{avg} \approx \log |V|$. Man spricht dann ebenfalls vom „Small-World-Effekt". Bei Netzwerken mit Power-Law-Verteilung im Knotengrad steigen nach [New03] die mittleren Durchmesser sogar noch langsamer an, nämlich wie $\log |V| / \log \log |V|$. Dieses Kriterium könnte weiterführend auf die sozialen Netzwerke angewendet werden, um den Small-World-Effekt zu unterstreichen.

5.3 Clustering

5.3.1 Definitionen

Netzwerke der realen Welt weisen meist eine stärkere Community-Bildung auf als zufällig generierte Netzwerke [CF06]. Wir sprechen dabei auch von der Netzwerkdichte oder vom Grad des „Clusterings". Dieser wird häufig durch die Transitivität in einem Graphen ausgedrückt, also die Wahrscheinlichkeit, dass die Nachbarn der Nachbarn eines Knoten auch mit ihm benachbart sind. Für unsere sozialen Netzwerke wird also untersucht, ob ein Akteur auch mit den Freunden seiner Freunde befreundet ist. Transitivität wird in Graphen durch Dreiecke ausgedrückt, wobei ein Dreieck ein Kreis der Länge 3 ist. Je größer also der Anteil Dreiecke in einem Graphen, desto größer der Grad des Clusterings. In der Literatur finden sich zwei ähnliche Maße für die Stärke dieses Effekts, die beide auf der Anzahl Dreiecke sowie der Anzahl Knotentripel basieren. Wir unterscheiden zwischen dem *Clustering-Koeffizienten* und der *Transitivität* eines Graphen.

5.3.1.1 Clustering-Koeffizient Watts und Strogatz führen zunächst den lokalen Clustering-Koeffizienten für einen Knoten i mit k_i Nachbarn und n_i Kanten zwischen den Nachbarn ein. Für gerichtete Graphen ist dieser nach [WS98] definiert als

$$C_i = \begin{cases} \frac{n_i}{k_i(k_i-1)} & \text{falls } k_i > 1, \\ 0 & \text{falls } k_i = 0 \text{ oder } 1. \end{cases}$$

Er berechnet sich also aus der Anzahl der existierenden Kanten zwischen den Nachbarn, dividiert durch die maximal mögliche Anzahl. Er kann demnach höchstens den Wert 1 annehmen, nämlich dann, wenn jeder Nachbar des Knotens auch zu allen anderen Nachbarn desselben Knotens verbunden ist. Der kleinstmögliche Wert hingegen ist 0 und tritt dann auf, wenn kein Nachbar Kanten zu anderen Nachbarn des Knotens aufweist. Der Clustering-Koeffizient stellt demnach die mittlere Wahrscheinlichkeit dar, dass zwei Knoten, die beide Nachbarn desselben Knotens sind, ebenfalls miteinander verbunden sind. Betrachten wir den gerichteten Graphen aus Abbildung 1 a), so hat Knoten 1 beispielsweise nur einen Nachbarn (nämlich Knoten 2) und daher ist $C_1 = 0$. Knoten 3 hingegen weist drei Nachbarn auf (Knoten 1, 2 und 4), wobei zwischen diesen Nachbarn zwei Kanten existieren ($1 \rightarrow 2, 4 \rightarrow 2$). Es ist folglich $C_3 = \frac{2}{3 \cdot 2} = \frac{1}{3}$.

Betrachtet man stattdessen ungerichtete Graphen, so können nur halb so viele Kanten zwischen den Nachbarn existieren wie im gerichteten Fall. Entsprechend erfordert dies eine Halbierung des Nenners in der Definition des Clustering-Koeffizienten. Die Größe n_i ist in diesem Fall äquivalent zu der Anzahl Dreiecke im Graphen, an denen der Knoten i beteiligt ist, während der Divisor der Anzahl Tripel im Graphen entspricht, welche Knoten i als mittleren Knoten haben. Ein Knotentripel besteht dabei aus drei verbundenen Knoten, wobei i der zentrale Knoten ist, der zu den anderen beiden adjazent ist. Der lokale Clustering-Koeffizient lässt sich für den ungerichteten Fall folglich umschreiben in

$$C_i = \frac{\text{Anzahl Dreiecke, an denen } i \text{ beteiligt ist}}{\text{Anzahl Tripel, in denen } i \text{ der mittlere Knoten ist}},$$

wobei er wieder 0 ist, falls $k_i = 0$ oder 1 [New03]. Nachdem nun für jeden Knoten ein Maß für die Dichte der Dreiecke bestimmt wurde, an denen er beteiligt ist, lässt sich nach Watts und Strogatz der globale Clustering-Koeffizient C für einen Graphen einfach als den Mittelwert der lokalen Clustering-Koeffizienten definieren:

$$C = \frac{1}{|V|} \sum_{i=1}^{|V|} C_i.$$

5.3.1.2 Transitivität Die alternative Definition für den Grad des Clusterings stellt der Anteil aller verbundenen Tripel dar, die Dreiecke bilden [CF06]. Dieser kann für einen ungerichteten Graphen ausgedrückt werden durch:

$$T = \frac{3 \times \text{Anzahl Dreiecke im Graphen}}{\text{Anzahl verbundener Knotentripel}}.$$

Der Faktor 3 ist dadurch begründet, dass ein Dreieck aus drei solchen verbundenen Tripeln besteht. Auch diese Definition lässt nur Werte für T zwischen 0 und 1 zu. Sie drückt somit die mittlere Wahrscheinlichkeit aus, dass

zwei Nachbarn eines Knotens ebenfalls benachbart sind. Im Gegensatz zur ersten Definition fallen hier Knoten mit niedrigen Graden weniger stark ins Gewicht [New03] (nach Watts und Strogatz kann ein Knoten mit Grad 2 beispielsweise nur $C_i = 0$ oder 1 haben). Für den Graphen aus Abbildung 1 b) ist $T = \frac{9}{14}$, da er drei Dreiecke und 14 verbundene Knotentripel aufweist.

Analog für den gerichteten Fall definiert Newman in [New03] die Transitivität als

$$T = \frac{6 \times \text{Anzahl Dreiecke im Graphen}}{\text{Anzahl Wege der Länge 2}}.$$

Für den Nenner zählen wir nun nicht mehr sämtliche Knotentripel, sondern stattdessen alle gerichtete Wege der Länge 2. Da ein Knotentripel nun doppelt so viele Kanten aufweisen kann wie im ungerichteten Fall, kommen jetzt maximal sechs solcher Wege auf ein Tripel, was den Faktor 6 in der Definition begründet. Das bedeutet allerdings auch, dass ein Dreieck hierbei nicht mehr als ein Kreis der Länge 3 aufgefasst werden kann, sondern als ein Knotentripel, welches die maximale Anzahl Kanten zueinander aufweist (also zwei Kreise der Länge 3). Um diese zu zählen, sind die Ansätze herkömmlicher Algorithmen zur Zählung von Dreiecken meist nicht mehr geeignet.

5.3.1.3 Reziprozität Vorangehend wurden zwei Ansätze vorgestellt, welche den Grad des Clusterings in einem Graphen zu erfassen versuchen und dafür die Dichte der Dreiecke betrachten. In gerichteten Graphen bietet es sich jedoch an, anstatt Kreisen der Länge 3 solche der Länge 2 zu betrachten. Wir untersuchen dabei also nicht mehr die Transitivität, sondern die Reziprozität (Wechselseitigkeit) eines Graphen. Für ungerichtete Graphen ist ein Knotenpaar *reziprok*, wenn zwischen den beiden Knoten zwei gerichtete Kanten (d.h. in beide Richtungen) existieren [CSW05]. Die *Reziprozität* eines gerichteten Graphen ist dann der Anteil aller durch mindestens eine Kante verbundenen Knotenpaare, die reziprok sind. Für die betrachteten sozialen Netzwerke wird also die Wahrscheinlichkeit untersucht, dass eine Freundschaft zwischen zwei Akteuren auf Gegenseitigkeit beruht. Die Berechnung der Reziprozität erfordert einen sequenziellen Durchlauf der Kantenliste, wobei für jede Kante untersucht wird, ob eine Kante in Rückrichtung besteht. Die Laufzeit ist also in $O(|E| \cdot \max(k))$ bzw. in $O(|E|)$, falls für die Kanten eines Knotens eine geeignete Hashstruktur verwendet wird.

5.3.2 Berechnung

Soll für jeden Knoten eines Graphen der lokale Clustering-Koeffizient bestimmt werden, so ist es für den Nenner der Definition erforderlich, den

(ausgehenden) Grad des jeweiligen Knotens zu bestimmen, um die maximal mögliche Kantenanzahl zwischen seinen Nachbarn zu berechnen. Dies kann einfach erreicht werden, indem die Kantenliste einmal sequenziell durchlaufen wird und dabei für jeden Knoten die Anzahl ausgehender Kanten gezählt wird. Für den Zähler hingegen gilt es, die Anzahl existierender Kanten zwischen den Nachbarn zu zählen bzw. im ungerichteten Fall die Anzahl Dreiecke zu zählen, an denen der jeweilige Knoten beteiligt ist. Methoden zur Bestimmung der Anzahl Dreiecke in einem Graphen widmen wir uns weiter unten.

Für die Transitivität im ungerichteten Fall gilt es, die Anzahl der verbundenen Tripel zu zählen. Wie beim lokalen Clustering-Koeffizienten genügt es dafür, die ausgehenden Grade aller Knoten zu bestimmen. Anhand dieser kann dann wieder die maximal mögliche Anzahl Kanten zwischen den Nachbarn des jeweiligen Knotens bestimmt werden. Diese kommt nämlich der Anzahl Tripel gleich, in denen der Knoten der mittlere ist. Anhand eines sequenziellen Durchlaufs der Kantenliste wird dabei, anstatt jeden Knoten einzeln zu betrachten, die Anzahl Tripel über alle Knoten aufsummiert. Die Transitivität für gerichtete Graphen ist dagegen komplexer zu bestimmen. Sie erfordert es, sämtliche Wege der Länge 2 zu zählen. Wenn man hierfür einen Breitendurchlauf für jeden Knoten in zwei Ebenen durchführt und die Summe der gefundenen Wege bildet, ist dies in $O(|V| \cdot |E|)$ Zeit möglich. Eine effizientere Möglichkeit ist es, jede Kante des Graphen sequenziell zu betrachten und jedes Mal den Grad des Zielknotens zur Gesamtsumme hinzuzuaddieren, was in $O(|E|)$ Zeit durchführbar ist, vorausgesetzt die Kanten eines Knotens sind in einem Array gespeichert oder es wird ein Zähler mitgeführt.

Während für den Clustering-Koeffizienten die Dreiecke *pro Knoten* gezählt werden, so erfordert die Transitivität eine Zählung der Gesamtanzahl Dreiecke des Graphen. Ein einfacher aber ineffizienter Ansatz für diese beiden Probleme besteht darin, alle möglichen Knotentripel des Graphen zu betrachten und zu zählen, wie oft alle drei Knoten miteinander verbunden sind [SW05b]. Die Laufzeit des Verfahrens ist allerdings in $O(|V|^3)$ und somit ungeeignet für große Graphen. Im Folgenden werden daher effizientere Methoden zur Zählung der Anzahl Dreiecke in einem Graphen vorgestellt.

5.3.2.1 Matrixmultiplikation Wie bereits in Abschnitt 5.2.2.2 erwähnt, gibt der Eintrag in der mit h potenzierten Adjazenzmatrix $A(G)$ eines Graphen G die Anzahl der Wege der Länge h zwischen den entsprechenden Knoten an. Wird daher $A(G)^3$ bestimmt, kann folglich in der Diagonalen der resultierenden Matrix die Anzahl der Wege der Länge 3 abgelesen werden, die die entsprechenden Knoten zu sich selbst haben [SW05a]. Für ungerichtete Graphen muss diese Anzahl noch halbiert werden, da ein Drei-

eck zwei solcher Wege aufweist. Auf diese Weise kann also die Anzahl Dreiecke bestimmt werden, an denen der entsprechende Knoten beteiligt ist. Um die Gesamtanzahl der Dreiecke des Graphen zu erhalten, muss die Summe aller gefundenen Dreiecke durch 3 dividiert werden, da ein Dreieck aus drei Knoten besteht. Da in der Definition der Transitivität jedoch die Anzahl Dreiecke wieder mit 3 multipliziert wird, kann der Wert hierfür auch direkt übernommen werden. Zur Erinnerung: Die Laufzeit der einfachen Matrixmultiplikation ist in $O(|V|^3)$ bzw. $O(|V|^{2.376})$ mit dem derzeit schnellsten bekannten Verfahren und der benötigte Speicher ist in $O(|V|^2)$.

5.3.2.2 Iteration über die Knoten Bei dieser Methode für ungerichtete Graphen, wird für jeden Knoten i überprüft, wie viele Kanten zwischen seinen Nachbarn existieren. Dies entspricht dann der Anzahl Dreiecke, an denen i beteiligt ist [SW05a]. Das Auffinden der Kanten kann anhand von zwei ineinander verschachtelten Schleifen über die Nachbarknoten realisiert werden. Für jedes mögliche Paar wird also überprüft, ob eine Kante zwischen ihm existiert. Die Gesamtanzahl Dreiecke erhält man wieder durch Division der Anzahl aller gefundenen Dreiecke durch 3. Wir wissen bereits, dass höchstens $k_i(k_i - 1)/2$ Kanten zwischen den Nachbarn für ungerichtete Graphen existieren können. Die Laufzeit für diese Methode ist daher in $O(|V| \cdot \max(k_i)^2)$, vorausgesetzt die Existenz einer Kante zwischen zwei Knoten kann in konstanter Zeit überprüft werden (z.B. mit einer Hashstruktur). Dies ist zwar theoretisch wieder in $O(|V|^3)$, wenn jedoch keine Knoten mit sehr hohen Graden im Netzwerk sind, zeigt sich das Verfahren meist deutlich schneller als die Matrixmultiplikation und benötigt vor allem viel weniger Speicherplatz.

Genauso wie für jeden Knoten beide Nachbarn betrachtet werden können, so ist es ebenso möglich, über die Kanten zu iterieren. Für jede Kante wird dabei überprüft, ob der Start- und Zielknoten einen gemeinsamen Nachbarn haben [SW05b]. Die Laufzeit ist entsprechend in $O(|E| \cdot \max(k_i)^2)$.

5.3.2.3 AYZ-Algorithmus Alon, Yuster und Zwick stellen in [AYZ97] ein schnelles exaktes Verfahren vor, um die Anzahl Dreiecke in einem Graphen zu bestimmen. Die Knoten werden dabei zunächst in zwei Gruppen aufgeteilt: Gruppe 1 enthält nur solche Knoten mit $k_i \leq \Delta$, wobei $\Delta = |E|^{\frac{\omega-1}{\omega+1}}$ mit dem Exponenten der Matrixmultiplikation ω, während Gruppe 2 entsprechend aus allen anderen Knoten gebildet wird. Zunächst werden die Dreiecke bestimmt, an denen die Knoten mit den kleineren Graden (Gruppe 1) beteiligt sind. Dazu werden sämtliche Pfade der Länge 2 (bzw. Tripel für ungerichtete Graphen) betrachtet, deren mittlerer Knoten aus Gruppe 1 stammt. Davon existieren höchstens $|E| \cdot \Delta$ viele, die in $O(|E| \cdot \Delta)$ Zeit gefunden werden können, indem für jede Kante überprüft wird, ob der

Zielknoten mehr als Δ Kanten aufweist. Für jeden solchen Pfad wird dann in konstanter Zeit nachgeschaut, ob eine Kante vom Endknoten zum Startknoten des Pfades verläuft und damit ein Dreieck bildet. Die Laufzeit für diesen Schritt ist also in $O(|E|^{1.5})$ für die naive Matrixmultiplikation und entsprechend in $O(|E|^{1.41})$ für $\omega \approx 2.38$. Nach diesem Schritt müssen noch solche Dreiecke gefunden werden, die nur aus Knoten mit hohen Graden (Gruppe 2) bestehen. Da es höchstens $|E|/\Delta$ solcher Knoten gibt (bzw. doppelt so viele im ungerichteten Fall), können Dreiecke zwischen diesen dann per Matrixmultiplikation in $O((|E|/\Delta)^{\omega})$ Zeit und $O((|E|/\Delta)^2)$ Speicher gefunden werden. Das bedeudet, die Laufzeit für diesen Schritt ist auch hier für $\omega = 3$ in $O(|E|^{1.5})$ und für $\omega \approx 2.38$ in $O(|E|^{1.41})$. Die Gesamtlaufzeit des Algorithmus ist folglich in

$$O(|E| \cdot \Delta + (|E|/\Delta)^{\omega}) = O(|E|^{\frac{2\omega}{\omega+1}}).$$

Für die Gesamtanzahl Dreiecke im Graphen muss die Summe der in Schritt 1 und Schritt 2 gefundenen Dreiecke wieder durch 3 dividiert werden, da wir im ersten Schritt für jeden Pfad der Länge 2 ein Dreieck zählen, wenn er Bestandteil eines solchen ist und ein Dreieck aus drei solcher Pfade besteht. Ähnlich ist es in Schritt 2, nur zählen wir dort Dreiecke pro Knoten.

5.3.2.4 Approximative Berechnung Schank und Wagner stellen in [SW05a] ein approximatives Verfahren zur Berechnung des gewichteten Clustering-Koeffizienten in ungerichteten Graphen vor, welches auf Sampling beruht. Hierbei wird jedem Knoten v ein Gewicht $w(v)$ zugesprochen, damit die Möglichkeit besteht, bestimmte Knoten stärker zu bewerten als andere. Der *gewichtete Clustering-Koeffizient* ist dann

$$C^w = \frac{1}{\sum_{v \in V} w(v)} \cdot \sum_{v \in V} w(v) \cdot C_v.$$

Die für den Clustering-Koeffizienten gewünschte Eigenschaft, dass nur Werte zwischen 0 und 1 angenommen werden, bleibt hierbei erhalten. Für das Gewicht eines Knotens eignet sich zum Beispiel der Grad des Knotens oder die Anzahl möglicher Kanten zwischen seinen Nachbarn. Setzt man letztere in die Definition an Stelle von $w(v)$ ein, so erhält man

$$\begin{aligned} C^w &= \frac{1}{\sum_{v \in V} k_v(k_v-1)/2} \cdot \sum_{v \in V} \frac{k_v(k_v-1)/2 \cdot n_v}{k_v(k_v-1)/2} \\ &= \frac{\sum_{v \in V} n_v}{\sum_{v \in V} k_v(k_v-1)/2}, \end{aligned}$$

was genau der Definition der Transitivität entspricht. Diese kann also als ein Spezialfall des gewichteten Clustering-Koeffizienten aufgefasst werden. Der Algorithmus ist in Abbildung 10 dargestellt. In der ersten Schleife

```
C^w-Approximation(Array von Knoten v ∈ V mit d(v) ≥ 2,
                  Adjazenzarray für jeden Knoten,
                  Gewichtungsfunktion w,
                  Anzahl t der Samples)
Gewichtsumme = 0
for each v ∈ V do
    Gewichtsumme = Gewichtsumme + w(v)
    W[v] = Gewichtsumme
l = 0
for each i ∈ (1, ..., t) do
    rand = Zufallszahl aus [0, Gewichtsumme]
    x = Suche Knoten mit W[x] ≥ rand und W[x] ≤ W[u] für
        alle Knoten u ∈ V mit W[u] ≥ rand
    y = Zufälliger adjazenter Knoten zu x
    repeat
        z = Zufälliger adjazenter Knoten zu x
    until y ≠ z
    if KanteExistiert(y,z) then
      l = l + 1
return l/k
```

Abbildung 10: Algorithmus zur Approximation von C^w.

wird ein Array W erzeugt, welches die kumulativen Gewichte der einzelnen Knoten enthält. Danach werden t Tripel herausgegriffen und es wird für jedes überprüft, ob eine Kante zwischen seinen beiden äußeren Knoten existiert. Die Wahrscheinlichkeit für ein Tripel, ausgewählt zu werden, ist dabei abhängig von dem Gewicht seines mittleren Knotens. Am Ende gibt der Algorithmus schließlich den Anteil aller betrachteten Tripel aus, die Dreiecke bilden.

Was die Laufzeit des Algorithmus betrifft, so ist die der ersten Schleife abhängig von der verwendeten Gewichtungsfunktion. Sie ist demnach in $O(g(|V|))$, wenn $g(|V|)$ die Laufzeit im schlechtesten Fall für die Bestimmung sämtlicher Knotengewichte ist. Für die zweite Schleife wird aufgrund der Array-Struktur für die Knoten angenommen, dass das Suchen eines zufälligen Knotens in konstanter Zeit möglich ist. Dasselbe gilt für das Nachweisen einer Kante zwischen zwei Knoten, vorausgesetzt es wird eine Hashstruktur für die Kanten verwendet. Lediglich die benötigten Schritte für die Suche nach dem richtigen Feldeintrag in W ist abhängig von der Knotenanzahl. Mit binärer Suche ist diese in $O(\ln |V|)$ möglich [SW05a]. Es ergibt sich somit eine Gesamtlaufzeit von $O(g(|V|) + k \cdot \ln |V|)$.

Netzwerk	R	C	T
A	0.992	0.3239	0.0018
B	0.917	0.1540	0.0725
C	0.985	0.1707	0.0079
D	0.947	0.1618	0.0037

Tabelle 6: Reziprozität R sowie Clustering-Koeffizient C und Transitivität T für die ungerichteten Graphen der sozialen Netzwerke.

Für die Transitivität gilt $g(|V|) = |E|$, somit ist sie in $O(|E| + k \cdot \ln |V|)$ Zeit approximierbar.

Für den ungewichteten Clustering-Koeffizienten kann der Algorithmus leicht modifiziert werden, indem zunächst die erste Schleife gänzlich ignoriert wird. Außerdem ist die Wahl des Knotens x nun nicht mehr abhängig von seinem Gewicht, sondern völlig zufällig. Dies hat zur Folge, dass ein direkter Zugriff in konstanter Zeit auf den Knoten im Array möglich ist. Die Laufzeit für den gesamten Algorithmus ist daher nur noch in $O(k)$ und somit konstant bezüglich der Anzahl Knoten und Kanten im Graph [SW05a].

5.3.3 Eigene Ergebnisse und Bewertung

Es wurden nun zum einen die Berechnung der Reziprozität und zum anderen eine Funktion zur Bestimmung des exakten Clustering-Koeffizienten bzw. der Transitivität implementiert. Die Definition letzterer im ungerichteten Fall erfordert es, Knotentripel mit sechs gerichteten Kanten zu zählen. Da jedoch keiner der vorgestellten Algorithmen für die Zählung solcher „doppelten“ Dreiecke geeignet ist, wurden die Graphen vorher in ungerichtete Graphen konvertiert. Eine ungerichtete Kante wurde immer dann zwischen zwei Knoten eingefügt, wenn mindestens eine gerichtete Kante zwischen ihnen existiert. Dies bringt einen Verlust an Genauigkeit mit sich, dessen Ausmaß sich aber aufgrund der hohen Reziprozität (siehe Tabelle 6) im Graphen gering zeigt. Das Konvertieren ist in $O(|E| \cdot \max(k_i))$ Zeit möglich, indem jede Kante $u \to v$ des gerichteten Graphen nacheinander betrachtet wird und die Kanten $u \to v$ sowie $v \to u$ der ungerichteten Version hinzugefügt werden, wenn sie nicht bereits existieren. Diese Überprüfung kann jedoch in konstanter Zeit geschehen, wenn eine Hashstruktur für die Kanten eines Knotens verwendet wird. Die Laufzeit ist dann nur noch in $O(|E|)$.

Bei der Implementierung wurde nun auf die Methode zurückgegriffen, welche über die Knoten iteriert (siehe Abschnitt 5.3.2.2). Das Programm

wurde dann auf die bereinigten Datensätze der sozialen Netzwerke A bis D angewendet, denn anhand der unbereinigten Kantenliste lässt sich kein gültiger Graph generieren. Tabelle 6 gibt einen Überblick über die erzielten Resultate. Mit einem Clustering-Koeffizienten von ca. 0.32 ist Netzwerk A mit Abstand das dichteste Netzwerk. Das bedeutet, dass hier im Durchschnitt ca. 32% der Nachbarn eines Knotens miteinander verbunden sind, oder anders formuliert: Mit einer mittleren Wahrscheinlichkeit von ca. 32% sind zwei Individuen, die einen gemeinsamen Freund haben, ebenfalls miteinander befreundet. Auffällig zeigt sich, dass trotz der unterschiedlichen Größen der anderen Netzwerke, sie alle einen fast identischen Clustering-Koeffizienten haben (zwischen 0.15 und 0.18). Obwohl Netzwerk A den größten Clustering-Koeffizienten aufweist, so besitzt es doch die kleinste Transitivität. Diese ist bei Netzwerk B hingegen mit ca. 0.07 deutlich am größten. Die Wahrscheinlichkeit beträgt hier also ca. 7%, dass für einen Akteur der Freund eines Freundes auch sein Freund ist. Bei allen anderen Netzwerken liegt sie unter 1%. Die Tatsache, dass die Transitivität viel geringer als der Clustering-Koeffizient ausfällt, ist wahrscheinlich auf die hohe Anzahl Knoten mit niedrigen Graden (jedoch größer als 1) zurückzuführen, welche Kanten zwischen ihren Nachbarn haben. Diese haben nämlich häufig einen Clustering-Koeffizienten von 1 und fallen im Mittel folglich stark ins Gewicht.

5.3.4 Vergleich mit anderen Ergebnissen

Watts und Strogatz zeigen für das Schauspieler-Netzwerk (siehe Abschnitt 5.2.4) einen Clustering-Koeffizienten von 0.79 auf [WS98]. Selbiges Netzwerk hat nach [New03] eine Transitivität von 0.2. Im gerichteten Graphen des E-Mail-Netzwerks dagegen wurde ein Clustering-Koeffizient von ca. 0.17 sowie eine Reziprozität von 0.231 herausgestellt [NFB02]. Zuletzt wurden in wissenschaftlichen Kollaborationsnetzwerken Werte von $C = 0.6$ und $T = 0.088$ für die Biologie, $C = 0.56$ und $T = 0.45$ für die Physik sowie $C = 0.34$ und $T = 0.15$ für die Mathematik nachgewiesen [New03]. Ein ähnliches Phänomen wie für unsere sozialen Netzwerke lässt sich für diese Netzwerke beobachten: Das mit Abstand größte Netzwerk der Biologie ($|V| = 1,520,251$) hat zwar den größten Clustering-Koeffizienten, dafür jedoch die kleinste Transitivität.

5.3.5 Ausblick

Neben Kreisen der Länge 2 (Reziprozität) und solchen der Länge 3 (Transitivität), lassen sich genauso gut noch längere Kreise in einem Graphen betrachten. Dabei geht es dann allerdings weniger darum, Aussagen über das Ausmaß des Clusterings zu machen. Stattdessen ist dies z.B. dann interessant, wenn es eine Beziehung zwischen der Anzahl Kreise der Länge

h und der Netzwerkgröße $|V|$ zu untersuchen gilt [BC02].

Dorogovtsev et al. untersuchen in [DGM02] den Clustering-Koeffizienten eines Knotens in Abhängigkeit seines Grads und kommen zu dem Ergebnis, dass für bestimmte Modelle von skaleninvarianten Netzwerken eine Power-Law-Relation zwischen diesen beiden Größen besteht: $C_i \approx k_i^{-1}$. Auch in großen realen Netzwerken, wie z.B. dem Schauspielernetzwerk, konnte eine solche Beziehung nachgewiesen werden [RB03]. Ob diese auch für die hier behandelten sozialen Netzwerke gilt, bleibt Gegenstand zukünftiger Analysen. Ähnlich wie in Abschnitt 5.1 lässt sich dies untersuchen, indem die beiden Größen in einem Log-Log-Plot gegeneinander aufgetragen und auf Linearität (anhand einer Regressionsgeraden mit Steigung 1) untersucht werden.

Ein ähnliches Phänomen, welches eine nähere Betrachtung nahe legt, ist, dass für viele reale Netzwerke angenommen wird, dass für $|V| \to \infty$ der Clustering-Koeffizient gegen eine konstante positive Zahl strebt und sich somit unabhängig von der Größe des Netzwerks zeigt [New03]. Wir werden im nachfolgenden Kapitel sehen, dass bei ausreichend hoher Knotenzahl der Clustering-Koeffizient daher in vielen Netzwerken der realen Welt zwangsläufig größer als in zufällig generierten Graphen derselben Größe ist.

6 Vergleich mit zufälligen Graphen

Es wurden bisher Eigenschaften von Graphen wie der Durchmesser und der Clustering-Koeffizient vorgestellt und experimentell für die sozialen Netzwerke bestimmt. Dabei wurden die Ergebnisse zum einen subjektiv und zum anderen anhand von Vergleichen zu anderen Studien bewertet. In diesem Kapitel soll es darum gehen, die betrachteten Größen in den entsprechenden zufälligen Graphen zu untersuchen, um folglich die getätigten Aussagen auch objektiv unterstreichen zu können. Wir werden dazu zunächst zufällige Graphen definieren und dann ihre Eigenschaften mit den in Kapitel 5 für die sozialen Netzwerke gewonnenen vergleichen.

6.1 Das Erdös-Rényi-Graphmodell

Wenn in der Literatur von „zufälligen“ Graphen gesprochen wird, so ist häufig jenes Modell gemeint, welches 1951 zum ersten Mal von Solomonoff und Rapoport, knapp zehn Jahre später dann unabhängig von Erdös und Rényi genauer analysiert wurde [New03]. Es ist daher auch als Erdös-Rényi-Graphmodell bekannt, man spricht jedoch auch häufig von einem Poisson- oder Bernoulli-Graphen. Die Konstruktion des Graphen ist einfach: Ausgehend von $|V|$ Knoten ohne Kanten wird für jedes mögliche Knotenpaar mit Wahrscheinlichkeit p eine Kante hinzugefügt [CF06]. Folg-

lich ist der mittlere Knotengrad z für diesen Graphen offensichtlich:

$$z = p \cdot |V|.$$

Wir werden im Folgenden die Eigenschaften des Modells untersuchen. Es sei jedoch erwähnt, dass die meisten der Aussagen erst für zufällige Graphen mit hinreichend großer Knotenanzahl zutreffen. Außerdem ist das Graphmodell nur für ungerichtete Graphen definiert. Um daher einen sinnvollen Vergleich mit den sozialen Netzwerken anstellen zu können, werden die ungerichteten Graphen dieser betrachtet, welche nach der in Abschnitt 5.3.3 beschriebenen Vorgehensweise erzeugt wurden und in ihren Eigenschaften aufgrund der hohen Reziprozität keine merklichen Unterschiede zu den gerichteten Versionen aufweisen. Tabelle 7 gibt einen Überblick über die Kantenanzahl und den mittleren Knotengrad der ungerichteten Graphen der sozialen Netzwerke.

Netzwerk	n	m	z
A	24,142	112,865	9.35
B	41,005	97,813	4.77
C	48,602	154,523	6.36
D	200,000	925,120	9.25

Tabelle 7: Eigenschaften der ungerichteten Graphen der sozialen Netzwerke: Anzahl gültiger Knoten n, Anzahl gültiger Kanten m und durchschnittlicher Knotengrad z.

6.2 Vergleich der Eigenschaften

6.2.1 Verteilung des Knotengrads

Die Wahrscheinlichkeit für einen Knoten, Grad k zu besitzen, beträgt im Erdös-Rényi-Graphen ([CF06]):

$$p(k) = \binom{|V|}{k} \cdot p^k (1-p)^{|V|-k}.$$

Der Knotengrad ist also binomialverteilt, was dadurch begründet werden kann, dass für einen Knoten die Kantenerzeugung $|V|$ Bernoulli-Prozessen zu Grunde liegt, von denen die Wahrscheinlichkeit ermittelt werden soll, dass genau k Erfolge erzielt wurden. Zudem ist die Existenz von Kanten statistisch unabhängig. Für große Knotenanzahl $|V|$ lässt die Verteilung sich durch eine Poisson-Verteilung approximieren ([New03]):

$$p(k) \approx \frac{z^k e^{-z}}{k!}.$$

Sie ist der Grund, warum das Modell auch Poisson-Graphmodell genannt wird. Anders als Graphen mit Power-Law-Verteilung des Knotengrads (siehe Abbildung 4 a), sind Erdös-Rényi-Graphen nicht skaleninvariant. Abbildung 11 zeigt zur Veranschaulichung einen zufälligen Graphen sowie einen Graphen mit den gleichen Werten für $|V|$ und z, in dem der Knotengrad wie bei den sozialen Netzwerken B, C und D einer Power-Law-Verteilung zu Grunde liegt.

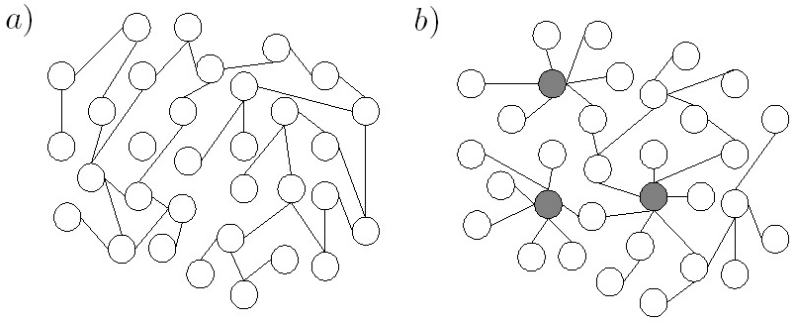

Abbildung 11: Beispielgraphen mit $|V| = |E| = 32$: a) Erdös-Rényi-Graph mit Poisson-verteiltem Knotengrad. b) Graph mit Power-Law-Verteilung des Knotengrads [Cas04].

6.2.2 Durchmesser

Betrachten wir den Durchmesser, beträgt die mittlere Anzahl Knoten, die d Hops von einem Knoten entfernt sind, im zufälligen Graphen z^d [New03]. Um demnach die Anzahl Hops zu bestimmen, die ein typischer Weg durch das Netzwerk benötigt, so suchen wir den Wert für d, mit dem das ganze Netzwerk umspannt wird, und erhalten so eine Approximation für den mittleren Durchmesser:

$$z^{d_{avg}} \approx |V|$$

$$d_{avg} \approx \log |V| / \log z.$$

Dieser wächst also mit dem Logarithmus der Knotenanzahl und ist demnach selbst in großen Netzwerken gering. Ähnliches gilt für den effektiven Durchmesser, der jedoch immer etwas höher ausfällt als d_{avg}. Tabelle 8 stellt nun die mittleren Durchmesser der sozialen Netzwerke denen gleich großer zufälliger Graphen mit demselben mittleren Knotengrad gegenüber. Es zeigt sich, dass die gemessenen Durchmesser ähnlich gering sind wie die der zufälligen Graphen. Für Netzwerke A, B und D ist er sogar noch kleiner. Wir können jedoch festhalten, dass zwischen den betrachteten sozialen Netzwerken und den zufälligen Graphen kein merkbarer Unterschied im Durchmesser besteht und beide den Small-World-Effekt aufweisen.

Netzwerk	d_{avg}	d_{avg}^{rand}	C	C^{rand}
A	2.69	4.52	0.3239	0.00039
B	5.54	6.80	0.1540	0.00012
C	3.18	2.10	0.1707	0.00013
D	3.00	5.49	0.1618	0.00005

Tabelle 8: Mittlerer Durchmesser und Clustering-Koeffizient für die ungerichteten Graphen der sozialen Netzwerke sowie den entsprechenden Erdös-Rényi-Graphen mit gleicher Knotenanzahl und gleichem mittleren Knotengrad. Für Netzwerk D wird für den Durchmesser der in Abschnitt 5.2.3 angenommene Wert von 3 verwendet.

6.2.3 Clustering-Koeffizient

Der Clustering-Koeffizient stellt die Wahrscheinlichkeit dar, dass zwei Nachbarn eines Knotens ebenfalls benachbart sind (siehe Abschnitt 5.3). Diese ist in zufälligen Graphen jedoch unabhängig von der Anzahl Nachbarn des Knotens, denn die Wahrscheinlichkeit, dass zwei Knoten miteinander verbunden sind, beträgt im Erdös-Rényi-Graphen immer p [CF06]. Dies entspricht wiederum $\frac{z}{|V|}$, somit gilt:

$$C = \frac{z}{|V|}.$$

Bei wachsender Knotenanzahl und konstantem mittleren Knotengrad strebt der Clustering-Koeffizient (und die Transitivität) demnach gegen 0, während er, wie bereits in Abschnitt 5.3.5 angedeutet, für viele reale Netzwerke unabhängig von ihrer Größe gegen eine konstante positive Zahl strebt [New03]. Einen Vergleich der Clustering-Koeffizienten der sozialen Netzwerke mit solchen der zufälligen Graphen zeigt Tabelle 8. Sie bestätigt die Vermutung: Der Clustering-Koeffizient ist in zufälligen Graphen äußerst niedrig (nahe 0) und somit viel kleiner als in den sozialen Netzwerken. Die These der stark erhöhten Community-Bildung in den betrachteten Netzwerken kann also bestätigt werden.

7 Einordnung in das Small-World-Modell

Abschließend wollen wir auf das Netzwerkmodell von Watts und Strogatz eingehen, welches die Verifikation von Netzwerken bezüglich einer Small-World-Natur ermöglicht. Watts und Strogatz untersuchten im Jahr 1998 den Clustering-Koeffizienten sowie den mittleren Durchmesser in Netzwerken mit verschiedenen Graden der Unordnung. Auf Basis ihrer Untersuchungen definieren sie in [WS98] Small-World-Netzwerke.

Wir beschreiben zunächst das ursprüngliche Netzwerkmodell und erläutern dann im Anschluss eine modifizierte Variante des Modells, welche 1999 von Monasson sowie von Newman und Watts jeweils unabhängig voneinander vorgeschlagen wurde [New03]. Danach werden die Eigenschaften des Modells behandelt, welche auf das Hauptergebnis von Watts und Strogatz hinführen. Schließlich folgt eine Einschätzung der sozialen Netzwerke bezüglich der Einordnung in das Small-World-Modell.

7.1 Ursprüngliches Netzwerkmodell

Watts und Strogatz betrachten in ihrem Modell einen ungerichteten Graphen mit geographischer Komponente. Sie gehen folglich davon aus, dass diese bei der Frage, welche Knoten miteinander verbunden sind, von Bedeutung ist. Der Graph besteht zu Beginn aus $|V|$ ringförmig angeordneten Knoten, die alle zu ihren k (geographisch) nächsten Nachbarn auf jeder Seite mit ungerichteten Kanten verbunden sind (siehe linker Graph in Abbildung 12). Jeder Knoten hat demnach denselben Grad und der Graph folglich den mittleren Knotengrad:

$$z = 2k.$$

Auf Basis dieses Ausgangsgraphen wird nun ein Zufallsgraph erzeugt,

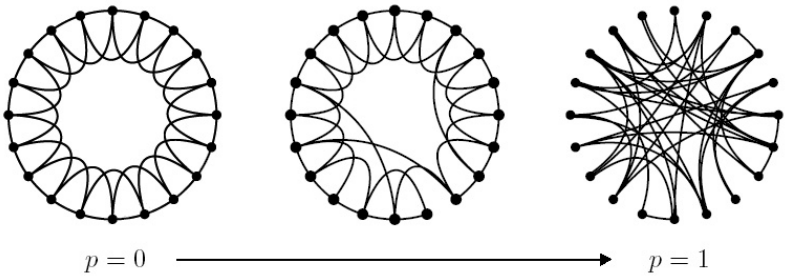

Abbildung 12: Graphen mit $|V| = 20$, $k = 2$ und verschiedener Wahrscheinlichkeit einer Kantenrestrukturierung p nach dem Netzwerkmodell von Watts und Strogatz [WS98].

indem Kanten zufällig umstrukturiert werden: Im Uhrzeigersinn werden alle Knoten nacheinander betrachtet und die Kante zum nächsten Nachbarn des jeweiligen Knotens wird mit Wahrscheinlichkeit p so umgeordnet, dass das Ende der Kante auf der Seite des Nachbarn zu einem zufällig ausgewählten Knoten umgelegt wird. Sollte zu diesem schon eine Kante bestehen, wird der Vorgang entsprechend oft wiederholt. Es sind zudem keine Kanten eines Knotens zu sich selbst erlaubt. Mit Wahrscheinlichkeit $1-p$ hingegen bleibt die Kante an ihrem Platz. Im nächsten Durchlauf wird

dann die Kante zum zweitnächsten Nachbarn betrachtet und nach demselben Schema behandelt, bis alle Kanten einmal betrachtet wurden. Da es $|V| \cdot k$ Kanten im Graphen gibt, endet die Prozedur nach k Durchläufen. Es fällt dabei auf, dass für wachsendes p das Netzwerk immer mehr Unordnung aufweist. Während für $p = 0$ das Netzwerk noch eine reguläre Struktur hat, so wurden für $p = 1$ alle Kanten völlig zufällig umgeordnet (rechter Graph in Abbildung 12). Der mittlere Knotengrad von $2k$ bleibt dabei für verschiedene Werte von p erhalten, da die Kanten nur umstrukturiert werden. Die Autoren fordern zudem, dass k so gewählt wird, dass $2k \gg ln(|V|) \gg 1$ gilt, denn dadurch wird garantiert, dass der Graph auch für $p = 1$ zusammenhängend bleibt.

7.2 Modifikation des Modells

Wie bereits erwähnt, ist das ursprüngliche Modell für $k = 1$ nicht definiert, da es hierbei wahrscheinlich ist, dass der Graph durch die Restrukturierung unzusammenhängend wird. Dies hat zur Folge, dass es unendliche Abstände zwischen Knoten gibt und demnach Größen wie der effektive Durchmesser, der ja das Mittel der kürzesten Abstände sämtlicher Knotenpaare ist, ebenfalls unendlich werden. Daher schlagen Newman und Watts in [NW99] einen anderen Ansatz vor, der auch für $k = 1$ definiert ist und für $k \geq 2$ bei hinreichend großer Knotenzahl annähernd den gleichen mittleren Durchmesser wie das ursprüngliche Modell aufweist. Dies ist möglich, indem die bestehenden Kanten nicht umstrukturiert werden, sondern stattdessen lediglich Kanten hinzugefügt werden. Der Graph kann folglich nicht unzusammenhängend werden. Es wird also für jede Kante des Ausgangsgraphen mit Wahrscheinlichkeit p eine zusätzliche Kante (im Folgenden mit „Abkürzung“ bezeichnet) zwischen zwei zufällig ausgewählten Knoten hinzugefügt (mehrere Kanten zwischen zwei Knoten und Kanten von Knoten zu sich selbst sind hier aus mathematischen Gründen erlaubt). Die mittlere Anzahl von Abkürzungen ist somit $|V| \cdot k \cdot p$, also für alle $|V| \cdot k$ Kanten des Graphen die entsprechende Wahrscheinlichkeit, dass eine zusätzliche Kante hinzugefügt wird. Für jede der Abkürzungen beträgt die Wahrscheinlichkeit für einen Knoten hingegen $\frac{2}{|V|}$, dass er diese Abkürzung berührt. Die mittlere Anzahl Abkürzungen, die mit einem Knoten verbunden sind, ist folglich $|V| \cdot k \cdot p \cdot \frac{2}{|V|} = 2kp$. Der mittlere Knotengrad z in diesem Modell beträgt

$$z = 2k + 2kp,$$

was dadurch einzusehen ist, dass jeder Knoten zu den $2k$ Anfangskanten zuzüglich den $2kp$ mit ihm verbundenen Abkürzungen inzident ist.

7.3 Eigenschaften

Im Folgenden sollen Eigenschaften für das Modell von Watts und Strogatz sowie für das alternative Modell von Newman und Watts herausgearbeitet werden. Dabei gilt es, diese in Hinblick auf verschiedene Werte von p zu untersuchen, also sowohl für den regulären als auch für den zufälligen Fall. Wir betrachten wieder die Verteilung des Knotengrads, den Durchmesser sowie den Clustering-Koeffizienten. Die Art und Weise der Kantenrestrukturierung lässt dabei vermuten, dass für $p = 1$ der Graph ähnliche Eigenschaften aufweist wie der entsprechende Erdös-Rényi-Graph (siehe Abschnitt 6.1).

7.3.1 Verteilung des Knotengrads

Für das alternative Modell gilt, dass jeder Knoten einen Grad von mindestens $2k$ hat, denn die Anfangskanten bleiben unverändert. Dementsprechend ist die Wahrscheinlichkeit für einen Knoten, Grad j zu besitzen, $p(j) = 0$ für $j < 2k$. Zu diesen $2k$ Kanten kommen nun noch die zusätzlich eingefügten Kanten, die wie im Erdös-Rényi-Graphen binomialverteilt sind. Wir betrachten also die Wahrscheinlichkeit, aus $|V|$ Versuchen $j - 2k$ Erfolge zu erzielen. Die Wahrscheinlichkeit, dass für einen Knoten ein Shortcut existiert, ist gleich der mittleren Anzahl der mit ihm verbundenen Shortcuts geteilt durch die maximale mögliche Anzahl Kanten des Knotens und demnach $\frac{2kp}{|V|}$. Für $j \geq 2k$ ist daher nach [New03]

$$p(j) = \binom{|V|}{j-2k} \cdot \left[\frac{2kp}{|V|}\right]^{j-2k} \left[1 - \frac{2kp}{|V|}\right]^{|V|-j+2k}.$$

Für das Modell von Watts und Strogatz gilt, dass aufgrund der Tatsachen, dass immer die Kante zum nächsten Nachbarn umstrukturiert wird und lediglich k Durchläufe getätigt werden, jeder Knoten im resultierenden Graphen einen Grad von mindestens k hat. Demnach ist $p(j) = 0$ für $j < k$. Für $j \geq k$ stellen Barrat und Weigt in [BW00] die folgende Wahrscheinlichkeit heraus:

$$p(j) = \sum_{n=0}^{\min(j-k,k)} \binom{k}{n} (1-p)^n \cdot p^{k-n} \frac{(pk)^{j-k-n}}{(j-k-n)!} \cdot e^{-pk}.$$

Es wird hier das Produkt der Wahrscheinlichkeit für n (mit Wahrscheinlichkeit $1-p$) unberührte Kanten eines Knotens mit der Wahrscheinlichkeit für die $j-k-n$ restlichen Kanten des Knotens betrachtet, die (mit Wahrscheinlichkeit $\frac{p}{|V|}$) zu diesem Knoten umstrukturiert wurden. Erstere sind dabei binomialverteilt und letztere (für großes $|V|$) Poisson-verteilt. Das Produkt wird dann für die verschiedenen Möglichkeiten des Verhältnisses zwischen unberührten und umstrukturierten Kanten aufsummiert, wobei die mögliche Anzahl unberührter Kanten von 0 bis maximal k reicht.

7.3.2 Durchmesser

Betrachten wir den Durchmesser, so kann in beiden Modellen im regulären Fall ($p = 0$) nur über die äußeren Kanten im Ring gelaufen werden, d.h. maximal in Sprüngen der Größe k. Der Durchmesser steigt daher linear mit der Netzwerkgröße $|V|$ und nähert sich $\frac{|V|}{4k}$ an für großes $|V|$ [New03]. Für ein Netzwerk der Größe $|V| = 10^5$, in dem jeder Knoten mit seinen 20 nächsten Nachbarn verbunden ist, wäre demnach $d_{avg} \approx 2,500$. Der Small-World-Effekt ist für das Modell im Fall $p = 0$ also nicht gegeben. Für $p = 1$ hingegen verhält sich der Durchmesser in beiden Modellen ähnlich wie im Erdös-Rényi-Graphen: Die mittlere Anzahl Knoten mit d Hops Abstand zu einem Knoten beträgt z^d, somit ist der Durchmesser auch hier ca. $\log |V| / \log z$ und wächst logarithmisch mit der Netzwerkgröße. Es kann also festgehalten werden, dass für $p = 0$ der Durchmesser linear mit der Knotenanzahl wächst, für $p = 1$ er jedoch nur logarithmisch ansteigt und den Small-World-Effekt zeigt. Eine exakte Berechnung von d_{avg} in Abhängigkeit von p wurde allerdings bisher noch nicht gefunden [NW99]. Abbildung 13, welche den mittleren Durchmesser für $0 \leq p \leq 1$ zeigt, basiert daher auf eigenen experimentellen Ergebnissen: Es wurde dazu ein Generator implementiert, der Graphen nach dem Netzwerkmodell von Watts und Strogatz erzeugt und als Eingabe die Parameter $|V|$, k und p entgegennimmt. Für die Abbildung wurden dann die Durchschnittswerte aus mehreren Durchläufen aufgetragen.

7.3.3 Clustering-Koeffizient

Betrachten wir den Clustering-Koeffizienten für das reguläre Netzwerk ($p = 0$) im ursprünglichen Modell, so existieren $\frac{3k(k-1)}{2}$ Kanten zwischen den Nachbarn eines Knotens [BW00]. Da maximal $\frac{2k(2k-1)}{2}$ Kanten möglich wären, ergibt sich somit aus der Division ein Clustering-Koeffizient von $C = \frac{3k-3}{4k-2}$, der für $k \to \infty$ gegen $\frac{3}{4}$ strebt. Für das zufällige Netzwerk ($p = 1$) hingegen ist er sehr klein und beträgt $C \approx 2k/|V|$ (wie im Erdös-Rényi-Graphen). Für $|V| \to \infty$ strebt er also bei konstantem mittleren Knotengrad gegen 0. Barrath gibt in [BW00] eine Formel für den Clustering-Koeffizienten in Abhängigkeit von p an. Sie basiert darauf, dass die Wahrscheinlichkeit $(1-p)^3$ beträgt, dass zwei miteinander verbundene Nachbarn eines Knotens nach der Umstrukturierung immer noch verbunden sowie Nachbarn desselben Knotens sind, da hierfür drei Kanten unberührt bleiben müssten. Die mittlere Anzahl Kanten zwischen den Nachbarn eines Knotens entspricht dann ca. $\frac{3k(k-1)}{2} \cdot (1-p)^3$ und der Clustering-Koeffizient demnach

$$C \approx \frac{3k-3}{4k-2} \cdot (1-p)^3$$

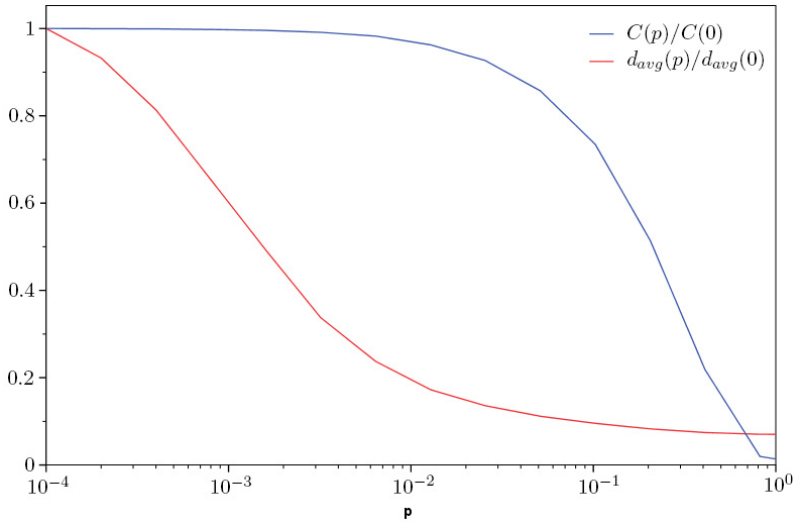

Abbildung 13: Clustering-Koeffizient C und mittlerer Durchmesser d_{avg} für unterschiedliche Wahrscheinlichkeiten p der Kantenrestrukturierung nach dem Small-World-Modell. Die geschätzten Funktionen basieren auf Mittelwerten aus 20 generierten Graphen mit den Eigenschaften $|V| = 1000$ und $k = 5$. Die Größen wurden zudem normalisiert, um sie miteinander vergleichbar zu machen.

für $0 \leq p < 1$. Laut dieser Definition stellt er nun nicht mehr den Mittelwert des Verhältnisses zwischen den Kanten unter den Nachbarn und den maximal möglichen Kanten dar, sondern stattdessen das Verhältnis zwischen der mittleren Kantenanzahl unter den Nachbarn und der mittleren maximal möglichen Anzahl. Die Kurve für verschiedene Werte von p ist für beide Definitionen jedoch fast gleich. Die Abweichung ist lediglich in $O(\frac{1}{|V|})$ und strebt somit gegen 0 für große Graphen [BW00]. Abbildung 13 zeigt den Übergang vom hohen Clustering-Koeffizienten in regulären Netzwerken hin zum kleinen in zufälligen Netzwerken. Es sei angemerkt, dass die Transitivität eine fast identische Kurve aufweist.

Für das alternative Modell, bei dem die Ausgangskanten unverändert bleiben, stellt Newman in [New03] die folgende Formel auf:

$$C = \frac{3(k-1)}{2(2k-1) + 4kp(p+2)}.$$

7.4 Ergebnisse

Es wurden nun der Clustering-Koeffizienten sowie der mittlere Durchmesser für das Modell betrachtet. In Abbildung 13 ist zu sehen, dass für $p = 0$ beide am größten sind und sie für $p = 1$ am geringsten ausfallen. Das

Netzwerk	d_{avg}	d_{avg}^{rand}	d_{avg}^{reg}	C	C^{rand}	C^{reg}
A	2.69	4.52	1291	0.3239	0.00039	0.66
B	5.54	6.80	4298	0.1540	0.00012	0.55
C	3.18	2.10	3821	0.1707	0.00013	0.61
D	3.00	5.49	10811	0.1618	0.00005	0.66

Tabelle 9: Durchmesser und Clustering-Koeffizient für die ungerichteten Graphen der sozialen Netzwerke sowie für reguläre ($p = 0$) und zufällige ($p = 1$) Graphen mit gleicher Knotenanzahl sowie gleichem mittleren Knotengrad nach dem Modell von Watts uns Strogatz. Für Netzwerk D wird für den Durchmesser der in Abschnitt 5.2.3 angenommene Wert von 3 verwendet.

Hauptergebnis von Watts und Strogatz ist nun aber, dass für verschiedene Werte von p die beiden Größen nicht immer einher gehen. Stattdessen nimmt C für einen großen Wertebereich von p kaum ab und erst bei $p \approx 0.1$ beginnt es zu fallen. d_{avg} hingegen fällt schon für viel kleinere Werten für p. Das bedeutet also, dass für viele Zwischenwerte von p der Graph stark geclustert ist (wie ein regulärer Graph), aber einen kleinen mittleren Durchmesser hat (wie ein zufälliger Graph). Watts und Strogatz nennen Netzwerke in diesem Bereich Small-World-Netzwerke. Ein Beispiel für solch ein Netzwerk stellt der mittlere Graph in Abbildung 12 dar.

Wenn wir nun ein Netzwerk dieses Modells betrachten, dessen Größe und mittlerer Knotengrad mit dem jeweiligen sozialen Netzwerk übereinstimmen, so können wir C und d_{avg} vergleichen und so auf eine eventuelle Small-World-Natur des sozialen Netzwerks schließen, falls $d_{avg} \approx d_{avg}^{rand}$ und $C \gg C^{rand}$. In Tabelle 9 werden die ermittelten Größen für die sozialen Netzwerke denen der entsprechenden regulären und zufälligen Netzwerke nach dem Modell von Watts und Strogatz gegenübergestellt. Wir sehen, dass die Durchmesser der sozialen Netzwerke ähnlich klein sind wie die der zufälligen Netzwerke, wenn wir den hohen Durchmesser in regulären Netzwerken gegenüberstellen. Der Clustering-Koeffizient hingegen liegt deutlich näher an dem für die regulären Netzwerke. Der geringe mittlere Durchmesser und der hohe Grad des Clusterings legen also nah, dass die sozialen Netzwerke eine ähnliche Struktur besitzen wie der mittlere Graph in Abbildung 12 und dass es sich nach Watts und Strogatz um Small-World-Netzwerke handelt.

8 Ausblick

Im Rahmen dieser Arbeit wurden vier soziale Online-Netzwerke erfasst, von denen eines jedoch aufgrund seiner Größe nur unvollständig extra-

hiert werden konnte. Es ist daher nicht sicher, dass die ermittelten Ergebnisse für dieses Netzwerk repräsentativ sind, obgleich sie keine merklichen Unterschiede zu denen der anderen Netzwerke aufweisen. Eine Extraktion des kompletten Datensatzes würde jedoch Aufschluss hierüber geben.

Die vier sozialen Netzwerke wurden zudem auf ihren Durchmesser sowie auf den Grad des Clusterings hin untersucht. Es gibt dabei noch viele weitere interessante Merkmale von Graphen, mit denen sich die Graph-Mining-Community beschäftigt. Solche, die sich sinnvoll auf die sozialen Netzwerke anwenden ließen, sollen an dieser Stelle kurz skizziert werden. Ein Aspekt ist z.B., inwiefern Knoten mit niedrigen Graden eher dazu neigen, mit solchen Knoten benachbart zu sein, die ebenfalls mit wenig Kanten inzident sind (Korrelation der Knotengrade) [New03]. Für unsere Netzwerke würde dies bedeuten, dass die Freunde von sozial wenig aufgeschlossenen Individuen auch nicht viele Freunde haben.

Eine Verallgemeinerung dieses Phänomens ist das Ausmaß, in dem Individuen ihre Freunde nach Ähnlichkeit zu sich selbst wählen („assortative mixing“) [New03]. Häufig lässt sich in sozialen Netzwerken beobachten, dass Akteure dazu tendieren, Beziehungen zu solchen Menschen zu haben, die ihnen beispielsweise in Nationalität oder ihren Hobbys ähnlich sind. In der Literatur finden sich dabei verschiedene Ansätze, die dieses Phänomen zu quantifizieren versuchen. Dabei sollte der Wert maximal sein, wenn jede Kante im Graph zwei Knoten desselben Typs (also mit gleichen Merkmalen) verbindet. Das Minimum hingegen sollte bei einer völlig zufälligen Paarung angenommen werden.

Eine weitere, verbreitete Technik, speziell in Bezug auf den Clustering-Koeffizienten, ist das Extrahieren von Communities. Besonders in sozialen Netzwerken lassen sich häufig Gruppen von Knoten ausmachen, innerhalb derer eine hohe Kantendichte besteht, die jedoch nur wenig Kanten zu anderen Gruppen aufweisen. Solche Gruppen, die sich meist durch ähnliche Eigenschaften auszeichnen, lassen sich anhand der Clusteranalyse identifizieren. Eine gebräuchliche Methode ist das hierarchische Clustering, bei dem allen Knotenpaaren eine Verbindungsstärke zugeordnet wird, basierend auf den betrachteten Eigenschaften. Anstatt Merkmale wie Nationalität oder Alter zu berücksichtigen, kann die Verbindungsstärke auch z.B. auf dem Grad der „strukturellen Äquivalenz“ basieren [New03]. Zwei Knoten sind strukturell äquivalent, wenn sie dieselben Nachbarn besitzen. Besonders in der Soziologie findet diese Eigenschaft meist große Beachtung. Ausgehend von einem Graphen ohne Kanten, werden beim hierarchischen Clustering nun nach und nach Kanten zwischen Knoten mit abnehmender Verbindungsstärke hinzugefügt. Nach jedem Schritt erhält man so eine Konstellation von Communities. Ein schneller bekannter Algorithmus benötigt hierfür $O(|E| \cdot d \cdot \log |V|)$ Zeit, wobei d der Tiefe des enstehenden Dendrogramms entspricht [CNM04].

Zuletzt sei ein ein aktuelles Problem aus der sozialen Netzwerkana-

lyse erwähnt, welches darin besteht, die wichtigsten Akteure eines Netzwerks zu identifizieren [CF06]. Es gilt dabei, die k Knoten zu finden, die bei Entfernung die Konnektivität des Netzwerks maximal reduzieren bzw. die zum Rest des Netzwerks maximal verbunden sind. Diese könnten z.B. am schnellsten Informationen im Netzwerk verbreiten. Auf der anderen Seite wäre bei einem Angriff auf das Netzwerk das Ausschalten dieser Knoten am wirkungsvollsten. Aufgrund der kombinatorischen Natur existieren jedoch keine effizienten Algorithmen für das Problem.

9 Zusammenfassung

Die zunehmende Größe und Popularität sozialer Netzwerke im WWW legt nicht nur eine nähere Untersuchung im Rahmen der „Knowledge Discovery in Databases“ nahe, sondern erfordert zugleich eine Datenstruktur, die relationale Informationen bewahrt und zudem effiziente Analysemethoden ermöglicht. Eine geeignete Struktur stellt dabei ein Graph dar.

Im Rahmen dieser Arbeit wurde eine Applikation entwickelt, welche Online-Netzwerke in Graphform extrahiert. Es wurden auf diese Weise vier soziale Netzwerke erfasst, in denen die Knoten die Akteure darstellen und eine Kante zwischen zwei Knoten dann existiert, wenn eine Freundschaftsbeziehung besteht. Anschließend wurden die Netzwerke dann auf Muster hin untersucht. Es zeigte sich, dass bei drei Netzwerken der Knotengrad einer Power-Law-Verteilung mit Exponenten zwischen 2 und 2.8 zu Grunde liegt. Während der mittlere Durchmesser den gemittelten kürzesten Weg zwischen sämtlichen Knotenpaaren darstellt, so gibt der effektive Durchmesser eines Graphen an, über wie viele Knoten sich der Großteil aller verbundenen Paare erreicht. Auf Basis eigener Implementierungen konnten für alle Netzwerke sehr geringe Durchmesser von maximal 7 bestimmt werden („Small-World-Effekt“). Dies bedeutet, dass sowohl Information als auch Computerviren oder Würmer schnell durch die Netzwerke propagiert werden können. Die Stärke der Community-Bildung kann durch den Clustering-Koeffizienten ausgedrückt werden, welcher die Wahrscheinlichkeit darstellt, dass zwei Nachbarn eines Knotens ebenfalls miteinander verbunden sind. Dieser zeigte sich im kleinsten Netzwerk am größten: Mit einer mittleren Wahrscheinlichkeit von ca. 32% sind zwei Individuen, die einen gemeinsamen Freund haben, ebenfalls miteinander befreundet.

Um die ermittelten Ergebnisse objektiv bewerten zu können, wurde ein Vergleich mit zufällig erzeugten Graphen derselben Größe und mit demselben mittleren Knotengrad nach dem Erdös-Rényi-Graphmodell hergestellt. In diesem ist der Knotengrad Poisson-veteilt. Die Durchmesser wachsen logarithmisch mit der Knotenanzahl und zeigen sich ähnlich gering wie in den sozialen Netzwerken. Der Clustering-Koeffizient strebt jedoch ge-

gen 0 für große Knotenanzahl und deutet auf einen erhöhten Grad der Community-Bildung in den sozialen Netzwerken hin.

Eine Einordnung in das Small World Modell von Watts und Strogatz bestätigt die Vermutung, dass es sich bei allen betrachteten sozialen Netzwerken um Small-World-Netzwerke handelt. Sie besitzen ähnlich große Clustering-Koeffizienten wie reguläre Graphen, dafür aber geringe Durchmesser wie zufällige Graphen.

Literatur

[ACL00] AIELLO, William ; CHUNG, Fan ; LU, Linyuan: A random graph model for massive graphs. In: *STOC '00: Proceedings of the thirty-second annual ACM symposium on Theory of computing*. New York, NY, USA : ACM Press, 2000. – ISBN 1–58113–184–4, S. 171–180

[Ada00] ADAMIC, Lada A.: Zipf, Power-law, Pareto - a ranking tutorial / Information Dynamics Lab, HP Labs. Version: October 2000. http://www.hpl.hp.com/research/idl/papers/ranking/. HP Labs, Palo Alto, CA 94304, October 2000. – Forschungsbericht. – Online Ressource

[AYZ97] ALON, Noga ; YUSTER, Raphy ; ZWICK, Uri: Finding and Counting Given Length Cycles. In: *Algorithmica* 17 (1997), Nr. 3, 209-223. citeseer.ist.psu.edu/article/alon95finding.html

[BC02] BIANCONI, Ginestra ; CAPOCCI, Andrea: *Number of loops of size h in growing scale-free networks*. http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0212028. Version: 2002

[BT02] BU, T. ; TOWSLEY, D.: *On distinguishing between Internet power law topology generators*. citeseer.ist.psu.edu/bu02distinguishing.html. Version: 2002

[BW00] BARRAT, A. ; WEIGT, M.: On the properties of small-world network models. In: *EUROP.PHYS.J.B* 13 (2000), 547. http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/9903411

[BWG06] BEIRLANT, Jan ; WET, Tertius de ; GOEGEBEUR, Yuri: A goodness-of-fit statistic for Pareto-type behaviour. In: *J. Comput. Appl. Math.* 186 (2006), Nr. 1, S. 99–116. http://dx.doi.org/http://dx.doi.org/10.1016/j.cam.2005.01.036. – DOI http://dx.doi.org/10.1016/j.cam.2005.01.036. – ISSN 0377–0427

[Cas04] CASTILLO, Carlos: *Effective Web Crawling*, Diss., 2004. http://chato.cl/research/crawling_thesis. – Online Ressource

[CF06] CHAKRABARTI, Deepayan ; FALOUTSOS, Christos: Graph mining: Laws, generators, and algorithms. In: *ACM Comput. Surv.* 38 (2006), Nr. 1. http://dx.doi.org/10.1145/1132952.1132954. – DOI 10.1145/1132952.1132954. – ISSN 0360–0300

[CH07] COOK, Diane J. ; HOLDER, Lawrence B.: *Mining Graph Data*. Wiley, 2007

[CNM04] CLAUSET, Aaron ; NEWMAN, M. E. J. ; MOORE, Cristopher: *Finding community structure in very large networks*. http://arxiv.org/abs/cond-mat/0408187. Version: August 2004

[CSW05] CARRINGTON, Peter J. ; SCOTT, John ; WASSERMAN, Stanley: *Models and Methods in Social Network Analysis*. Cambridge University Press, 2005

[CT99] CROVELLA, M. ; TAQQU, M.: *Estimating the heavy tail index from scaling properties*. citeseer.ist.psu.edu/crovella99estimating.html. Version: 1999

[DGM02] DOROGOVTSEV, S. N. ; GOLTSEV, A. V. ; MENDES, J. F. F.: Pseudofractal Scale-free Web. In: *Physical Review E* 65 (2002), 066122. http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0112143

[EMB02] EBEL, Holger ; MIELSCH, Lutz-Ingo ; BORNHOLDT, Stefan: Scale-free topology of e-mail networks. In: *Physical Review E* 66 (2002), 035103. http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0201476

[FFF99] FALOUTSOS, Michalis ; FALOUTSOS, Petros ; FALOUTSOS, Christos: On Power-law Relationships of the Internet Topology. In: *SIGCOMM*, 251-262

[FM85] FLAJOLET, Philippe ; MARTIN, G. N.: Probabilistic Counting Algorithms for Data Base Applications. In: *Journal of Computer and System Sciences* 31 (1985), Nr. 2, 182-209. citeseer.ist.psu.edu/flajolet85probabilistic.html

[GMY04] GOLDSTEIN, Michel L. ; MORRIS, Steven A. ; YEN, Gary G.: *Problems with Fitting to the Power-Law Distribution*. http://arxiv.org/abs/cond-mat/0402322v3. Version: Aug 2004

[LEA+01] LILJEROS, Fredrik ; EDLING, Christofer R. ; AMARAL, Luis A. N. ; STANLEY, H. E. ; ABERG, Yvonne: The Web of Human Sexual Contacts. In: *Nature* 411 (2001), 907. http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0106507

[Mil67] MILGRAM, Stanley: The Small World Problem. In: *Psychology Today* 1 (1967), May, S. 61–67

[New01] NEWMAN, M. E. J.: The structure of scientific collaboration networks. In: *PROC.NATL.ACAD.SCI.USA* 98 (2001),

404. http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0007214

[New03] NEWMAN, M.: *The structure and function of complex networks*. citeseer.ist.psu.edu/newman03structure.html. Version: 2003

[New04] NEWMAN, M. E. J.: *Power laws, Pareto distributions and Zipf's law*. http://arxiv.org/abs/cond-mat/0412004. Version: December 2004

[NFB02] NEWMAN, M. E. J. ; FORREST, Stephanie ; BALTHROP, Justin: Email networks and the spread of computer viruses. In: *Physical Review E* 66 (2002), Nr. 3, S. 035101+

[NW99] NEWMAN, M. E. J. ; WATTS, D. J.: Renormalization group analysis of the small-world network model. In: *Physics Letters A* 263 (1999), 341. http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/9903357

[OW02] OTTMANN, Thomas ; WIDMAYER, Peter: *Algorithmen und Datenstrukturen*. Spektrum Akademischer Verlag GmbH, 2002

[PGF02] PALMER, C. ; GIBBONS, P. ; FALOUTSOS, C.: *ANF: A fast and scalable tool for data mining in massive graphs*. citeseer.ist.psu.edu/palmer02anf.html. Version: 2002

[PSF+01] PALMER, C. ; SIGANOS, G. ; FALOUTSOS, M. ; FALOUTSOS, C. ; GIBBONS, P.: *The connectivity and fault-tolerance of the Internet topology*. citeseer.ist.psu.edu/palmer01connectivity.html. Version: 2001

[RB03] RAVASZ, Erzsebet ; BARABASI, Albert-Laszlo: Hierarchical Organization in Complex Networks. In: *Physical Review E* 67 (2003), 026112. http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0206130

[SS04] SAAKE, Gunter ; SATTLER, Kai-Uwe: *Algorithmen und Datenstrukturen*. dpunkt, 2004

[Sto07] STOCK, Wolfgang G.: *Information Retrieval*. Oldenbourg, 2007

[SW05a] SCHANK, Thomas ; WAGNER, Dorothea: Approximating Clustering Coefficient and Transitivity. In: *Journal of Graph Algorithms and Applications* 9 (2005), Nr. 2, 265-275. http://www.ubka.uni-karlsruhe.de/vvv/ira/2004/9/9.pdf

[SW05b] SCHANK, Thomas ; WAGNER, Dorothea: Finding, Counting and Listing All Triangles in Large Graphs, an Experimental Study. In: *WEA*, 2005, S. 606–609

[TM69] TRAVERS, Jeffrey ; MILGRAM, Stanley: An Experimental Study of the Small World Problem. In: *Sociometry* 32 (1969), Nr. 4, 425–443. http://links.jstor.org/sici?sici=0038-0431%28196912%2932%3A4%3C425%3AAESOTS%3E2.0.CO%3B2-W

[Wit01] WITT, Kurt-Ulrich: *Algebraische Grundlagen der Informatik*. vieweg, 2001

[WS98] WATTS, D. J. ; STROGATZ, S. H.: Collective dynamics of 'small-world' networks. In: *Nature* 393 (1998), June, Nr. 6684, 440–442. http://dx.doi.org/10.1038/30918. – DOI 10.1038/30918. – ISSN 0028–0836

Abbildungsverzeichnis

Tabellenverzeichnis