Entdecken von Web-Communities

Entdecken von Web-Communities

- Web-Community
  - Menge von Menschen, die ein gemeinsames Interesse teilen und im Internet elektronische Dokumente darüber verbreiten
- Datenquellen
  - Web-Seiten
  - Emails
  - Textdokumente
- Warum sind Web-Communities interessant?
  - sind wertvolle, vertrauenswürdige, aktuelle Informationsquellen für die Interessenten
  - repräsentieren die Soziologie des Webs, Evolution des Webs
  - sind eine interessante Zielgruppe für Werbung

Problem Definition

- Web-Community ist definiert als eine Teilmenge von G von S={s1, s2,..., sn} von Objekten, die mit einem Thema T assoziert sind.
  - Das Thema definiert die Community, d.h. zwei Communities sind gleich, wenn ihre Themen gleich sind.
  - Nicht alle Aspekte werden abgedeckt, z.B. zeitliche Entstehung
  - Communities können hierarchisch strukturiert sind, durch Untermengen in G und Unterthemen
- Communities sind nur implizit in den Daten präsent, d.h. die Mitglieder und die Themen müssen erst durch durch einen Algorithmus entdeckt werden.

Beispiele

- Web-Seiten
  - Hyperlinks: eine Gruppe von Web-Seitenbetreibern, die ein gemeinsames Interesse teil, verlinken typischerweise ihre Seiten
  - Schlüsselwörter: Web-Seiten einer Community enthalten Wörter, die auf das Thema bezogen sind.
- Emails
  - Email-Austausch zwischen Mitgliedern: Mitglieder einer Community tauschen sich öfter per Email aus
  - Schlüsselwörter: Emails innerhalb einer Community enthalten Wörter, die auf das Thema bezogen sind.
- Textdokumente
  - Gemeinsames Auftreten von Mitgliedern: Mitglieder eine Community tauchen mit höherer Wahrscheinlichkeit in einem Satz gemeinsam auf.
  - Schlüsselwörter: Texte enthalten Wörter, die auf das Thema bezogen sind

HITS zum Finden von Communities

- HITS findet die 1. Eigenvektoren von AA’ und A’A, welche die am stärksten verbundenen Autoritäten und Hubs in G repräsentiert (G ist durch Anfrage definiert)
- Nachfolgende Eigenvektoren repräsentieren alternative Communities zu den Anfragetermen
- Weitere Eigenvektoren können mittels Powermethode oder QR Zerlegung berechnet werden.

Bipartite Kerne als Communities

- (nahezu) vollständige Teilgraphen des Web als Communities?
  - Macht wenig Sinn, weil zentrale Seiten einer Community sich aus Konkurenz nicht gegenseitig verlinken (Daimler und BMW)
  - Die Autoren der interessanten Seiten wissen nicht in allen Fällen von einander
- Zusammenhänge zwischen zentralen Seiten einer Community werden durch gemeinsames Zitieren geschaffen (Co-citation).
- Durch Co-Citation können Communities entdeckt werden, bevor die Mitglieder sich bewußt werden, dass sie eine Community bilden.

Bipartite Kerne

- Bipartiter Graph
  - G=(V,E) mit V=V1 + V2, und Kanten gehen nur von Knoten aus V1 zu Knoten aus V2
- Bipartiter Teilgraph als Community
  - besteht aus den Knotenmengen F und C
  - viele der möglichen Kanten zwischen Knoten aus F und C existieren
  - Kanten dürfen auch innerhalb von F bzw. C existieren
- Bipartiter Kern
  - Teilgraph mit Knotenmengen F und C und alle gerichteten Kanten zwischen F und C existieren
  - Kerne sind meist klein, |F|,|C| ist <6

Bipartite Kerne

- Dichte bipartite Web-Teilgraphen enthalten einen bipartiten Kern mit hoher Wahrscheinlichkeit
  - eine Community kann auch mehrere Kerne enthalten
- Ziel
  - finde alle bipartiten Kerne und erweitere diese dann zu Communities

Vorverarbeitung

- Gegeben ist eine große Sammlung von Webseiten mit URLs
- Bipartite Kerne allein lieferen viele falsche Positive
- Probleme
  - kopierte Seiten
  - sehr bekannte Seiten
  - Selbstzitierungen, Links von der gleichen Domäne

Kopierte Seiten

- Gespiegelte oder kopierte Seiten mit kleinen Veränderungen (z.B. 90% der Links bleiben erhalten) bilden zufällige bipartite Kerne, die falsche Positive sind
- Filtern von Duplikaten mit LSH und Shingling
  - als Token werden nur die Links verwendet, keine Wörter
  - Shingle sind sehr klein, w=5 Token
  - Parametereinstellung ist eher grob, d.h. lieber unterschiedliche Seiten als Kopien anzeigen als eine Kopie zu übersehen
- Beispiele
  - URLs der gleichen Seite unterscheiden sich nur geringfügig, z.B. .htm und .html
  - kleine Unterschiede im Domännamen
  - viele Kopien von Yahoo Seiten
  - etwa 60% der wurden im Beispieldatensatz entfernt

Sehr bekannte Seiten

- Sehr bekannte Seiten erzeugen allgemeine Communities, wie „Informatik“
- Solche Communities sind uninteressant
- Ansatz
  - entferne alle Seiten mit einem hohen Eingangsgrad
- Eingangsgrad-Verteilung ist in etwa ein Pareto-Verteilung
  - Wahrscheinlichkeit, dass eine Seite den Eingangsgrad i hat ist etwa 1/i^2
  - Schwellwert für Eingangsgrad wurde empirisch auf k=50 gesetzt
  - Seiten mit Eingangsgrad >= 50 werden entfernt

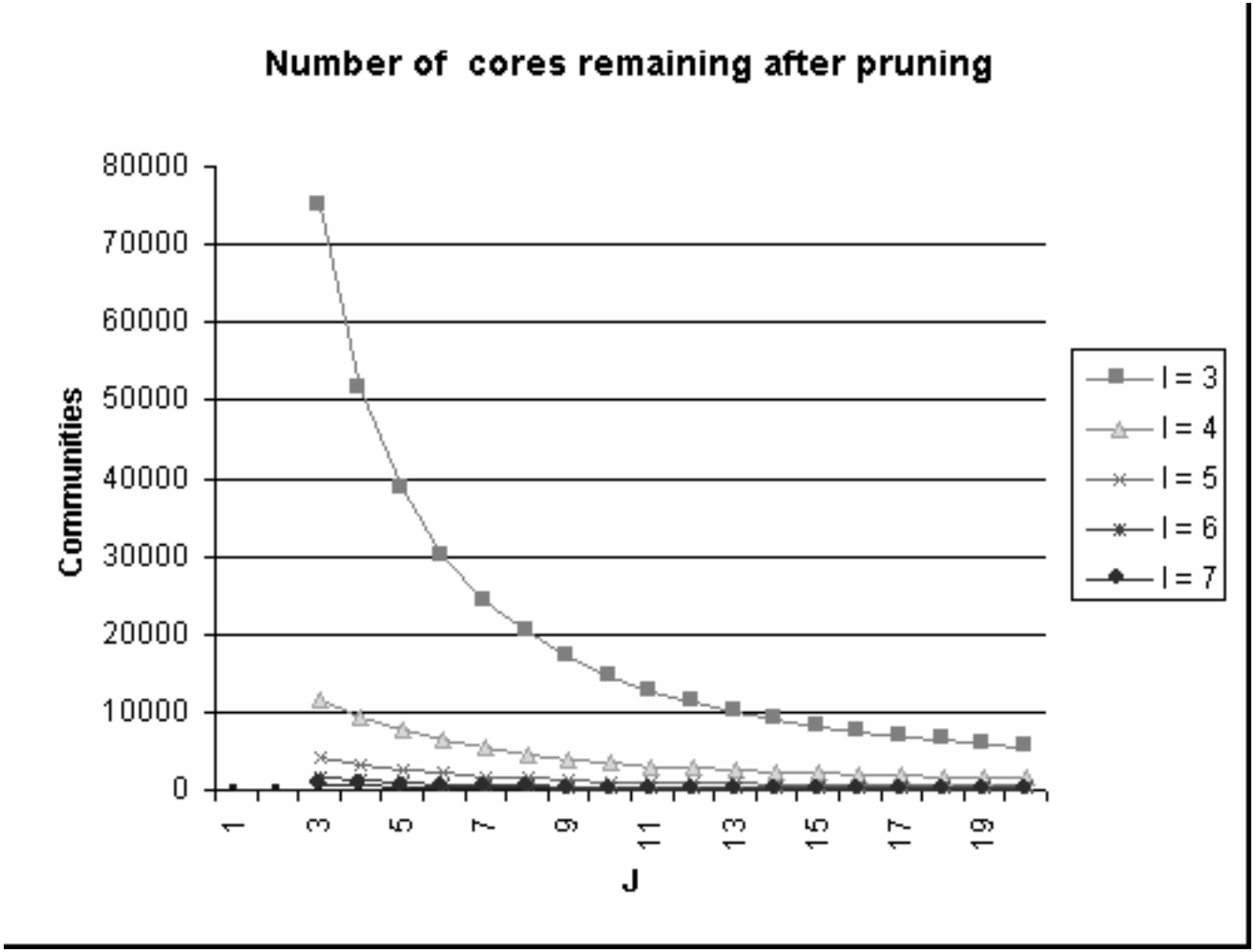
Iteratives Streichen

- Iteratives Streichen von Fans (F) und Centers (C)
  - Ein Fan eines (i,j) Kerns muss mindestens einen Ausgangsgrad von j haben
  - Ein Center eines (i,j) Kerns muss mindestens einen Eingangsgrad von i haben
- Das Streichen eines Fan kann ein Center unter den Schwellwert drücken
- Das Streichen eines Centers kann einen Fan unter den Schwellwert drücken
- Ansatz
  - sortiere Kanten nach Startknoten ID und lösche Fans mit zu kleinem Ausgangsgrad
  - sortiere Kanten nach Endknoten ID und lösche Center mit zu kleinem Eingangsgrad
  - Wiederhole die beiden Schritte
  - Insgesamt muss nur zweimal sortiert werden und pro Iteration muss eine Löschliste im Hauptspeicher gehalten werden

Erzeugen der (i,j) Kerne

- j wird im folgenden fest gewählt, z.B j=3
- Start
  - finde alle (1,j) Kerne durch Sortieren
- Iteration, i=2
  - erzeuge alle (i,j) Kerne durch das Prüfen aller Fans, ob sie auch auf die j Center in einem (i-1,j) Kern zeigen
- Algorithmus ist ähnlich zu A priory
- Das Ausgeben von Unter-Kernen wird verhindert

Beispieldaten



Von Bipartiten Kernen zu Communities

- Bipartite Kerne zu Communities erweitern
  - HITS: Umgebung eines Kerns mit HITS untersuchen
  - Pagerank: Umgebung des Kerns untersuchen, Knoten aus dem Kern haben höheren Quellrang
- Nachteile
  - Bipartite Kerne sind eine Annahme, die möglicherweise nicht auf alle Communities zutrifft
  - Größe der Kerne ist schwer zu justieren, s.d. sinnvolle Communities entdeckt werden

Web-Communities basierend auf dichten bipartiten Graphen

Ziel: finde alle Communities in einer großen Sammlung von Web-Seiten.

Vorgeschlagene Lösung:

- Analysiere Link-Muster
- Finde dichte bipartite Graphen (DBG) in der Sammlung

Definitionen

Bipartiter Graph

Ein BG ist ein Graph, der in zwei nicht-leere Teilmengen T und I zerlegt werden kann, s.d. jede gerichtete Kante einen Knoten aus T mit einen Knoten aus I verbindet

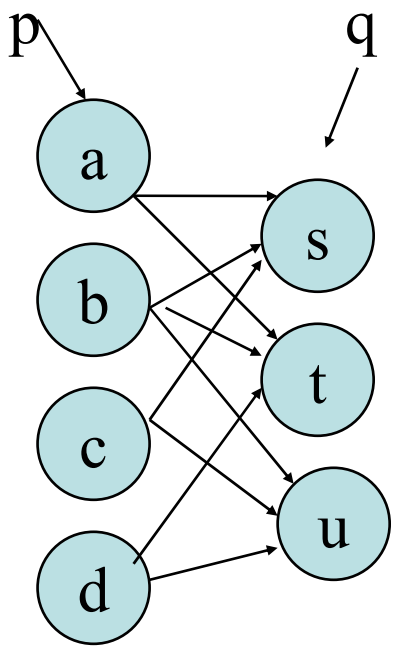
Dichter bipartiter Graph

Ein DBG ist ein BG, wobei jeder Knoten aus T mit mindestens alpha Knoten aus I verbunden ist und jeder Knoten aus I hat mindestens beta Knoten als Eltern

Community

Die Knotenmenge T enthält die Mitglieder einer Community, wenn ein DBG(T,I,alpha,beta) existiert, mit alpha>= alpha_t und beta>=beta_t, wobei alpha_t and beta_t > 0.

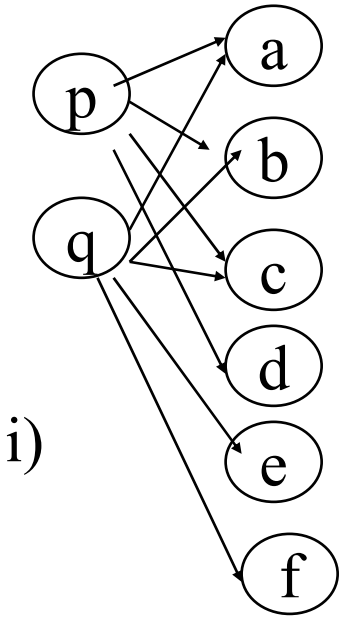
DBG(T,I,p,q)



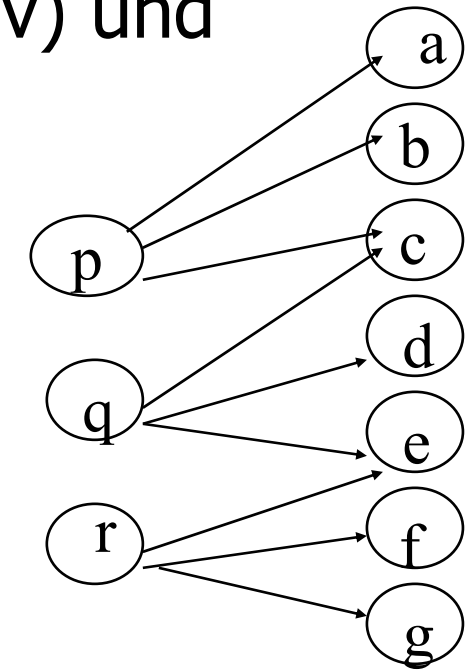
Definitions

Cocite: Relation zwischen Seiten mit gemeinsamen Kindern (URL's).

Relax Cocite: u,v,w werden zu einer Gruppe zusammgefasst, wenn cocite(u,v) und cocite(v,w) gelten.



i)



Algorithmus

1. Gegeben eine URL finde T (Menge von URL's) mit Relax-Cocite Faktor 1.

a)While num_iterations<=n

- Für einen festen Relax-Cocite Faktor, finde alle w's, s.d. Relax-Cocite(w,y) =true
- T= w U T

2. Community Extraktion
   - Eingabe enthält Menge der Seiten und outputDBG(T,I,alpha,beta)
   - Kanten-Liste besteht aus Tupeln <p,q>, mit p ist Elternteil von q.

Algorithmus

- Für alle p aus T, füge die Kante <p,q> in Kanten-Liste
- Sortiere Kantenliste von Startknoten und erstelle T1 mit <source,freq>. Entferne <p,q> aus der Kantenliste, falls freq<alpha.
- Sortiere Kanteliste nach Zielknotenund erstelle I1 mit <q,freq>. Entferne <p,q> aus der Kantenliste, falls freq<beta.
- Das Ergebnis ist ein DBG(T,I,alpha,beta).

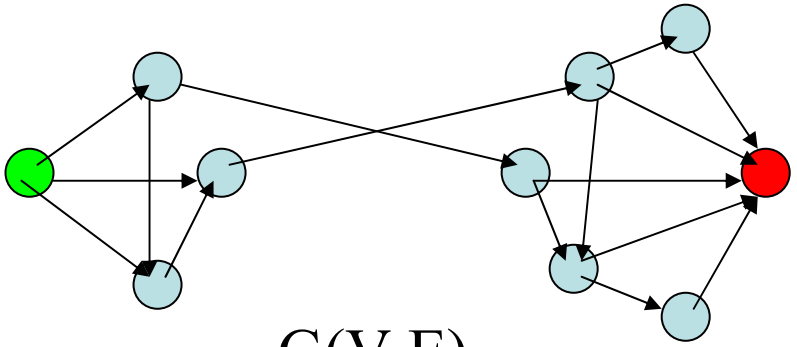
Vorteile/Nachteile

- Vorteile
  - Findet alle DBG’s in einer Menge von Seiten.
  - Community hat signifikante Größe.
- Nachteile
  - Brauchte eine URL zum starten.
  - Mitglider der Community müssen hinreichend verlinkt seinen

Effiziente Identifikation von Web Communities

Gary William Flake, Steve Lawrence & C. Lee Giles

s-t Max Flow & Min Cut



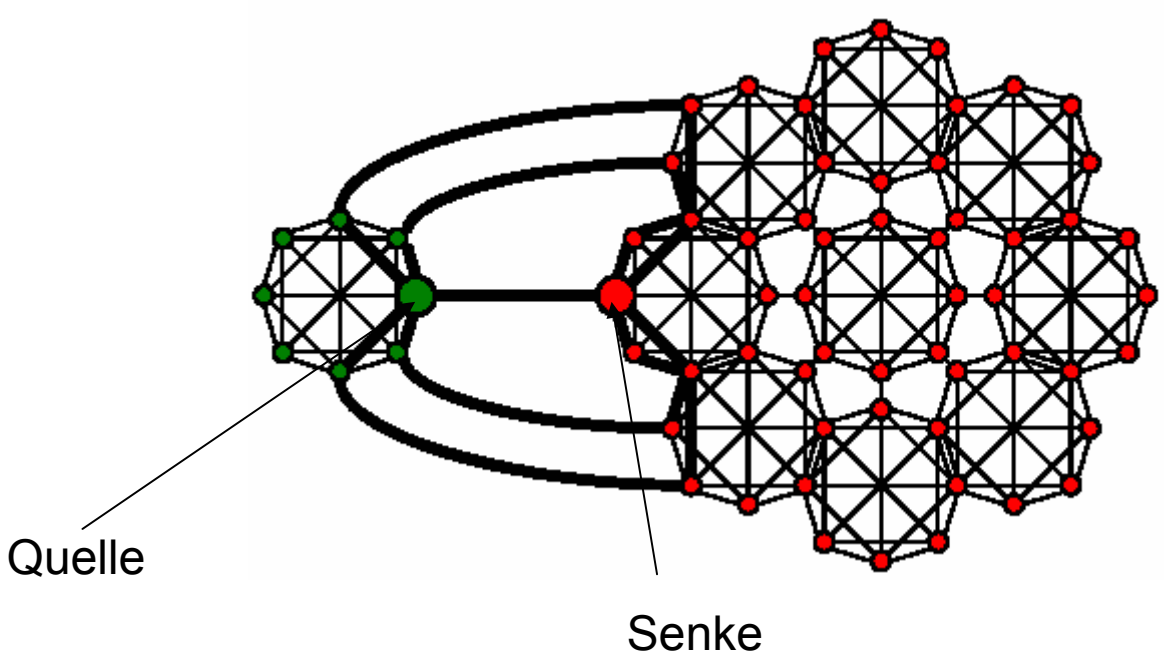
G(V,E)

- Capacity weights
- Source & Sink
- Water In, Water Out!

- Floyd & Fulkerson's Max Flow = Min Cut Theorem
- Incremental Shortest Augmentation Algorithmus in Polynomieller Zeit

s-t Max Flow & Min Cut

- In der Web Community existiert ein Kern, der die Quelle s bildet
- Außerhalb dieser befindet sich eine Senke t



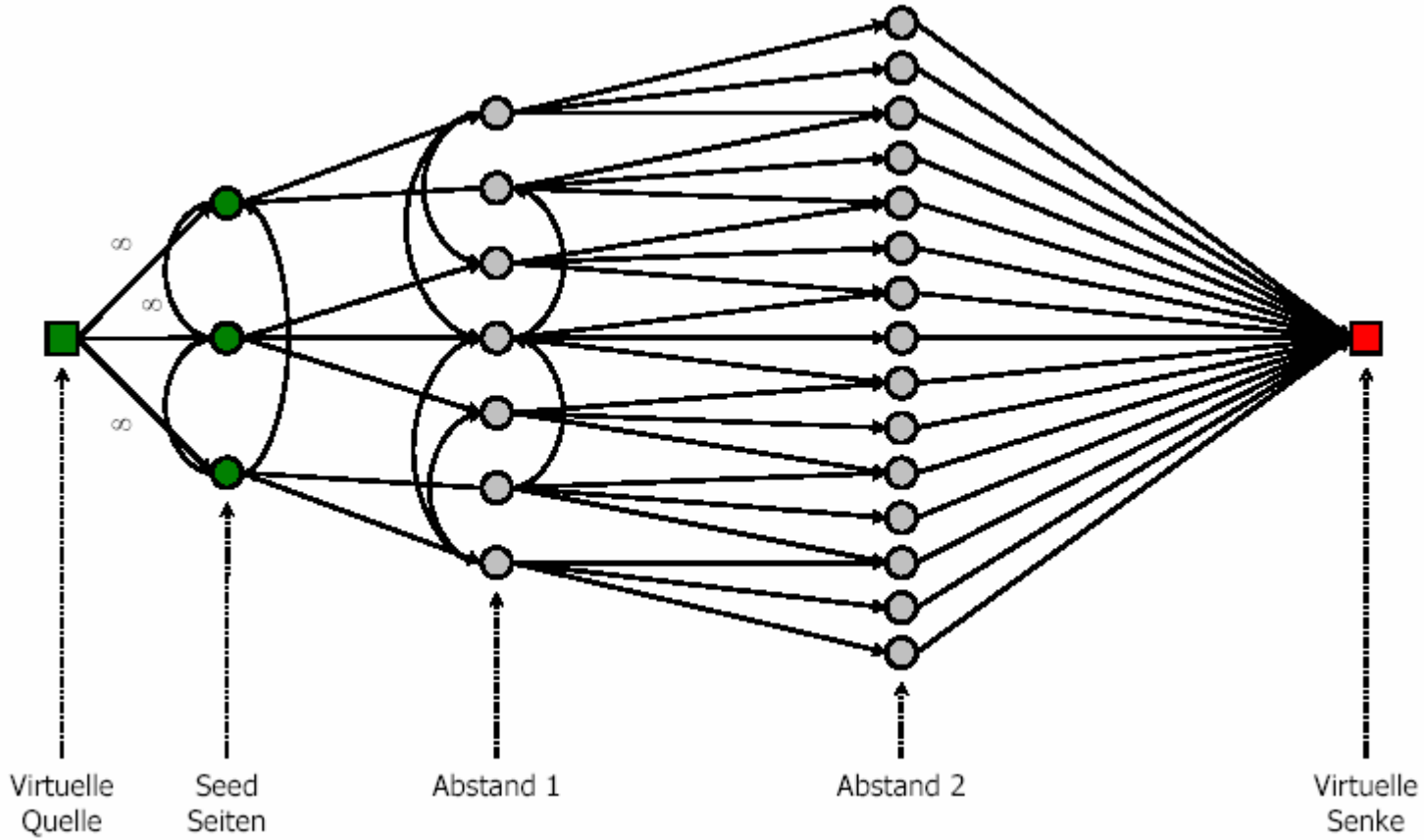
Idee

- die ideale Community $C \subset V$ sei bekannt

Theorem: Eine Community C kann durch s-t minimum cut mit passender Quelle und Senke gefunden werden.

Algorithmus

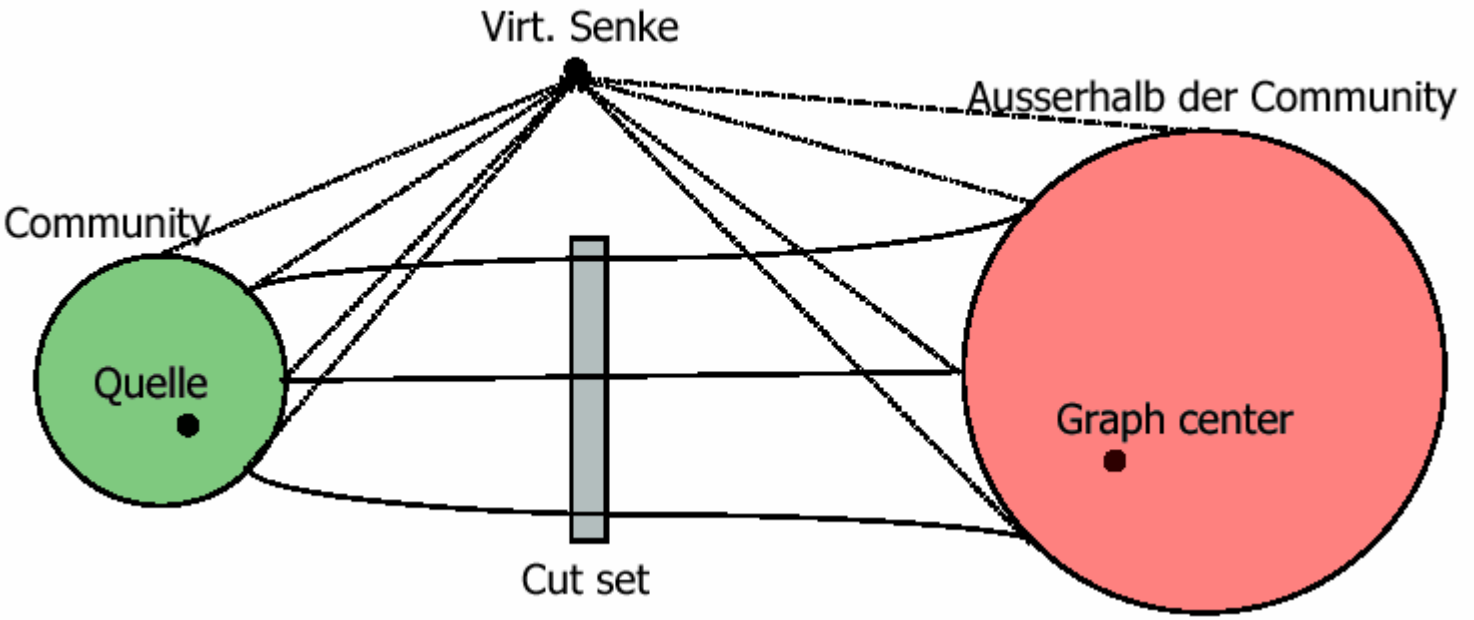
1. Starte mit Seed Seiten und virtueller Quelle
2. Erzeuge G(V,E) mit Crawler
3. Finde *s-t* Min Cut



Expectation Maximization

- Implementation Fragen
  - Kleiner Graph G(V,E) führt zu niedriger Ausbeute
  - Abhängig von der Wahl der Seed-Menge
- Rekursiver Algorithmus
  - Community die in einer Iteration gefunden wurde, dient als Seeds in der nächsten Iteration
- Abbruch ist nicht garantiert

Expectation Maximization



Experimental Results

- Testing neighborhoods …
  - Support Vector Machine (SVM)
  - The Internet Archive
  - Ronald Rivest
- Criterion
  - Precision & Recall
  - Seed set size
  - Running time

SVM Community

- Characterization
  - Recent: Not listed in any portal
  - Relatively small research community
- Seed Set
  - svm.first.gmd.de, svm.research.bell-labs.com, www.clrc.rhbnc.ac.uk/research/SVM, www.support-vector.net
- Performance
  - 4 iterations of EM
  - 11,000 URLs in the graph, 252 member web pages

Internet Archive Community

- Characterization

  Large, internal communities
- Seed Set : 11 URLs
- Performance
  - 2 iterations of EM
  - 7,000 URLs, 289 web pages

Ronald Rivest Community

- Characterization
  - Community around an individual
- Seed set
    - http://theory.lcs.mit.edu/~rivest
- Performance
  - 4 iterations of EM
  - 38,000 URLs, 150 pages
  - Cormen's pages as 1st and 3rd result