Ein suchunterstützender Algorithmus in verteilten Communities

Martin Luboschik
martin.luboschik@stud.uni-rostock.de
Universität Rostock, Fachbereich Informatik

Betreuer der Arbeit: PD Dr.-Ing. habil Herwig Unger, hunger@informatik.uni-rostock.de
Art der Arbeit: Forschungsarbeit
Fachbereich der GI: 3

Zusammenfassung

Ca. 160 Millionen Computer haben heutzutage Zugang zum Internet - einem der größten, schnellstwachsenden und meistgenutzten Medien. Durch die ständig wachsende Struktur des Internets kommen allerdings nicht nur neue, sondern auch redundante und veraltete Daten in diesem Rechnerverbund hinzu und es stellt sich immer öfter die Frage, wo die relevanten Daten zu finden sind. Diese Arbeit untersucht deshalb einen verteilten suchunterstützenden Algorithmus, der vor allem auf der Relevanz von Daten beruht.

1. Einleitung

Nach [1] wird das Auffinden von Daten im ständig wachsenden Internet und dem damit wachsenden Ressourcenraum aus verschiedenen Gründen (mangelnde Verfügbarkeit, geringe Abdeckung und Aktualität, geringe Treffsicherheit) immer schwieriger und eine effektive Suche mit Hilfe zentralisierter Systeme (z.B. Suchmaschinen) fraglich.

Mit der Einführung des Konzeptes der Communities, deren Strukturen nach [2, 3, 7] auch im Internet zu finden sind, ergeben sich für die oben genannten Problemstellungen innovative Lösungen, die vor allem auf dezentralisierten adaptiven Algorithmen basieren [3, 8]. Eine Community versteht sich hierbei als Gruppe von Objekten, die ein gemeinsamer Kontext verbindet. Diese Objekte besitzen eine Menge gleicher oder ähnlicher Interessen und sind in der Lage untereinander zu kommunizieren. Communities bilden sich lediglich durch Informationen über die Nachbarn, die jedes Mitglied in einem Warehouse (Buddylist, Hotlist, Links etc.) speichert. In diesen Strukturen kennt jedes Mitglied nur eine geringe Anzahl der Mitglieder der gesamten Community und kann auch nur mit diesen kommunizieren. Die Community verfügt im Allgemeinen über mehr Wissen und Problemlösungskompetenz, als ein einzelnes Mitglied haben könnte, und das, obwohl der Aufbau der Community keinem Mitglied vollständig bekannt ist.

Solche Communities entsprechen nach [4] in der Struktur sozialen Netzwerken, die die Small-World-Eigenschaft aufweisen (über 6-8 Stufen lassen sich mit einer Wahrscheinlichkeit von 96% Verbindungen zwischen zwei entfernten Knoten eines Netzwerkes aufbauen).

2. Problemstellung

Wie unter Kap. 1 bereits erwähnt wurde, stellt das Auffinden aktueller und vor allem relevanter Informationen in Communities ein Problem dar. Ansätze wie der Page-Rank-Algorithmus versuchen zwar über den In-Degree (auf den Knoten verweisende Links) eines Knoten dessen Relevanz zu bewerten, können dessen Aktualität und tatsächliche Relevanz aber nicht sicherstellen. Dezentrale Suchalgorithmen in Communities basieren unter Vermeidung hoher Netzlasten (z.B. Message-Chain-Algorithmen) auf zufällig gewählten Suchpfaden und sind deshalb in großen Communities wenig effizient. Auch sie berücksichtigen nicht die Relevanz von Informationen. Für eine effiziente Suche, bei der zudem die Aktualität und Relevanz einer gesuchten Information eine Rolle spielen, sollte aufgrund der wachsenden Struktur ein dezentraler Algorithmus verwendet werden, der nur wenig Netzlast produziert, sich den aktuellen Umständen der Community (Aktualität) anpasst und lediglich auf der Grundlage der in den Netzknoten lokal vorhandenen Informationen, sowie dessen direkten Nachbarn arbeitet. Ziel sollte es sein, sich dynamisch anpassende Pfade durch die Struktur einer Community zu legen, die auf kurzen Wegen zur aktuellsten und relevantesten Information führen.

3. Thermosearch

Als Maß für die Relevanz von Knoten werden die Aktualisierungsrate sowie die Zugriffsrate auf einen Knoten innerhalb der Community aufgefasst. Umso öfter ein Knoten geändert und aktualisiert bzw. auf diesen zugegriffen wird, desto relevanter scheint der entsprechende Knoten bzw. die enthaltenen Informationen für die Community zu sein. Solche Knoten können metaphorisch als „heiß“ eingestuft werden und führen so zur Idee des Thermosearch-Algorithmus. Um diesem Ansatz zu folgen, erhält jeder Knoten der Community eine eigene Temperatur, die von den o.g. Zugriffs- und Aktualisierungsraten abhängt und somit angibt, wie relevant ein solcher Knoten ist. Außerdem ist jeder Knoten in der Lage, die Temperaturen seiner Nachbarn zu speichern und nur beim Zugriff auf einen Nachbarknoten diese Temperatur zu messen. Hierdurch wird die Netzlast gering gehalten. Beim Zugriff auf einen Nachbarknoten kommt es zudem zu einem Wärmefluss, sodass sich nach kurzer Zeit ein Gradientenfeld zu den heißesten Knoten, den sog. „Hotspots“ etabliert.

3.1. Algorithmus

Der folgende Thermosearch-Algorithmus arbeitet unabhängig lokal auf jedem Knoten der Community und soll hier nun wiedergegeben werden. X sei die Menge aller Knoten einer Community. Jeder Knoten $x \in X$ hat eine eigene Temperatur $\theta(x)$ sowie ein Warehouse $N(x)$, wobei $N(x) \subset X$.

1. Initialisierung der Temperaturwerte $\theta(x) \ \forall x \in X$
2. Aktualisierung von $\theta(x)$ des aktuellen Knoten x abhängig von Zugriffs- und Aktualisierungsraten seit letzter Temperaturaktualisierung oder exponentielle Abkühlung [$\theta(x) = \theta(x) * e^{-\lambda t}$] des Knoten wenn keine Zugriffe oder Aktualisierungen vorlagen
3. exponentielles Senken von $\theta(y)$ in $N(x) \ \forall y \in N(x)$
4. warte bis Message-Chain ankommt oder der nächste Knotenaktualisierungszeitpunkt t erreicht wird.
5. wenn Message-Chain ankommt
   dann
   i. wähle $Next\_Node$ zufällig aus $N(x)$.
   ii. wenn ein Knoten y in $N(x)$ existiert, mit $\theta(y) > \theta(Next\_Node)$
   dann setze $Next\_Node = y$.
   iii. wenn $\theta(x) < \theta(Next\_Node)$
   dann leite Message-Chain an $Next\_Node$ weiter;
   sonst terminiere die Message-Chain;
   antworte der Quelle der Message-Chain;
   goto 6.
   iv. empfange die Temperatur von $Next\_Node$ als Acknowledgement der Sendeoperation und aktualisiere $\theta(Next\_Node)$ in $N(x)$.
   v. wenn Temperaturdifferenz $\theta(Next\_Node) - \theta(x)$ über einem Grenzwert liegt
   dann erhöhe $\theta(x)$ um einen Betrag abhängig von der Temperaturdifferenz.
6. wenn der nächste Knotenaktualisierungszeitpunkt t gekommen ist
   dann goto 2
   sonst goto 4.

Mit diesem Algorithmus etabliert sich bereits nach kurzer Zeit ein Temperaturfeld über der Community, dessen mittlerer Temperaturwert stark von der Abkühlung λ abhängt. Unter einem optimalen Temperaturfeld kann hierbei dasjenige verstanden werden, bei dem möglichst jeder Knoten eine Temperatur $\neq 0$ aufweist und somit wegbeschreibende Informationen gespeichert hat. Dies kann zum einen dadurch erreicht werden, in dem zum Aufbau des Temperaturfeldes möglichst viele Knoten der Community eine Suchanfrage nach den in den Hotspots zu findenden Informationen absenden. Zum anderen wirkt sich eine wiederholte Anfrage nach gleichen Informationen positiv auf die Generierung eines optimalen Temperaturfeldes aus.

Bedeutsam ist dieser Algorithmus wegen seiner Anpassungsfähigkeit. Sollte ein Hotspot nicht mehr als solcher aktiv sein, so kühlt er selbst und auch der Pfad zu ihm ab. Das Gradientenfeld in der Community richtet sich neu aus und weist nun auf andere Hotspots.

4. Simulation

Durch die Ausbildung eines Gradientenfeldes in der Community wird das Auffinden aller Hotspots einer Community bzw. das Auffinden anderer Hotspots zu einem Problem. Message-Chains, die nach dem Hotspot A suchen, können auf einen „Gradientenpfad“ zum Hotspot B gelangen und werden unweigerlich von diesem angezogen. Folglich sinkt mit der Zunahme der Hotspots die Wahrscheinlichkeit, alle Hotspots zu finden.

In der durchgeführten Simulation wird deshalb in jedem Knoten mit Hilfe einer vorgegebenen Wahrscheinlichkeit entschieden, ob der klassische Message-Chain-Algorithmus (zufällige Wahl des nächsten Nachbarn) oder der Thermosearch-Algorithmus anzuwenden ist. Dadurch ist ein Verlassen des Gradientenpfades – abhängig von der gegebenen Wahrscheinlichkeit – möglich.

Ziel der Simulation ist nun die Untersuchung der Suchzeiten abhängig von der Wahrscheinlichkeit für eine zufällige Suche. Zu diesem Zweck wird ein Temperaturfeld mit n Hotspots generiert und danach eingefroren, d.h. es findet keine weitere Temperaturänderung mehr statt. In diesem Temperaturfeld werden von einer fest gewählten Menge von Startknoten Suchanfragen pro Hotspot und pro Durchlauf generiert. Bei jedem Durchlauf wird dabei die Wahrscheinlichkeit für die zufällige Suche erhöht.

4.1 Simulationsergebnisse

Mit Hilfe eines selbstprogrammierten Simulationstools wurde nach dem Modell des Edge-Reassigning Small-World Network [5] eine zufällige Community generiert. Auf dieser wurde mit dem gleichen Tool unter Verwendung des oben beschriebenen Thermosearch-Algorithmus ein Temperaturfeld aufgebaut, auf dem wiederum die Simulationsläufe stattfanden.

Abb. 1 zeigt die Simulationsergebnisse für die Suche nach einer unterschiedlichen Anzahl von Hotspots innerhalb einer Community. Dabei wird für die gleiche Community in Abhängigkeit der Wahrscheinlichkeit die erforderliche Suchzeit zum Auffinden aller Hotspots dargestellt. Die dargestellten Werte sind Mittelwerte aller suchenden Startknoten. Erfolgreiche Suchvorgänge sind durch Suchzeiten, die kleiner als die Lebensdauer der Message-Chains (hier 255) sind, erkennbar. Man sieht, dass der Thermosearch-Algorithmus für die Suche nach einem einzigen Hotspot gegenüber der zufälligen Suche sehr viel effizienter ist (ca. 40fach schneller). Jede zusätzliche Randomisierung der Suche führt zu einer höheren Suchzeit. Bereits bei zwei Hotspots ist ihr Auffinden unter alleiniger Anwendung des Thermosearch-Algorithmus nicht mehr möglich und führt somit zu einer maximalen Suchzeit von 255 Zeitschritten. Erst durch die Hinzunahme der zufälligen Suche wird ein Auffinden aller Hotspots möglich und erreicht bei $p = 0{,}32$ ein Optimum in der Suchzeit. Der geringe Anteil der zufälligen Suche wird hier lediglich für einen Wechsel zwischen den Gradientenpfaden benötigt und jeder zusätzliche Gebrauch dieser, führt zu einer Verschlechterung der Suchzeiten. Wird ein dritter Hotspot in das Thermofeld eingebaut, so verschlechtern sich die Suchergebnisse weiter. Gleichzeitig gewinnt die zufällige Suche an Bedeutung: Sie muss hier öfter eingesetzt werden, um auch einen dritten Gradientenpfad und somit den Weg zum dritten Hotspot zu erreichen. Das Suchoptimum stellt sich hier bei einer Wahrscheinlichkeit für die zufällige Suche um $p = 0{,}5$ ein. Die größeren Suchzeiten ergeben sich vor allem durch die Anziehung nicht für einen Hotspot bestimmter Message-Chains und deren dortiger Terminierung. Dies entspricht einem Nichtauffinden des eigentlich gesuchten Hotspots und somit einer Suchzeit von 255. Trotz einer Kombination mit einem zufälligen Suchalgorithmus sinkt also die Wahrscheinlichkeit für das Auffinden aller Hotspots mit deren Zunahme.

Im Verlauf der Simulationsexperimente fiel auf, dass Startknoten ohne Informationen über ihre Nachbarn (Temperatur = 0) in einzelnen Fällen dazu in der Lage sind, alle Hotspots sogar bei der Wahrscheinlichkeit $p = 0$ zu finden. Dies ist durch die zufällige Wahl des nächsten Knoten im Thermosearch-Algorithmus (siehe 3.1 5i) sowie durch Verbindungen zu jedem Gradientenpfad zu erklären. Wenn beide Bedingungen (kein Wissen und die richtigen Verbindungen) erfüllt sind, können jene Knoten als Sattelpunkte im Gradientenfeld angesehen werden, von wo aus jeder Hotspot erreichbar ist. Die zufällige Wahl entscheidet, welcher der Hotspots gefunden wird, und so kann durch ein zufälliges Absenden der einzelnen Message-Chains in die jeweils richtige Richtung jeder dieser Hotspots gefunden werden. Die Existenz solcher Knoten ist allerdings nicht gesichert.

Abb. 2 zeigt die Simulationsergebnisse für die Suche nach einer festen Anzahl von Hotspots innerhalb unterschiedlicher Communities. Dabei wird für die verschiedenen Communities in Abhängigkeit der Wahrscheinlichkeit ebenfalls die erforderliche Suchzeit zum Auffinden aller Hotspots dargestellt. Auffällig ist hierbei die Verschiebung des Optimums mit der Größe der Community. So führt in kleinen Communities ein höherer Anteil der zufälligen Suche ($p = 0{,}65$) schneller zum Auffinden der Hotspots als

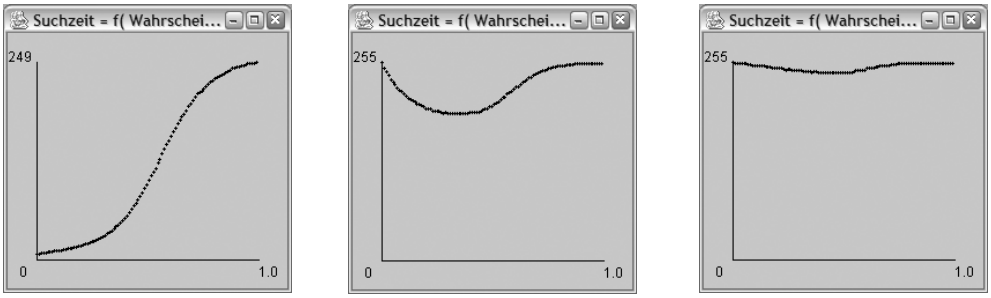
bei großen Communities. Bei diesen verhält es sich genau umgekehrt: Ein geringer Anteil der zufälligen Suche (p = 0,32) führt hier zu einem schnellen positiven Suchergebnis. In kleinen Communities reicht die alleinige zufällige Suche bereits aus, um alle Hotspots zu finden. Vergrößert sich die Community, so versagt die zufällige Suche zunehmend und die Suchergebnisse im Anteil der zufälligen Suche verschlechtern sich. Der „zufällige Anteil“ der Kurven wird nach oben gezogen, während der „Thermo-Anteil“ relativ gleich bleibt. Das Optimum verschiebt sich also als „Schnittpunkt der Anteile“.

5. Zusammenfassung

Mit dem Thermosearch-Algorithmus wurde ein Algorithmus gefunden, der als reiner Suchalgorithmus in nur sehr begrenztem Umfang anwendbar ist. Er basiert aber auf der Relevanz von Information und dient somit bei der Anwendung einem sehr schnellen und effizienten Auffinden von diesen. Durch den direkten Bezug auf die Zugriffs- sowie die Aktualisierungsrate eines Knotens ist dieser Algorithmus in der Lage, die Suchpfade den sich ändernden Umgebungen anzupassen.

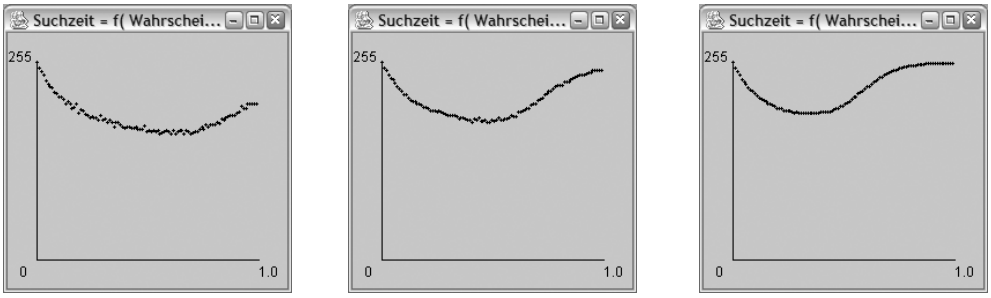
Zum Auffinden mehrerer Hotspots innerhalb einer Community ist Thermosearch bei alleiniger Anwendung unbrauchbar und entwickelt erst in Kombination z.B. mit der zufälligen Suche eine gewisse Effizienz. Aus diesem Grund kann dieser Algorithmus nur als suchunterstützender Algorithmus eine Rolle spielen. Außerdem ist das Mischverhältnis der angewandten Algorithmen von hoher Bedeutung. Dieses hängt aber direkt von der Communitygröße ab, die sich im Realfall kontinuierlich ändert. Eine Berechnung der aktuellen Communitygröße sowie die Propagierung dieser an alle Knoten der Community ist jedoch zeitaufwendig und produziert Netzlast, sodass die Anpassung eines gemischten verteilten Algorithmus kaum möglich sein wird.

Interessant ist die Entdeckung der Sattelpunke innerhalb des Gradientenfeldes. Ein Absenden von Kopien einer Message-Chain an alle Nachbarn eines solchen Punktes kann zum Auffinden aller Hotspots in kürzester Zeit führen. Da die Existenz dieser Knoten allerdings nicht sichergestellt und ein Auffinden dieser Knoten schwer, eventuell sogar unmöglich ist, ist eine Nutzung dieser Sattelpunkte fraglich.



a) b) c)

Abb.1 : Suchzeiten in Abhängigkeit der Wahrscheinlichkeit für die zufällige Suche
Edge-Reassigning Small-World Network mit 4096 Knoten, 16 Nachbarn
und Reassignment Wahrscheinlichkeit p = 0,7
a) 1 Hotspot, b) 2 Hotspots, c) 3 Hotspots



a) b) c)

Abb.2 : Suchzeiten in Abhängigkeit der Wahrscheinlichkeit für die zufällige Suche
Edge-Reassigning Small-World Network mit 16 Nachbarn, Reassignment Wahrscheinlichkeit p = 0,7
und 2 Hotspots pro Community.
a) 256, b) 768, c) 4096 Knoten in der Community

6. Literatur

[1] S. Lawrence, C. Lee Giles; Accessibility of Information on the web. Nature, 400: 107 – 109, 1999.

[2] D. Gibson, J.M. Kleinberg, P. Raghavan; Inferring web communities from link topology. UK Conference on Hypertext, 225-234, 1998.

[3] R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins; Trawling the web for emerging cyber-communities. WWW8 / Computer Networks, 31 (11-16): 1481-1493, 1999.

[4] Stanley Milgram. The small-world problem. Psychology Today, 1967.

[5] N.Deo, P.Gupta; Graph-Theoretic Web Algorithms: An Overview. IICS 2001, University of Central Florida, 2001.

[6] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, J. Wiener; Graph structure in the web: experiments and models. Ninth International World Wide Web Conference, Amsterdam, The Netherland, 2000.

[7] J. Hummel, U. Lechner; Profiling virtual communities. In R. Sprague, editor, Hawaiian Int. Conf. on System Sciences (HICSS 2002), IEEE Press, 2002.

[8] G. Flake, S. Lawrence, C. Lee Giles; Effizient identication of web communities. In Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 150-160, Boston, 2000.