Accepted for Kommunikation in Verteilten Systemen (KiVS) 2007
26. Februar - 2. März 2007 in Bern, Schweiz

Netzwerkeffizienz stabiler Overlay-Streaming-Topologien

Thorsten Strufe[1], Jens Wildhagen[2] und Günter Schäfer[1]

[1] Fachgebiet Telematik/Rechnernetze, Technische Universität Ilmenau
[2] Fachgebiet Integrierte Hard- und Softwaresysteme, Technische Universität Ilmenau

Zusammenfassung. Bei der Konstruktion von Overlay-Topologien für multimediale Live-Streaming-Anwendungen sind zwei Eigenschaften von besonderer Bedeutung: die *Netzwerkeffizienz* der Topologie in Bezug auf die Paketverteilung und die *Stabilität* der Topologie sowohl im Fall vorsätzlicher Sabotageangriffe als auch bei zufälligen Knotenausfällen. Während ein Großteil der existierenden Ansätze hauptsächlich Effizienzkriterien optimiert und nur wenige Ansätze Stabilität gegen zufällige Ausfälle betrachten, ist es uns in früheren Arbeiten gelungen, Verfahren für die Konstruktion angriffsstabiler Topologien zu entwickeln [17,3]. Da in diesen Arbeiten zum Zweck der Steigerung der Stabilitätseigenschaften der Topologie eine Verschlechterung der Effizienzeigenschaften bewusst in Kauf genommen wurde, wird in dem vorliegenden Artikel ein Verfahren vorgestellt, dass es ermöglicht, einen Kompromiss zwischen Effizienz und Stabilität bei der Konstruktion der Topologie zu finden.
Hierzu werden zunächst Stabilitäts- und Effizienzeigenschaften in Form von Kostenmetriken operationalisiert und darauf aufbauend ein verteilter Algorithmus zur dynamischen Topologieoptimierung vorgestellt, der eine Gesamtkostenfunktion optimiert, die durch eine parametrisierbare, gewichtete Kombination der Einzelmetriken definiert ist. Mit Hilfe einer Simulationsstudie wird gezeigt, dass auf diese Weise gute Kompromisse zwischen Effizienz und Stabilität bei der Topologiekonstruktion gefunden werden können.

1 Einleitung

In den vergangenen Jahren wurde für den Internet-basierten Transport von Multimedia-Inhalten hin zu großen Benutzerpopulationen der Einsatz des sogenannten *"Application Level Multicast"*-Ansatzes (ALM) vorgeschlagen [7]. Der prinzipielle Vorteil dieses Ansatzes ist, dass die den Datenstrom empfangenden Systeme diesen für andere Systeme replizieren und somit das "Angebot" an potentiellen Quellen für einen bestimmten Datenstrom automatisch mit der Nachfrage nach diesem Datenstrom steigt. Der Ansatz weist daher theoretisch eine beliebige Skalierbarkeit in der Anzahl der Empfänger auf.

Bei der Realisierung eines ALM-basierten Verteildienstes für die Live-Übertragung multimedialer Daten ("Live-Streaming") sind neben den üblicherweise

an Übertragungsdienste für Multimedia-Daten gestellten Dienstgüteanforderungen nach einer möglichst geringen *Ende-zu-Ende-Verzögerung (Delay)* und *Schwankung dieser Verzögerung (Jitter)* auch Anforderungen in Bezug auf die Effizienz der Verteiltopologie zu beachten – letztere charakterisiert durch das *Verhältnis der Pfadlängen in der Topologie im Vergleich zum kürzesten Pfad ("Path Stretch")* und die *Anzahl von Kopien identischer Pakete auf einzelnen Teilstrecken ("Link Stress") [16].* Von ebenso großer Bedeutung für einen kommerziellen Einsatz dieses Ansatzes ist jedoch die *Gewährleistung einer hohen Verfügbarkeit* des Verteildienstes bei zufälligen Störungen sowie bei vorsätzlichen Sabotageangriffen.

In der Vergangenheit wurden mit unterschiedlichen Methoden bereits effiziente oder stabile Overlays konstruiert. Die bisherigen Verfahren zur Stabilisierung und Sicherung der Qualität übertragener Daten basieren hierbei in der Regel auf dem Hinzufügen von Redundanzen oder auf der Minimierung der Auswirkung einzelner ausfallender Teilnehmer auf das Gesamtsystem. Die erstgenannte Strategie nutzt die natürliche Robustheit multimedialer Daten gegen eine geringe Paketverlustrate und zielt darauf ab, die Auswirkungen, die einzelne ausfallende Knoten auf das restliche System haben, zu minimieren. Zu diesem Zweck wird zum einen versucht, durch die Konstruktion flacher Bäume die Anzahl der Vorgänger so gering wie möglich zu halten [2]. Um zum anderen weniger abhängig von einzelnen Vorgängern zu sein, versuchen weitere Ansätze den Datenstrom über möglichst unterschiedliche Pfade zu beziehen und dadurch den Knotenzusammenhang des Overlays zu erhöhen [5,13]. Beide Verfahren verhelfen potentiellen Angreifern jedoch zu dem Wissen über die Wichtigkeit von Knoten und versetzen sie so in die Lage besonders zentrale Knoten als Ziele auszuwählen. Insgesamt richten sich die vorgeschlagenen Ansätze damit lediglich gegen zufällige Ausfälle und nicht gegen vorsätzliche Angriffe.

Aufgrund der zeitlichen Anforderungen an den Transport von Multimediadaten und der Vermeidung von unnötigen Übertragungsvorgängen im Netzinneren (bezogen auf die Übertragungsvorgänge pro Netzwerk-Link) ist es für ALM weiterhin besonders wichtig, dass sich die Strukturen der konstruierten Topologien den Strukturen der unterliegenden Transportnetze anpassen. Ineffiziente Topologien, mit einer Vielzahl von Verbindungen zwischen weit entfernten Knoten führen nicht nur zu erhöhten Ende-zu-Ende-Verzögerungen und verstärktem Schwanken dieser Verzögerung (Jitter), sondern bewirken zugleich eine stärkere, unnötige Belastung des Transportnetzes. Die Konstruktion effizienter Topologien zielt daher darauf ab, Pakete vorrangig über kurze Ende-zu-Ende-Verbindungen zu übertragen und lange Distanzen im Netz zu vermeiden.

Um global möglichst gute Ergebnisse zu erreichen, verfolgen einige Ansätze [14] die Strategie, die gesamte Topologiekonstruktion einem Verwaltungsknoten zu überlassen. Der Ausfall eines solchen verwaltenden Knotens führt jedoch zu einem Zusammenbruch des Systems, so dass dieser Knoten ein gutes und allen Teilnehmern bekanntes Ziel für Angriffe darstellt.

Weitere Ansätze implementieren die Nachbarwahl verteilt und folgen hierbei grundsätzlich einer von drei Strategien: entweder wird mit Hilfe eines eigenen Signalisierungs-Overlays zunächst die eigene Position bestimmt und die Verbin-

dung zu einem der Knoten in der eigenen Region aufgebaut, oder der Knoten tritt dem Streaming-Overlay direkt bei und wird in diesem durch den lokalen Tausch mit Nachbarn sukzessive an eine kostengünstige Position gebracht [11,12]. In hybriden Verfahren sucht jeder Knoten zuerst einen nahe liegenden Nachbarn und nach dem Beitritt werden die lokalen Verbindungen optimiert [10]. Für die erste Strategie bedarf es eines Verfahrens, um die Positionen der einzelnen Teilnehmer initial zu ermitteln. Der zuverlässigste Ansatz hierfür ist es, eine Distanzbestimmung aller Knoten untereinander durchzuführen [7]. Er ist mit wachsender Teilnehmerzahl auf Grund seiner hohen Nachrichtenkomplexität jedoch nicht mehr durchzuführen. Eine vereinfachte Lösung des Verfahrens verfolgt lediglich das Ziel, Knoten der näheren eigenen Umgebung im Netzwerk zu identifizieren, diese zu gruppieren und daraufhin Verbindungen präferiert innerhalb der entstehenden Gruppen aufzubauen [15,1]. Diese Lösungen kommen mit einem signifikant geringeren Nachrichtenaufwand aus, der jedoch zu einer geringeren Qualität der Lösung führen kann.

Indirekt kann eine Reduktion der überbrückten Distanzen auch dadurch erreicht werden, dass der ALM auf einem Peer-to-Peer-Suchmechanismus, der die Lokalitäten der Knoten berücksichtigt, aufgesetzt wird und alle Pakete durch diesen geroutet werden [5,6]. Diese Lösungen beziehen jedoch in der Regel Knoten, die den empfangenen Datenstrom selbst nicht beziehen, zur Weiterleitung ein, was für diese nicht wünschenswert ist.

In unseren eigenen Vorarbeiten auf diesem Gebiet haben wir uns bisher vor allem auf die Konstruktion von Topologien konzentriert, die möglichst stabil gegen Angriffe sind. In [3] wurde eine Klasse von Topologien für einen Spezialfall (in Bezug auf Quellenkapazität und Teilnehmeranzahl) vorgestellt und die optimale Resistenz dieser Klasse gegen Sabotageangriffe bewiesen. Weiterhin wurde in [17] ein Verfahren für die verteilte und dynamische Konstruktion stabiler Streaming-Topologien vorgeschlagen und bewertet, das jedoch Effizienzkriterien noch nicht mit einbezieht.

Eine gemeinsame Betrachtung der Kriterien Effizienz und Stabilität gegen Sabotageangriffe von Topologien existiert unseres besten Wissens nach bisher nicht. Sie ist daher das Ziel des vorliegenden Artikels.

Die weiteren Abschnitte gliedern sich wie folgt: Abschnitt 2 beinhaltet eine formale Beschreibung der Konstruktionsaufgabe und der Bewertungsmetriken für Stabilität und Effizienz. In Abschnitt 3 stellen wir unseren Ansatz zur gemeinsamen Optimierung der beiden Kriterien vor. Hierzu werden zunächst Kostenfunktionen zur Abbildung von Stabilitäts- und Effizienzeigenschaften aufgestellt und darauf aufbauend ein verteilter Algorithmus für die dynamische Topologieoptimierung beschrieben. In Abschnitt 4 beschreiben und diskutieren wir die Ergebnisse einer Simulationsstudie des Ansatzes und Abschnitt 5 fasst die Ergebnisse des Artikels kurz zusammen und gibt einen Ausblick auf zukünftige Arbeiten.

2 Stabilität und Effizienz in Streaming-Overlays

Grundlegend ist ein Overlay ein ungerichteter, schleifenfreier Graph $G = (V, E)$, bestehend aus einer endlichen Knotenmenge $V = \{v_1, \ldots, v_n\}$ (den teilnehmenden End-Systemen) und einer Menge Kanten $E \subseteq \{(u, v) | u, v \in V, u \neq v\}$ (den Verbindungen zwischen den End-Systemen).

Alle Endysteme sind in der Lage über das unterliegende Netzwerk paarweise miteinander Verbindungen aufzubauen, so dass der Graph $G = (V, E)$ im Allgemeinen zusammenhängend ist. Im konkreten Fall wählt jeder Knoten jedoch nur die Teilmenge $N_v \subseteq V$ als *Nachbarn*, wodurch in Overlays nur eine Teilmenge der Kanten vorkommt.

Da sich die Teilnehmer in Internets in paarweise unterschiedlichen Entfernungen zu einander befinden, ist zusätzlich die nichtnegative *Distanzfunktion*, die den Kosten der jeweiligen Kante entspricht, definiert: $\mathrm{d} : E \to \mathbb{R}^+$.

Zusätzlich sind die Netzzugänge der Teilnehmer in der Bandbreite beschränkt. Aus diesem Grund ist die *Knotenkapazität* wie folgt definiert: $\mathrm{c} : V \to \mathbb{R}^+$.

Im Overlay existiert eine Datenquelle $v_s \in V$, welche den Datenstrom mit der Bitrate R_0 erzeugt und als Ursprung den anderen Knoten zur Verfügung stellt. Dieser Datenstrom $\mathcal{S} = \{P_1, \ldots, P_n\}$ besteht aus n Datenpaketen, die auf jedem Knoten beliebig dupliziert und weitergeleitet werden können. Mittels Unterteilung des Datenstroms in l aufeinander folgende Sequenzen mit jeweils k Paketen und durch Zusammenfassung jedes i-ten Paketes aus allen Sequenzen kann er in k gleich große Teildatenströme, oder „*Stripes*“, unterteilt werden:

$$\mathcal{S} = \{\{p_1^1, \ldots, p_k^1\}, \ldots, \{p_1^l, \ldots, p_k^l\}\} \text{ mit } p_j^i = P_{(i-1)\cdot k+j}$$

Mit C wird die Knotenkapazität der Quelle bezeichnet: $C = c(v_s)$, die Quelle hat in Konsequenz maximal $C \cdot k$ ausgehende Kanten und die Bandbreite des Netzzugangs der Quelle beträgt damit $C \cdot R_0$.

Im allgemeinen Fall, dass alle Knoten den gesamten Datenstrom erhalten, werden die Pakete des Datenstromes entlang n Spannbäumen T_i, Out-Trees mit der Wurzel v_s, über den Graphen verteilt. Im Weiteren wird von der Unterteilung des Datenstroms in Stripes ausgegangen. In diesem Fall gilt, bei Vernachlässigung der Knotendynamik, dass die jeweils l Spannbäume eines Stripes identisch sind und lediglich k unterschiedliche Spannbäume existieren: $T_1 = (V, E_1), \ldots, T_k = (V, E_k)$. Aufgrund der Bandbreitenbeschränkung der einzelnen Knoten gilt darüber hinaus, dass die Summe der Grade in den Spannbäumen T_i für jeden Knoten $v \in V$ höchstens $c(v)$ ist:

$$\sum_{i=1}^{k} \deg_{T_i}(v) \leq c(v) \quad \text{für alle } v \in V$$

Mit den Distanzen als Kosten berücksichtigt ergeben sich die Gesamtkosten der Topologie in diesem Fall zu:

$$\mathrm{totalcost}(\mathcal{T}) = \sum_{i=1}^{k} d(T_i) = \sum_{i=1}^{k} \sum_{e \in E_i} d(e).$$

Die Sequenz $\mathcal{T} = (T_1, \ldots, T_k)$ der Spannbäume wird im Weiteren als Streaming-Topologie $\mathcal{T}$ bezeichnet.

Verlässt ein Knoten v das Overlay, so führt dies bis zur erneuten vollständigen Verbindung der Topologie zu einem Verlust der Pakete des Stripes i bei allen seinen Nachfolgern $\text{succ}_{T_i}(v)$. Die Anzahl der dadurch nicht mehr empfangenen Stripes beschreibt die Funktion $a_{T_i}(\mathcal{A}) := |\text{succ}_{T_i}(\mathcal{A}) \cup \mathcal{A}|$. Die Anzahl der durch eine Ausfallmenge $\mathcal{A}$ in der gesamten Topologie $\mathcal{T}$ verursachten Paketverluste, ergibt sich zu: $a_{\mathcal{T}}(\mathcal{A}) := \sum_{i=1}^{k} a_{T_i}(\mathcal{A})$.

Als Maß für die Robustheit eines Overlays wird damit die *Angriffsstabilität* der Topologie definiert. Sie beschreibt die Anzahl der Knoten, die aus einem Overlay entfernt werden müssen, um eine Schranke Θ_{drop} insgesamt im System verlorener Pakete zu überschreiten: Gegeben ist die Topologie $\mathcal{T}$ und eine Paketverlustschranke $\Theta_{Drop} \in (0, 1)$.
Gesucht ist die minimale Ausfallmenge $\mathcal{A} \subseteq \mathcal{T} \setminus \{v_s\}$ von Knoten, so dass gilt:

$$a_{\mathcal{T}}(\mathcal{A}) \geq \Theta_{Drop} \cdot k \cdot |V|.$$

Die Metriken Link-Stress und Path-Stretch für die Bewertung der Effizienz der konstruierten Topologien sind stark von der Struktur des unterliegenden Transportnetzes abhängig: In großen Infrastrukturnetzen, die aus vielen Netzzwischensystemen und Verbindungen zwischen diesen bestehen, überdecken sich die kürzesten Pfade zwischen unterschiedlichen Endsystemen seltener, als in kleineren Infrastrukturnetzen.

Zudem sind die direkten Netzwerkpfade zwischen der Datenquelle und vielen der Endsysteme größer als in kleinen Infrastrukturnetzen, während die Netzwerkpfade zwischen nahe liegenden Endsystemen (die etwa am gleichen Access-Router angeschlossen sind) gleich bleiben.

Diese Tatsachen führen dazu, dass sowohl der Link-Stress als auch der Path-Stretch der gleichen Overlay-Konstruktionsprozedur in großen Infrastrukturnetzen mit vielen Netzwerklinks geringer als in kleinen Infrastrukturnetzen sind.

Aus diesem Grund bedarf es einer Metrik, mit Hilfe derer exaktere und vergleichbare Aussagen über die Effizienz einer Konstruktionsprozedur getroffen werden können. An dieser Stelle wird die Anzahl der gegenüber einer optimalen Topologie zusätzlich auf Punkt-zu-Punkt-Verbindungen übertragenen Netzwerk-Pakete (Hops) als Maß für die Effizienz in Bezug auf die Netzwerkkosten vorgeschlagen.

Hierfür wird als Optimalitätsmaß eine rein theoretische Lösung ohne Berücksichtigung jedweder Bandbreitenbegrenzungen gewählt. Der so theoretisch minimale Wert lässt sich mit globalem Wissen über das gesamte Netzwerk über das Overlay-Modell berechnen: Der effizienz-optimale Verteilbaum, der die geringsten Netzwerkkosten erzeugt, entspricht dem minimalen Spannbaum (MST) über alle Knoten und kann als Referenzwert $\text{totalcost}(MST(G))$ für jedes Overlay in einem zugehörigen Infrastrukturnetzwerk angegeben werden.

Das Maß für die Netzwerkeffizienz einer Topologie ergibt sich zu:

$$\text{hop\_penalty}(G, \mathcal{T}) := \frac{\text{totalcost}(\mathcal{T})}{\text{totalcost}(MST(G))}.$$

3 Ein Ansatz zur gemeinsamen Optimierung

Ein neu zum System hinzukommender Knoten verbindet sich initial mit anderen Knoten in der Topologie, von denen er daraufhin die Datenstrompakete erhält. Zur Verbesserung der Eigenschaften der so zufällig entstehenden Topologie wird diese daraufhin optimiert. Um eine Skalierbarkeit auf große Gruppen zu ermöglichen und einen einzelnen, allen bekannten, Single-Point-of-Failure zu vermeiden, wird die Optimierung auf allen Knoten, basierend auf lokalem Wissen, verteilt implementiert.

Stabilität und Effizienz der Topologien sollen durch die eine lokale Optimierung mit Hilfe einer Gesamtkostenfunktion erreicht werden. Zur Optimierung der Topologie analysiert jeder Knoten seine aktuelle Situation und versucht die auftretenden Kosten zu minimieren. Dabei werden die Kosten aller Kanten zu den Kindern analysiert und einzelne Kanten gegebenenfalls untereinander weitervermittelt. Alle Veränderungen der Topologie sind damit von Vaterknoten initiiert. Um ein Abwägen zwischen den Optimierungszielen zu ermöglichen, werden die Kosten einer Kante in Bezug auf die Effizienz über den Faktor s gewichtet mit den Stabilitätskosten zusammengefaßt. Für die Kante des Knotens v zu seinem Kind w im Spannbaum des Stripes i ergeben sich damit die Gesamtkosten zu:

$$K_i(v, w) = s \cdot K_{stabil}(v, w, i) + (1 - s) \cdot K_{distanz}(v, w).$$

Zur Kostenminimierung müssen die alternativen Situationen überprüft werden, um den besten alternativen Vater u und dadurch die lokale Veränderung zu ermitteln, welche zur stärksten Kostensenkung führt.

Um eine Stabilität der Topologien gegen Ausfälle zu erreichen, müssen die Abhängigkeiten zwischen den Knoten minimiert werden.

Zunächst ist es für jeden Knoten wichtig, von keinem anderen Knoten große Anteile des Datenstroms zu beziehen, damit dessen Ausfall nicht zu hohem Paketverlust führt. Um dieses Ziel zu erreichen, werden der Datenstrom an der Quelle in Stripes unterteilt und für jeden Knoten Kosten für die Weiterleitung mehrerer Stripes eingeführt.

$$K_{sel}(v, i) := 1 - \frac{\text{fanout}_{T_i}(v)}{c(v)}.$$

Hierbei beschreibt $fanout_{T_i}(v)$ die Anzahl der ausgehenden Kanten des Knotens v im Spannbaum i. Durch die Minimierung dieser Kosten wird im besten Fall von jedem Knoten nur ein Stripe weitergeleitet.

Global ist es für die Topologie wichtig, dass Knoten, die im betrachteten Spannbaum Daten weiterleiten können, der Quelle möglichst nahe sind, um

die verfügbare Bandbreite zu erhöhen, und die anderen Knoten als Blätter der Quelle möglichst weit entfernt sind. Aus diesem Grund erhalten Knoten w, die den Stripe i weiterleiten können, die Kosten $K_{forw}(w,i) = 0$, anderenfalls $K_{forw}(w,i) = 1$, was dazu führt, dass bei Minimierung der Kosten die Spannbäume flach gehalten werden.

Außerdem sollten Knoten w mit vielen Nachfolgern von einem Vater v nicht tiefer gehängt werden, um die durchschnittliche Tiefe der Knoten in den Bäumen und damit die Abhängigkeit nicht zu erhöhen. Für die Angriffsstabilität ist es zusätzlich wichtig, dass der Ausfall eines bestimmten Knotens nicht zu deutlich höheren Paketverlusten führt, als der eines beliebigen anderen. Wenn einzelne Knoten mit einer hohen Anzahl von Nachfolgern $a_{\mathcal{T}}(v)$ existieren, so wird ein Angreifer in die Lage versetzt, durch das Ausschalten dieser wenigen Knoten große Teile des Systems vom Dienst zu trennen und leicht großen Schaden anzurichten. Beide Ziele werden durch die Balancierung der Topologie erreicht und es ergibt sich die Kostenfunktion:

$$K_{bal}(v,w,i) := \frac{\left(\frac{\mathrm{succ}_{T_i}(v)}{\mathrm{fanout}_{T_i}(v)} - 1\right) - \mathrm{succ}_{T_i}(w)}{\left(\frac{\mathrm{succ}_{T_i}(v)}{\mathrm{fanout}_{T_i}(v)} - 1\right)}$$

Aus diesen drei Kostenarten ergeben sich die Stabilitätskosten insgesamt zu:

$$K_{stabil}(v,w,i) = K_{sel}(v,i) + K_{forw}(w,i) + K_{bal}(v,w,i).$$

Zur Konstruktion zusätzlich netzwerkeffizienter Topologien ist es wichtig, dass alle Kanten in den Spannbäumen eine möglichst geringe Distanz $d(v,w)$ aufweisen, um die Datenstrompakete auf kurzen Wegen zu übertragen. Aus diesem Grund werden alle Distanzen in den Spannbäumen minimiert.

$$K_{distance}(v,w,u) := 1 - \frac{d(e_{(v,w)}) + d(e_{(w,u)})}{d(e_{(v,w)}) + d(e_{(v,u)})}.$$

Diese lokale Optimierung führt auch global für die Topologie $\mathcal{T}$ zu einem niedrigen $totalcost(\mathcal{T})$ und damit zu geringem Link-Stress und $hop\_penalty(\mathcal{T})$.

Da ein Vater nicht alle lokalen Stabilitätsinformationen seiner Kinder und ihrer Nachfolger kennt, wird die Gesamtoptimierung in zwei Schritten durchgeführt. Als erstes wird die Kante ausgewählt, welche die höchsten Gesamtkosten $K_i(v,w)$ verursacht. In einem zweiten Schritt wird bei der Optimierung der lokalen Situation von den Stabilitätskosten lediglich K_{bal} berücksichtigt und insgesamt so optimiert, dass der Nutzen $G_i(v,w,u) = K_i(v,w) - (s \cdot K_{bal}(v,u,i) + (1-s) \cdot K_{distance}(v,w,u))$ maximiert wird. Da durch die Optimierung die Baumhöhe lediglich erhöht werden kann, was bei einer Senkung von Stress und *totalcost* zu einer Erhöhung des Path-Stretch führt, wird als Nutzenschranke Θ_{pass} eingeführt und Optimierungen nur dann ausgeführt, wenn ihre Nutzensteigerung G diese Schwelle überschreitet. Die Entscheidung über die Weiterleitung eines Knotens kann nicht über eine konstante Schwelle parametrisiert werden, da einerseits vermieden werden muß, dass ein Knoten, der noch gar keine Bandbreite

verbraucht hat, sofort alle Knoten, die eine Verbindung zu ihm aufbauen, weiterleitet. Andererseits muß ein Knoten mit ausgelasteter Bandbreite in der Lage sein, eines seiner Kinder weiterzuleiten. Die Schwelle muß folglich durch eine Funktion über die Bandbreite berechnet werden. Sie soll mit steigender anteiliger Bandbreitennutzung eines Vaterknotens $b \in (0,1)$ stetig von 1 auf 0 fallen, wobei: $b = \frac{\deg(v)}{c(v)}$. Zusätzlich soll mit dem Gewicht t eine Optimierung auf geringeren Stretch oder geringere *totalcost* ermöglicht werden. Hierfür wurde die Funktion $\Theta_{pass}(b) = (1 - b^{(2 \cdot t)^3} \cdot (1 - t^3)) + t^3$ gewählt, wobei der Funktionswert von $\Theta_{pass}(1)$ auf $-\infty$ definiert ist. Stellt ein Knoten fest, dass er Bandbreite frei

Algorithmus 1 Topologieoptimierung auf Knoten v

Input: v, N_v
  $d \leftarrow \emptyset$ {zu entfernende Kante}; $a \leftarrow \emptyset$ {alternatives Parent}; $b \leftarrow deg(v)$
  gain $\leftarrow wahr$; i $\leftarrow$ präferierter Stripe
  while gain **do**
    gain $\leftarrow falsch$
    $d \leftarrow w$: $K(v, w, \text{Childs}_{T_i}(v), i) = \max\{K(v, w, \text{Childs}_{T_i}(v), i) \mid w \in \text{Childs}_{T_i}(v)\}$
    $a \leftarrow w$: $G(v, w, a, i) = \max\{G(v, w, a, i) \mid w \in \text{Childs}_{T_i}(v) \setminus \{d\}\}$
    if G(v,a,d,i)$\geq \Theta_{pass}$ **then**
      drop(d,a)
      gain $\leftarrow wahr$
    end if
  end while
  while $b < c(v)$ **do**
    a $\leftarrow w = rand\{\text{Childs}_{T_i}\}$
    requestChild(a,Θ_{pass}, $(\frac{\text{Succ}_{T_i}(v)}{\text{fanout}_{T_i}(v)} - 1)$)
    b $\leftarrow b + 1$
  end while
  a $\leftarrow w$: $\text{Succ}_{T_i}(w) = max\{\text{Succ}_{T_i}(w) \mid w \in \text{Childs}_{T_i}(v)\}$
  if $\text{Succ}_{T_i}(a) > \frac{\text{Succ}_{T_i}(v)}{2}$ **then**
    requestChild(a,Θ_{pass}, $(\frac{\text{Succ}_{T_i}(v)}{\text{fanout}_{T_i}(v)} - 1)$)
  end if

hat, so fordert er von einem seiner Kinder einen beliebigen Nachfolger an. Hierdurch wird erreicht, dass bei freiwerdenden Bandbreiten die Höhe der Topologie gesenkt werden kann.

Mit den so definierten Funktionen für Kosten K und die Kostensenkung G optimiert der Algorithmus zur lokalen Topologieoptimierung (vgl. Algorithmus 1) die lokale Nachbarschaft jedes teilnehmenden Knotens. Die Zeitkomplexität dieser Optimierung ist in jedem Optimierungsschritt für jeden Knoten quadratisch in der Anzahl seiner Kinderknoten im optimierten Spannbaum.

4 Analyse und Simulationsstudie

Zur Überprüfung der Hypothese, dass mit Hilfe lokaler Optimierungen auf Basis der vorgestellten Kostenfunktionen stabile und effiziente Topologien konstruiert werden können, wurde eine zweiteilige Simulationsstudie durchgeführt.

Zu diesem Zweck wurden zunächst Algorithmus und Signalisierungsprozedur im Simulationswerkzeug OMNeT implementiert und so Topologien konstruiert.

Als Backbone kam ein durch den BRITE-Topologiegenerator konstruiertes Netzwerk, welches dem Internet-Modell aus [4] folgend generiert wurde, mit 750 Netzzwischensystemen zum Einsatz. Mit den Zwischensystemen wurden gleichverteilt eine Datenstromquelle und zwischen 50 und 250 End-Systeme verbunden, die einem Nutzermodell nach [8] folgend, dem System beitreten.

Um zur Erlangung der für die Effizienzoptimierung benötigten Entfernungen eine permanente Distanzmessung zwischen den Knoten zu vermeiden, wurde ein synthetisches Koordinatensystem basierend auf Vivaldi [9] implementiert. Als Metrik kommt in der Implementierung die Anzahl der Punkt-zu-Punkt-Verbindungen (IP-Hops) im Netzwerk zum Einsatz. Die Abweichung zwischen geschätzten und gemessenen Entfernungen zum Ende der Simulationszeit betrug dabei im Maximum ca. 218% und im Schnitt ca. 45%.

Zur Untersuchung wurde der Aufbau bei gegebenem Nutzermodell simuliert und die entstehenden Topologien danach offline analysiert.

Um die Effizienz der entstandenen Topologien zu bewerten, wurden die zur Übertragung des Streams benötigten Kosten *totalcost* und die Güte in Form des *hop_penalty* ermittelt.

Dabei wurde zunächst das Stabilitätsgewicht s auf 0 festgelegt und das Gewicht t variiert. Eine Erhöhung von t führte zu insgesamt flacheren Overlay-Bäumen und dadurch zu geringerem Path-Stretch, da die Nutzenschranke, die der Nutzen einer Optimierung erbringen muß, damit sie durchgeführt wird, steigt. Parallel zur Erhöhung des Stretch sinkt der Link-Stress, da mehr Knoten an der tatsächlichen Verteilung des Datenstroms beteiligt und die Daten über mehr unterschiedliche Netzwerklinks übertragen werden.

Bei einer Senkung von t konnte außerdem ein verringertes *hop_penalty* beobachtet werden. Auch diese Beobachtung war zu erwarten, da bereits Optimierungen mit geringerem Effizienznutzen durchgeführt wurden. Eine Grenze stellte sich bei einem Wert von 0.2 ein. Wurde der Parameter unter diesen Wert gesenkt, konnten die Effizienzeigenschaften nicht stetig verbessert werden. Diese Tatsache ist damit zu erklären, dass die Optimierungen bei geringeren Werten nur einen sehr kleinen Nutzen erbringen müssen. In der verteilten Optimierung führt dies dazu, dass im Aufbau der Topologie auch lokale Optimierungen, die sich global betrachtet als ungünstig erweisen, durchgeführt werden, und die Optimierung daraufhin nur noch lokale Minima erreicht. Auf Grund der guten Ergebnisse für die Effizienz wurde der Parameter für t daraufhin auf 0.2 festgelegt.

Insgesamt konnte der minimale Spannbaum bei steigenden Gruppengrößen immer weniger gut angenähert werden und das minimale *hop_penalty* stieg stetig mit wachsender Teilnehmerzahl. Diese Tatsache ist nicht überraschend, da die Kapazitäten der Knoten bei der realen Optimierung berücksichtigt werden müssen, während sie für den minimalen Spannbaum keine Rolle spielen.

Im zweiten Schritt sollte der Trade-Off zwischen der Effizienz und der Stabilität untersucht werden, wozu das Gewicht s variiert wurde.

Die Resultate dieser Untersuchung (vgl. Abb. 1) zeigten, daß die Topologien bei niedrigem Gewicht s erwartungsgemäß effizient wurden. Mit steigendem Gewicht und zunehmender Optimierung der Topologien auf Stabilität stieg diese Anzahl in allen Simulationen erwartungsgemäß stetig an. Für alle Gruppengrößen gilt hierbei, daß die Anzahl verschickter Netzwerkpakete immer geringer war, als in einem Client-Server-Szenario.

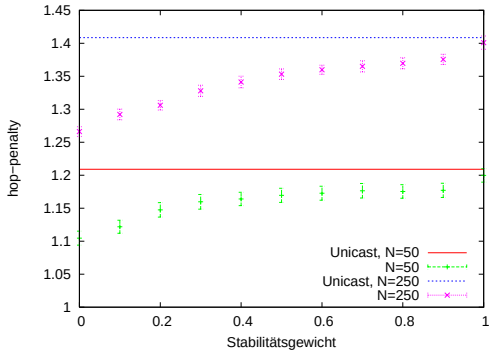

Abb. 1. Hop-Penalty entstehender Topologien im Vergleich zu Client-Server-Unicast-Streaming (16 Simulationsläufe, 98% Konfidenz)

Da die Optimierung stark von der Güte der Koordinaten abhängig ist, wurde das synthetische Koordinatensystem bei der Evaluation als Schwachstelle identifiziert. Die Dauer der Lernphase, die benötigt wurde, um die Distanzen zwischen Knoten zuverlässig genug vorhersagen zu können, lag um ein Weites höher als erwartet. Um eine Abweichung der Koordinatenschätzung unter 700% zu senken bedurfte es einer Vorlaufzeit von 300 Simulationssekunden, wobei der Nachrichtenaufwand in $\mathcal{O}(N)$ blieb.

Resultierend kann festgestellt werden, daß die Effizienzoptimierung den Erwartungen entspricht und die Topologien bei geringerer Gewichtung der Effizienz höhere Kosten im Netzwerk erzeugen.

Zur Untersuchung der Stabilitätseigenschaften der Topologien wurden sie zusätzlich auf ihre Angriffsstabilität untersucht. Zu diesem Zweck wurde ein auf globalem Wissen basierender Greedy-Angriff implementiert, der die Knoten nach der Anzahl ihrer Nachfolger aus der Topologie entfernt, und die daraus resultierende Paketverlustrate im Gesamtsystem gemessen. Die Angriffsstabilität gibt damit an, wieviele Knoten mindestens aus der Topologie entfernt werden müssen, um die insgesamt noch ausgelieferten Pakete auf eine vorgegebene Schwelle zu senken. Im Ergebnis zeigt sich somit der maximale durch korrelierten Knotenausfall auftretende Paketverlust im Gesamtsystem für den Zeitraum bis zur Reparatur der Topologie.

Diese zweite Auswertung zeigte, daß sich die Topologien bei einer reinen Stabilitätsoptimierung ($s = 1.0$) einer optimalen Topologie annäherten und die Paketverluste den Ergebnissen aus vorherigen Arbeiten entsprachen [3].

Eine Senkung von s führte wie erwartet zu einer Verringerung der Stabilität der Topologien (vgl. Abb. 2). Allerdings konnten starke Verringerungen in der Stabilität der Topologien erst ab Gewichtungen von $s < 0.2$ beobachtet werden.

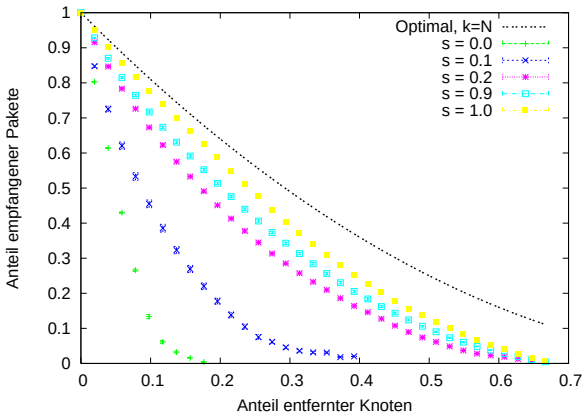

Abb. 2. Minimaler Anteil empfangener Pakete im System nach Anteil entfernter Knoten bei unterschiedlicher Stabilitätsgewichtung (16 Simulationsläufe, 98% Konfidenz)

Bereits mit geringer Gewichtung s führte die Optimierung folglich zu Topologien mit hoher Knotenkonnektivität und balancierten Bäumen.

Eine zusätzliche Untersuchung zeigte darüber hinaus, dass eine weitere Variation des Gewichtes t bei beliebigen s erwartungsgemäß zu keiner weiteren Effizienzsteigerung führte.

Zusammenfassend ist festzustellen, dass der Algorithmus mit einer Gewichtung von $s = 0.2$ und $t = 0.2$ zu sowohl angriffsstabilen als auch netzwerkeffizienten Topologien führt.

5 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde ein formales Modell für die Analyse von Overlay-Streaming-Systemen aufgestellt. Mit Hilfe des Modells wurde als neue Effizienzmetrik $hop\_penalty(G, \mathcal{T})$, das Verhältnis der Distanzsummen von konstruierter Topologie $\mathcal{T}$ und einem minimalen Spannbaum in G, eingeführt.

Zur Konstruktion von Topologien, die einen guten Kompromiss zwischen Angriffsstabilität und Netzwerkeffizienz realisieren, wurde ein verteilter Algorithmus beschrieben, der durch die Verbindung der Effizienz- und Stabilitätskosten in einer gewichteten Summe eine Abwägung zwischen beiden Optimierungszielen ermöglicht.

In einer Simulationsstudie wurde gezeigt, dass der Algorithmus trotz einer Beschränkung auf lokales Wissen mit Hilfe der Kostenoptimierung in der Lage ist, stabile und effiziente Topologien zu konstruieren. Dabei stieg den Erwartungen entsprechend die Angriffsstabilität bei einer erhöhten Gewichtung der Stabilität und sank das $hop\_penalty$ der Topologien bei einer erhöhten Gewichtung der Effizienz. Als Gewicht für die Kostensumme wurde ein Bereich ermittelt,

in dem die Topologien sowohl gute Effizienz- als auch Stabilitätseigenschaften aufweisen. In allen Simulationsszenarien lag das *hop_penalty* dabei unter dem *hop_penalty* eines Client-Server-Unicast für die gleiche Knotengruppe.

Offene Fragen für zukünftige Arbeiten sind die Bewertung der Stabilität gegenüber zufälligen Ausfällen von Knoten, für die die Angriffsstabilität lediglich eine obere Schranke angibt, sowie die Analyse der Signalisierungsprozedur auf Schwachstellen gegenüber vorsätzlichen Angriffen.

Literatur

[1] Banerjee, S. ; Bhattacharjee, B. ; Kommareddy, C.: Scalable application layer multicast. In: *ACM Computer Communication Review*, 2002, S. 205–217

[2] Birrer, S. ; Lu, D. ; Bustamante, F.E. ; Qiao, Y. ; Dinda, P.: FatNemo: Building a resilient multi-source multicast fattree. In: *WCCD*, 2004, S. 182–196

[3] Brinkmeier, M. ; Schäfer, G. ; Strufe, T.: *A Class of Optimal Stable P2P-Topologies for Multimedia-Streaming.* 2007. – submitted to IEEE INFOCOM'07

[4] Bu, T. ; Towsley, D.: On distinguishing between Internet power law topology generators. In: *INFOCOM Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies* Bd. 2, 2002 (Proceedings), S. 638–647

[5] Castro, M. ; Druschel, P. ; Kermarrec, A. ; Nandi, A. ; Rowstron, A. ; Singh, A.: SplitStream: High-bandwidth content distribution in a cooperative environment. In: *Proceedings of (IPTPS'03)*, 2003, S. 298–313

[6] Castro, M. ; Druschel, P. ; Kermarrec, A.M. ; Rowstron, A.: Scribe: a large-scale and decentralized application-level multicast infrastructure. In: *IEEE JSAC* 20 (2002), Nr. 8, S. 1489 – 1499

[7] Chu, Y. H. ; Rao, S. G. ; Seshan, S. ; Zhang, H.: A Case for End System Multicast. In: *IEEE JSAC* 20 (2002), Oktober, Nr. 8, S. 1456–1471

[8] Costa, C. ; Cunha, I. ; Borges, A. ; Ramos, C. ; Rocha, M. ; Almeida, J. ; Ribeiro-Neto, B.: Analyzing client interactivity in streaming media. In: *Proceedings of World Wide Web*, 2004, S. 534–543

[9] Dabek, F. ; Cox, R. ; Kaashoek, F. ; Morris, R.: Vivaldi: A Decentralized Network Coordinate System. In: *Proceedings of the ACM SIGCOMM'04*, 2004

[10] Francis, P.: *Yoid: Extending the internet multicast architecture.* 2000

[11] Jannotti, J. ; Gifford, D. ; Johnson, K. ; Kaashoek, M. ; O'Toole, J.: Overcast: Reliable Multicasting with an Overlay Network. In: *Proceedings of the Symposium on Operating System Design and Implementation*, 2000, S. 197–212

[12] Li, Z. ; Mohapatra, P.: HostCast: a new Overlay Multicasting Protocol. In: *IEEE ICC*, 2003, S. 702 – 706

[13] Liang, J. ; Nahrstedt, K.: DagStream: Locality Aware and Failure Resilient Peer-to-Peer Streaming. In: *Proceedings of MMCN* Bd. 6071, 2006. – to appear

[14] Padmanabhan, V. ; Wang, H. ; Chou, P. ; Sripanidkulchai, K.: Distributing streaming media content using cooperative networking. In: *Proceedings of ACM/IEEE NOSSDAV*, 2002, S. 177–186

[15] Ratnasamy, S. ; Handley, M. ; Karp, R. ; Shenker, S.: Topologically-Aware Overlay Construction and Server Selection. In: *Proceedings of IEEE INFOCOM* Bd. 3, 2002, S. 1190 – 1199

[16] Shimbel, A.: Structural parameters of communication networks. In: *Bulletin of Mathematical Biophysics* 15 (1953), S. 501 – 507

[17] Strufe, T. ; Wildhagen, J. ; Schäfer, G.: Towards the construction of Attack Resistant and Efficient Overlay Streaming Topologies. In: *Proceedings of STM*, 06