Dienstplatzierung in Ad-hoc-Netzen

Georg Wittenburg

Institut National de Recherche en Informatique et en Automatique (INRIA)
Laboratoire d'Informatique de L'École Polytechnique
Route de Saclay, 91128 Palaiseau (CEDEX), France
georg.wittenburg@inria.fr

Abstract: Weit verbreiteten Diensten wie dem World Wide Web (WWW) oder E-Mail liegt die Client/Server-Architektur zugrunde. Diese ist jedoch nur eingeschränkt auf drahtlose Ad-hoc-Netze übertragbar. In der aktuellen Forschung werden daher Alternativen untersucht, die die Diensterbringung intelligent auf einer Mehrzahl von am Ad-hoc-Netz teilnehmenden Geräten verteilen. Das in dieser Arbeit vorgestellte System SP*i* ermöglicht erstmalig den experimentellen Vergleich von unterschiedlichen Ansätzen zur verteilten Diensterbringung. Die experimentelle Untersuchung von SP*i* und dem ebenfalls vorgestellten Graph Cost / Multiple Instances Dienstplatzierungsalgorithmus zeigt, dass die verteilte Diensterbringung als neuartige Architektur die Skalierbarkeit von Diensten in Ad-hoc-Netzen entscheidend verbessert.

1 Einleitung

Die Erbringung von Diensten wie dem Domain Name System (DNS), dem World Wide Web (WWW) oder E-Mail in drahtlosen Ad-hoc-Netzen stellt eine Herausforderung dar, weil die zugrunde liegende Client/Server-Architektur eine hohe Netzlast auf dem Funkkanal am dienstanbietenden Gerät bedingt. Dieser Umstand stellt eine grundsätzliche Skalierbarkeitsgrenze bei der Diensterbringung in Ad-hoc-Netzen dar [LBC+01]. In aktuellen Forschungsansätzen wird daher der Ansatz verfolgt, die Diensterbringung auf mehrere am Netz teilnehmende Knoten zu verteilen und so den für einen Dienst erforderlichen Datenverkehr gleichmäßig im Netz zu verteilen [HCB02, LSO+07, Her10]. Die zentrale Komponente zur Diensterbringung, der Server, wird dabei in eine variable Anzahl von Dienstinstanzen aufgeteilt, die frei auf den teilnehmenden Knoten des Ad-hoc-Netzes platziert werden können. Dieser Prozess wird von einer neuen Komponente, dem *Dienstplatzierungssystem*, gesteuert.

In der bisherigen Forschung wurde das Dienstplatzierungsproblem in Ad-hoc-Netzen zumeist entweder im Zuge der Entwicklung von Middleware-Architekturen oder als ein Anwendungsfall der Facility Location Theory [MF90] betrachtet. Beide Ansätze beschäftigen sich jedoch zumeist jeweils nur einen Teilbereich des Problems [WS08]: Middleware-basierte Ansätze konzentrierend sich auf die netztechnischen Aspekte der Dienstplatzierung und überlassen die Auswahl der Dienstinstanzen einfachen Heuristiken. Ansätze aus der Facility Location Theory bieten qualitativ hochwertige Lösungen des Platzierungspro-

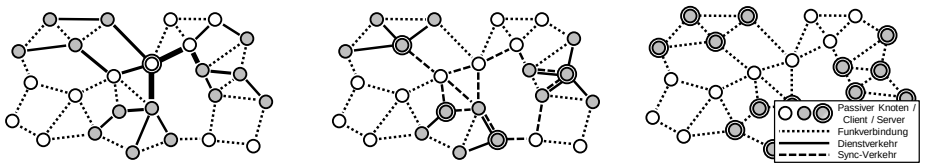
blems an, vernachlässigen dabei aber in der Regel die zusätzliche Netzlast, die durch das Dienstplatzierungssystem selbst erzeugt wird. Darüber hinaus gibt es bisher mangels entsprechender Werkzeuge weder aussagekräftige Messungen bezüglich der positiven Auswirkungen von Dienstplatzierungssystemen allgemein noch qualitative oder quantitative Vergleiche der Ansätze untereinander.

Unser Ansatz in diesem Forschungsfeld ist das SP*i* Dienstplatzierungssystem[1] [WS10, Wit10], ein neuartiger Ansatz zur Dienstplatzierung in Ad-hoc-Netzen. Der wissenschaftliche Beitrag dieser Arbeit stellt sich hierbei wie folgt dar:

- Mit einer leichtgewichtigen **Programmierabstraktion für verteilte Systeme** schaffen wir die Grundlage dafür, Softwarekomponenten und Protokolle auf einer Vielzahl von Evaluationsumgebungen zu betrachten, darunter Geräte mit gängigen Betriebssystemen wie Microsoft Windows und Linux, der Netzsimulator `ns-2` und eingebettete System-on-a-Chip-Plattformen wie sie in drahtlosen Sensornetzen zum Einsatz kommen.
- Darauf aufbauend entwickeln wir das **SP*i* Dienstplatzierungssystem** selbst, das grundlegende Protokolle zur Dienstplatzierung erstmalig effizient implementiert und über eine API zur Entwicklung von Dienstplatzierungsalgorithmen zugänglich macht. Durch diese Trennung von netztechnischen und algorithmischen Aspekten des Dienstplatzierungsproblems wird es in der Forschung nun möglich, Algorithmen leichtgewichtig zu implementieren und sinnvoll miteinander zu vergleichen.
- Wir entwickeln außerdem die **Graph Cost / Single Instance (GCSI) und Graph Cost / Multiple Instances (GCMI) Dienstplatzierungsalgorithmen**. Diese Algorithmen gehen in zweierlei Hinsicht über den aktuellen Stand der Forschung hinaus: Zum Einen betrachten sie explizit die Kommunikation zwischen den Dienstinstanzen, die erforderlich ist, um den globalen Zustand des Dienstes über seine Instanzen hinweg synchron zu halten. Außerdem berücksichtigen die beiden Algorithmen bei der Berechnung des optimalen Zeitpunktes zur Anpassung der Dienstinstanzen die zu erwartende Netzlast der Maßnahmen, die für diese Anpassung erforderlich sind.
- In einer umfassenden **experimentellen Untersuchung von Dienstplatzierungsalgorithmen** vergleichen wir acht Platzierungsalgorithmen miteinander – neben GCSI und GCMI sechs weitere Ansätze aus der aktuellen Forschung – und betrachten, inwiefern diese Ansätze in der Lage sind, die Skalierbarkeit der Diensterbringung in Ad-hoc-Netzen grundsätzlich zu verbessern. Somit enthält diese Arbeit die ersten belastbaren Messergebnisse bezüglich der Vorteilte der verteilten Diensterbringung in Ad-hoc-Netzen.

Nach einer kurzen Einführung in das Dienstplatzierungsproblem in Ad-hoc-Netzen in Abschnitt 2 stellen wir anschließend in Abschnitt 3 das SP*i* Dienstplatzierungssystem mit seinen Hauptkomponenten und dem GCMI Dienstplatzierungsalgorithmus vor. Danach geben wir in Abschnitt 4 einen Überblick über das Vorgehen und die Resultate unserer experimentellen Evaluation, um dann in Abschnitt 5 mit einer zusammenfassenden Betrachtung unserer Ergebnisse zu enden.

[1] `http://cst.mi.fu-berlin.de/projects/SPi/`



(a) Hoher Synchronisationsbedarf; einzelne Dienstinstanz im Netz

(b) Mittlerer Synchronisationsbedarf; mehrere regionale Instanzen

(c) Keine Synchronisation; eine Instanz pro dienstnehmenden Knoten

Abbildung 1: Dienstkonfigurationen für unterschiedliche Synchronisationsanforderungen

2 Dienstplatzierung in Ad-hoc-Netzen

Die Aufgabe eines Dienstplatzierungssystems für Ad-hoc-Netze besteht in der Auswahl einer optimalen Menge von dienstanbietenden Knoten angesichts der aktuellen Nachfrage nach dem Dienst und der gegenwärtigen Netztopologie. Diese Problemstellung umfasst folgende drei Kernfragen: *Wie viele* Instanzen eines Dienstes sollen im Netz zur Verfügung stehen, um gemeinsam die Anfragen der dienstnehmenden Knoten abzuarbeiten; *wo* sollen diese Dienstinstanzen platziert werden, d.h. welche Knoten sind für die verteilte Dienster-bringung am besten geeignet; und *wann* soll eine Dienstkonfiguration angepasst werden. Hierbei sind die *Dienstinstanzen* eines verteilt arbeitenden Dienstes exakte Kopien der Softwarekomponente, die den Dienst erbringt — einschließlich des ausführbaren Programms und der Anwendungsdaten. Die Menge der Knoten im Ad-hoc-Netz, die jeweils eine Dienstinstanz beherbergen, wird *Dienstkonfiguration* genannt. Stellt ein Dienstplatzierungssystem eine gute Dienstkonfiguration her, so ist hierdurch eine Verbesserung der Qualität der Diensterbringung bei gleichzeitiger Verminderung der Netzlast zu erwarten.

Neben der Netztopologie und der Nachfrage nach dem Dienst ist der *Synchronisationsbedarf* zwischen Dienstinstanzen von zentraler Bedeutung für die Wahl der Dienstkonfiguration [YV01]. Der Synchronisationsbedarf bedingt sich durch die Notwendigkeit den globalen Zustand des Dienstes, z.B. in Form anwendungsspezifischer Daten, zwischen den Dienstinstanzen konsistent zu halten. Wie in Abbildung 1 veranschaulicht, ergeben sich aus unterschiedlichen, dienstspezifischen Synchronisationsanforderungen entsprechend unterschiedliche Dienstkonfigurationen: Bei einem hohen Synchronisationsbedarf (Abb. 1a), wie er beispielsweise bei einer transaktionalen Datenbank auftritt, ist es nicht lohnenswert, den Dienst aufzuteilen, weil sich durch das Volumen des Synchronisationsverkehrs zwischen den Instanzen eine insgesamt höhere Netzlast ergäbe als bei einer einzelnen Dienstinstanz. Für den entgegengesetzten Extremfall eines Dienstes ohne Synchronisationsbedarf (Abb. 1c), wie beispielsweise einer Rechtschreibprüfung, besteht die optimale Dienstkonfiguration aus einer Dienstinstanz auf jedem dienstnehmenden Knoten, da somit alle Dienstanfragen lokal bearbeitet werden können und jegliche Netzlast entfällt.

Der im Kontext dieser Arbeit interessantere Fall eines moderaten Synchronisationsbedarfs (Abb. 1b) tritt auf, wenn keine strikten Konsistenzanforderungen bezüglich der Anwendungsdaten des Dienstes bestehen und die Anzahl der lesenden Zugriffe auf die Daten wesentlich höher ist als die der schreibenden Zugriffe. Dieser Fall trifft auf weit verbreite-

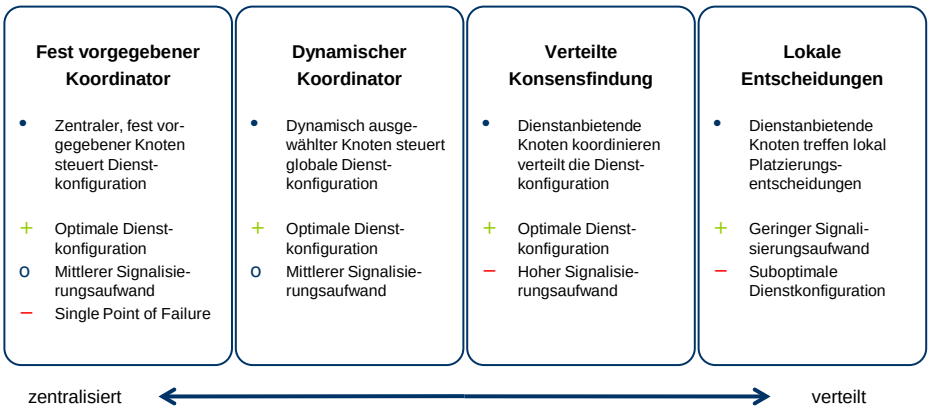


Abbildung 2: Designalternativen für Dienstplatzierungssysteme

te Dienste wie beispielsweise das Domain Name System (DNS) oder das World Wide Web (WWW) zu, sodass in diesem und vergleichbaren Fällen ein Dienstplatzierungssystem zur Optimierung eingesetzt werden kann. Das Ziel der Dienstplatzierung ist beispielsweise eine Dienstkonfiguration, die die durch die Diensterbringung bedingte Netzlast minimiert. Je nach Dienst und Anwendungsfall sind aber auch andere Optimierungen denkbar.

Aus theoretischer Sicht ähnelt die Optimierung der Dienstkonfiguration dem Uncapacitated Facility Location (UFL) Problem aus der Facility Location Theory [MF90]. Im Gegensatz zum Dienstplatzierungsproblem modelliert das UFL-Problem jedoch nicht den Synchronisationsverkehr zwischen Dienstinstanzen, betrachtet keine zeitlich veränderlichen Eingabedaten (z.B. Dienstnachfrage und Netztopologie) und berücksichtigt nicht die durch die Ermittlung der Eingabedaten bedingte zusätzliche Netzlast. Lösungsansätze für das UFL-Problem sind folglich nicht direkt auf das Dienstplatzierungsproblem übertragbar, stellen jedoch aufgrund der ähnlichen Struktur der Problemstellung einen Ausgangspunkt für die Forschung dar. In der Tat basiert der in Abschnitt 3 vorgestellte GCMI Dienstplatzierungsalgorithmus auf solch einem Ansatz von Domínguez et al. [DMJ03].

Wie in Abbildung 2 dargestellt, sind Lösungsansätze für das Dienstplatzierungsproblem grundsätzlich in vier Klassen einzuteilen: Vollständig verteilte Ansätze verwenden regelbasierte, *lokale Entscheidungen* (z.B. [Her10]). Der wesentliche Vorteil dieser Ansätze ist der sehr geringe Bedarf an Signalisierungsverkehr; allerdings leidet mangels globaler Information über das Gesamtnetz die Qualität der erzielten Dienstkonfigurationen. Qualitativ höherwertige Konfigurationen erzielen Ansätze, die eine *verteilte Konsensfindung* implementieren (z.B. [LSO+07]). Diese Algorithmen sind zumeist rundenbasiert und benötigen mehrere Iterationen, um zu einer optimalen Lösung zu konvergieren. Sie verursachen daher einen nicht zu unterschätzenden Signalisierungsverkehr. Gänzlich zentralisierte Ansätze, die einen *fest vorgegebenen Koordinator* einsetzen (z.B. [HCB02]), haben keinen der Nachteile der zuvor genannten Ansätze. Da jedoch ein spezieller Knoten den Algorithmus ausführt, ergibt sich somit ein Single Point of Failure für das Dienstplatzierungssystem.

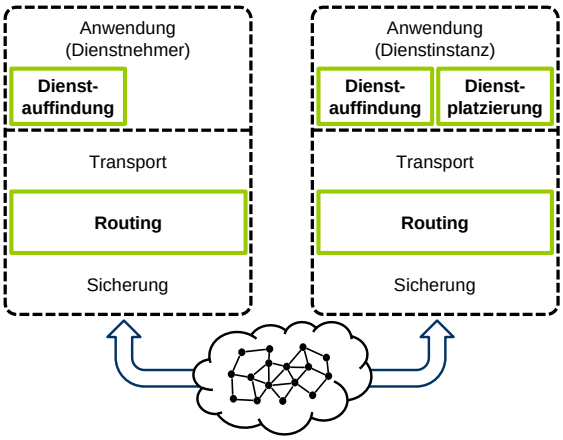


Abbildung 3: Komponenten des SP*i* Dienstplatzierungssystems

3 Dienstplatzierung mit SP*i*

Angesichts der im vorangegangenen Abschnitt besprochenen Vor- und Nachteile bisheriger Ansätze, haben wir ein Dienstplatzierungsalgorithmus entwickelt, der auf einen *dynamisch zugewiesenen Koordinatorknoten* ausgeführt wird (vgl. Abb. 2). Durch diesen neuartigen Ansatz lassen sich sowohl das Ziel einer qualitativ hochwertigen Dienstkonfiguration als auch jenes eines handhabbaren Signalisierungsverkehrs realisieren. Die bekannten Nachteile eines zentralisierten Ansatzes (insbesondere Skalierbarkeit und Ausfallsicherheit) lassen sich im Rahmen der vorgestellten Architektur vermeiden, indem in großen Netzen eine baumartige Hierarchie von Koordinatorknoten eingesetzt wird und einzelne Koordinatoren im Fehlerfall von den Dienstinstanzen neu gewählt werden.

3.1 Das SP*i* Dienstplatzierungssystem

Das SP*i* Dienstplatzierungssystem [WS10, Wit10] verfolgt einen Cross-Layering-Ansatz. Wie in Abbildung 3 dargestellt, umfasst es Komponenten für Dienstplatzierung, Dienstauffindung und Routing von Dienstanfragen. Dies liegt in der Tatsache begründet, dass ein Dienstplatzierungssystem zwangsläufig die exakte Funktionsweise des Routings und der Dienstauffindung berücksichtigen muss, um eine effiziente Dienstkonfiguration herstellen zu können.

Die *Dienstplatzierungskomponente* erhebt statistische Daten bezüglich der Nachfrage nach dem Dienst und implementiert die notwendigen Protokolle zu deren Migration und Replikation. Die *Dienstauffindungskomponente* lokalisiert Dienstinstanzen und wählt zwischen den gefundenen Instanzen die für den dienstnehmenden Knoten am günstigsten gelegene aus. Die *Routingkomponente* übermittelt Dienstanfragen und -antworten zwischen dienstnehmenden und dienstanbietenden Knoten. Sie verwendet dabei erweiterte Routing-Pakete, um Informationen bezüglich der regionalen Netztopologie zu sammeln.

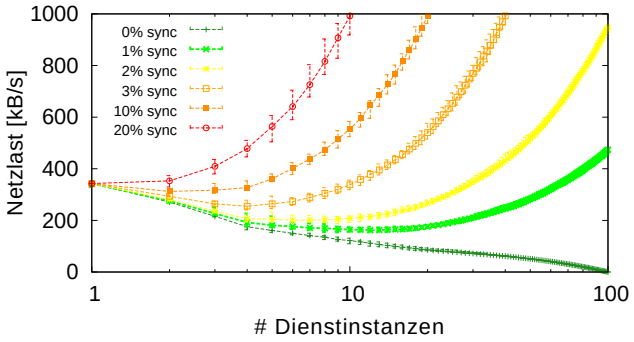


Abbildung 4: Durch die Diensterbringung bedingte Netzlast

SP*i* nutzt die wechselseitigen Abhängigkeiten zwischen Dienstplatzierung, Dienstauffindung und Routing aus, um den für die Dienstplatzierung erforderlichen Signalisierungsverkehr zu minimieren. Die Funktionalität der drei Komponenten wird über eine API gekapselt, die den einfachen Zugriff auf die aggregierten Daten bezüglich Netztopologie und Dienstnachfrage ermöglicht. Außerdem können über diese API die zur Anpassung der Dienstkonfiguration benötigten Aktionen angestoßen werden. Aufbauend auf dieser Grundlage sind nun diverse Dienstplatzierungsalgorithmen effizient implementierbar.

3.2 Der GCMI Dienstplatzierungsalgorithmus

Der Graph Cost / Multiple Instances (GCMI) Dienstplatzierungsalgorithmus ist einer von acht Algorithmen, die wir aufbauend auf der Funktionalität des SP*i* Dienstplatzierungssystems implementiert haben. Basierend auf den von SP*i* erhobenen Daten bezüglich Netztopologie und Dienstnachfrage berechnet GCMI die durch die Diensterbringung bedingte Netzlast als Summe aus anfragebedingtem Verkehr und Synchronisationsverkehr. Der dienstspezifische Synchronisationsbedarf wird hierbei als Verhältnis zwischen schreibenden und lesenden Dienstanfragen modelliert.

Die Netzlast als konfigurationsspezifische Kostenfunktion ist in Abbildung 4 für ein Ad-hoc-Netz bestehend aus 100 zufällig platzierten Knoten dargestellt. Deutlich zu erkennen ist in dieser Abbildung zunächst einmal der Spezialfall für einen Dienst ohne Synchronisationsbedarf (“0% sync”): In diesem Fall ist die Netzlast am niedrigsten, wenn auf jedem der 100 Knoten eine Dienstinstanz verfügbar ist (vgl. auch Abb. 1c). Bei höherem Synchronisationsbedarf ergeben sich unterschiedliche Minima der Kostenfunktion, die jeweils der optimalen Dienstkonfiguration für das gegebene Szenario entsprechen. Erwartungsgemäß ist hierbei festzuhalten, dass bei höherem Synchronisationsbedarf die Anzahl der Dienstinstanzen in den optimalen Dienstkonfigurationen abnimmt bis schließlich der Dienst wieder von einer einzelnen Instanz erbracht wird (vgl. Abb. 4, “20% sync”).

Der Berechnung der optimalen Dienstkonfiguration in GCMI liegt die Beobachtung zugrunde, dass eine jede Dienstkonfiguration ein Clustering in dem als Graph modellierten Ad-hoc-Netz bedingt, wobei jeder dienstnehmende Knoten dem ihm nächstgelegenen

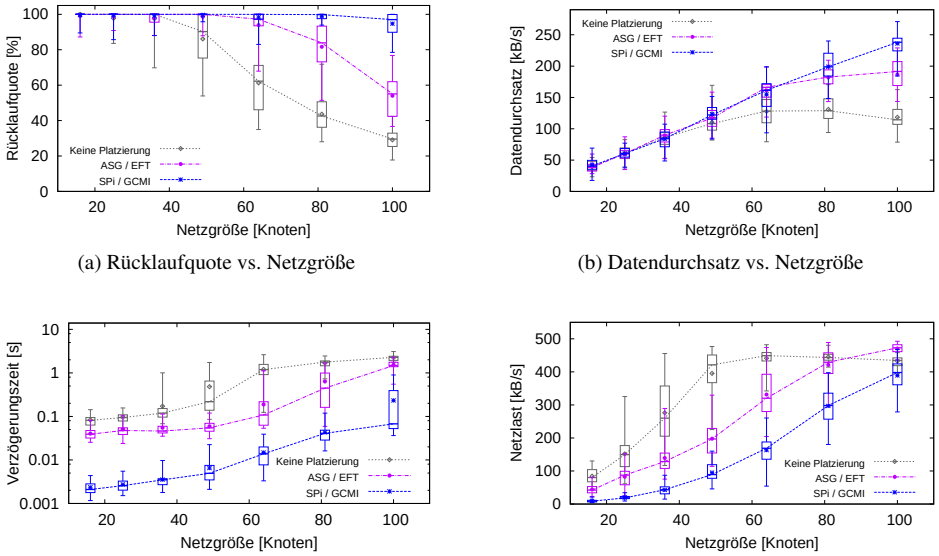
dienstanbietenden Knoten zugeordnet wird. Basierend auf diesem Ansatz ermittelt GCMI die optimale Dienstkonfiguration beginnend mit der korrekten, aber im Regelfall suboptimalen Dienstkonfiguration mit einer Instanz pro dienstnehmendem Knoten. Mit dieser Konfiguration als Ausgangspunkt wird nun iterativ jeweils das benachbarte Clusterpaar mit der geringsten gemeinsamen Dienstnachfrage zu einem Cluster verschmolzen. In jedem Iterationsschritt wird die dienstbedingte Netzlast berechnet, wobei die zentralen Knoten aller Cluster als Dienstinstanzen betrachtet werden. Tritt bei dieser Berechnung das globale Minimum der dienstbedingten Netzlast auf, so ergibt sich aus den entsprechenden Dienstinstanzen die optimale Dienstkonfiguration. Diesen Prozess kann man sich intuitiv als eine Bewegung entlang einer Kostenfunktion von vielen hin zu wenigen Dienstinstanzen vorstellen (vgl. Abb. 4). Die Berechnung terminiert sobald sich aus dem iterativen Verschmelzen der benachbarten Cluster ein einzelner letzter Cluster ergeben hat, der alle dienstnehmenden Knoten des Netzes beinhaltet.

Sobald die optimale Dienstkonfiguration ermittelt wurde, stellt sich die Frage, ob die aktuelle Dienstkonfiguration in die optimale überführt werden sollte. Da die Anpassung der Dienstkonfiguration selbst eine signifikante Netzlast erzeugt, muss hierbei sichergestellt werden, dass sich diese Investition bezahlt macht. Hierzu berechnet GCMI zunächst die Liste der zur Anpassung erforderlichen Aktionen (Migration, Replikation und Auflösung von Dienstinstanzen) und dann basierend auf dieser Liste die resultierende Netzlast. Diese anpassungsbedingte Netzlast wird nun mit der Differenz zwischen den jeweiligen Netzlasten der aktuellen und der ermittelten optimalen Dienstkonfiguration verglichen. Fällt dieser Vergleich zu Gunsten der Anpassung der Dienstkonfiguration aus, so wird diese durch Übermittlung der erforderlichen Aktionen an die derzeitigen Dienstinstanzen umgesetzt. Andernfalls wird der GCMI Algorithmus zu einem späteren Zeitpunkt erneut ausgeführt.

4 Experimentelle Evaluation

Wir untersuchen die Leistungsfähigkeit des SP*i* Dienstplatzierungssystems in Simulationen und Experimenten auf einem drahtlosen IEEE 802.11 Testbed. Dabei vergleichen wir die im Rahmen dieser Arbeit entwickelten Dienstplatzierungsalgorithmen mit anderen Ansätzen aus der aktuellen Forschung.

Für die Evaluation verwenden wir folgende vier Metriken: Die *Rücklaufquote* entspricht dem Anteil der erfolgreichen Dienstanfragen am Gesamtanfragevolumen. Gute Dienstplatzierungsalgorithmen verbessern diese Metrik, da sie Dienstinstanzen in der Nähe der dienstnehmenden Knoten platzieren und somit Problemen bei der drahtlosen Kommunikation über mehrere Hops vorbeugen. Der *Datendurchsatz* gibt das Gesamtvolumen des Netzverkehrs zwischen dienstnehmenden und dienstanbietenden Knoten pro Zeit an. Ebenso wie bei der Rücklaufquote ist hier eine Steigerung zu erwarten, wenn gute Dienstplatzierungsalgorithmen zum Einsatz kommen. Die *Verzögerungszeit* entspricht der Zeit die verstreicht bis ein dienstnehmender Knoten eine Antwort auf eine Dienstanfrage erhält. Analog zu den beiden zuvor genannten Metriken ist hier eine Reduktion zu erwarten, wenn bessere Dienstkonfigurationen erzielt werden. Die *Netzlast* gibt schließlich das Gesamtvolumen des Netzverkehrs pro Zeit an. Dieses umfasst dienstbedingten Verkehr, Si-



(a) Rücklaufquote vs. Netzgröße

(b) Datendurchsatz vs. Netzgröße

(c) Verzögerungszeit vs. Netzgröße

(d) Netzlast vs. Netzgröße

Abbildung 5: Dienstplatzierung für unterschiedlich große Netze

gnalisierungsverkehr des Dienstplatzierungssystems und Verkehr, der bei den Anpassungen der Dienstkonfiguration anfällt. Werden Dienstplatzierungsentscheidungen intelligent getroffen, so ist auch hier mit einer Reduktion zu rechnen.

Zunächst untersuchen wir die Skalierbarkeit von Diensten bezüglich der Größe des Ad-hoc-Netzes unter Verwendung unterschiedlicher Dienstplatzierungsalgorithmen. Wir betrachten hierbei ein mit dem Netzsimulator `ns-2` simuliertes IEEE 802.11 Netz mit einer variablen Anzahl von zufällig platzierten Knoten. In diesem Netz kommen bei konstantem Knotengrad und einer proportional zur Netzgröße wachsenden Dienstnachfrage zwei unterschiedliche Dienstplatzierungsalgorithmen zum Einsatz: der in Abschnitt 3.2 vorgestellte SPi / GCMI-Algorithmus und der von Herrmann [Her10] vorgeschlagene Algorithmus ASG / Event Flow Tree (ASG / EFT). Zum Vergleich betrachten wir auch das gleiche Szenario unter Verwendung einer Client/Server-Architektur ohne Dienstplatzierung.

Die Ergebnisse dieses Experiments sind in Abbildung 5 dargestellt. Jedem Datenpunkt liegen 30 Simulationsläufe zugrunde, von denen wir Mittelwerte, Minima und Maxima sowie unteres und oberes Quartil aller Messungen zeigen. Wir können feststellen, dass SPi / GCMI Rücklaufquote (Abb. 5a) und Datendurchsatz (Abb. 5b) im Vergleich zu sowohl ASG / EFT als auch gegenüber der Client/Server-Architektur substantiell verbessert. Im Gegensatz zu den beiden letzteren Alternativen stößt ein mit SPi / GCMI platzierter Dienst aufgrund der höheren Qualität der erzielten Dienstkonfigurationen erst wesentlich später an die Kapazitätsgrenze des Ad-hoc-Netzes. Damit einher geht insbesondere auch eine Reduktion der Verzögerungszeit (Abb. 5c) um mehr als eine Größenordnung. Bezüglich der Netzlast (Abb. 5d) ist festzuhalten, dass beide Dienstplatzierungsalgorithmen

Tabelle 1: Durchschnittswerte der Evaluation auf dem IEEE 802.11 Testbed

	Keine Platzierung	**Zentrale Instanz**		**SPi / GCMI**	
		absolut	**relativ**	**absolut**	**relativ**
Rücklaufquote [%]	59.21	67.20	**13.49%**	81.23	**37.20%**
Datendurchsatz [kB/s]	4.70	4.89	**4.06%**	6.12	**30.16%**
Verzögerungszeit [s]	0.1840	0.1596	**-13.29%**	0.0881	**-52.13%**
Netzlast [kB/s]	29.53	26.87	**-9.02%**	22.98	**-22.19%**

diese gegenüber der Client/Server-Architektur trotz Signalisierungs- und Anpassungsverkehr reduzieren. Auch hier erzielt SPi / GCMI durchgängig die besseren Ergebnisse.

In einem weiteren Experiment untersuchen wir die Auswirkungen verteilter Diensterbringung in einem realen Ad-hoc-Netz. Hierfür verwenden wir 36 Knoten des IEEE 802.11 DES-Testbeds am Institut für Informatik der Freien Universität Berlin [BGJS10]. Wir variieren die Dienstnachfrage zwischen 0,1 und 5 Anfragen pro Sekunde pro Knoten und betrachten die Güte der Diensterbringung eines mit SPi / GCMI platzierten Dienstes. Zum Vergleich verwenden wir wiederum einen Dienst ohne Platzierung und zusätzlich einen weiteren Dienst, der mit einer einzelnen, dynamisch platzierten Instanz arbeitet.

Tabelle 1 enthält für jede Metrik die arithmetischen Mittelwerte aus jeweils zehn Messungen für fünf Nachfrageraten. Zusätzlich geben wir jeweils die relative Veränderung gegenüber dem Szenario ohne Dienstplatzierung an. Wir beobachten, dass bereits die Platzierung einer einzelnen Dienstinstanz zu einer Verbesserung der Dienstqualität bei einer gleichzeitigen Reduktion der Netzlast führt. Dieser Effekt ist jedoch wesentlich ausgeprägter, wenn mit SPi / GCMI eine verteilte Diensterbringung mit dynamisch platzierten Dienstinstanzen zum Einsatz kommt.

5 Zusammenfassung

Die experimentelle Evaluation des in dieser Arbeit vorgestellten Dienstplatzierungssystems SPi zeigt, dass dieses in Kombination mit dem Dienstplatzierungsalgorithmus GCMI in der Lage ist, in einer Vielzahl von Szenarien bessere Dienstkonfigurationen als andere Ansätze zu finden. Daraus resultiert eine Verbesserung der Güte des platzierten Dienstes, während weniger Datenverkehr im Netz für dessen Erbringung erforderlich ist. Kurz zusammengefasst können wir also festhalten, dass sich mit SPi / GCMI eine Steigerung der Dienstqualität unter gleichzeitiger Verminderung der Netzlast erzielen lässt.

Unsere Ergebnisse zeigen darüber hinaus, dass eine verteilte Diensterbringung mit aktiver Dienstplatzierung – wie sie in SPi implementiert ist – die Leistungsfähigkeit eines mit einer herkömmlichen Client/Server-Architektur erbrachten Dienstes übertrifft. Daraus schließen wir, dass unser Ansatz zur verteilten Diensterbringung die Skalierbarkeit von Diensten in Ad-hoc-Netzen grundsätzlich verbessern kann und somit eine interessante Alternative zu etablierten Architekturen darstellt.

Literatur

[BGJS10] Bastian Blywis, Mesut Günes, Felix Juraschek und Jochen Schiller. Trends, Advances, and Challenges in Testbed-based Wireless Mesh Network Research. *Mobile Networks and Applications*, 15(3):315–329, Juni 2010.

[DMJ03] E. Domínguez, J. Muñoz und J. Jerez. Neural Network Algorithms for the p-Median Problem. In *ESANN '03*, Bruges, Belgium, April 2003.

[HCB02] Wendi B. Heinzelman, Anantha P. Chandrakasan und Hari Balakrishnan. An Application-Specific Protocol Architecture for Wireless Microsensor. *IEEE Transactions on Wireless Networking*, 1(4):660–670, Oktober 2002.

[Her10] Klaus Herrmann. Self-organized Service Placement in Ambient Intelligence Environments. *ACM Transactions on Autonomous and Adaptive Systems*, 5(2):1–39, Mai 2010.

[LBC+01] Jinyang Li, Charles Blake, Douglas De Couto, Hu Imm Lee und Robert Morris. Capacity of Ad Hoc Wireless Networks. In *MobiCom '01*, August 2001.

[LSO+07] Nikolaos Laoutaris, Georgios Smaragdakis, Konstantinos Oikonomou, Ioannis Stavrakakis und Azer Bestavros. Distributed Placement of Service Facilities in Large-Scale Networks. In *INFOCOM '07*, Anchorage, AK, USA, Mai 2007.

[MF90] Pitu Mirchandani und Richard Francis, Hrsg. *Discrete Location Theory*. Wiley-Interscience, Dezember 1990.

[Wit10] Georg Wittenburg. *Service Placement in Ad Hoc Networks*. Dissertation, Fachbereich Mathematik und Informatik, Freie Universität Berlin, Oktober 2010.

[WS08] Georg Wittenburg und Jochen Schiller. A Survey of Current Directions in Service Placement in Mobile Ad-hoc Networks. In *PerCom '08*, Hong Kong, März 2008.

[WS10] Georg Wittenburg und Jochen Schiller. Service Placement in Ad Hoc Networks. *PIK - Praxis der Informationsverarbeitung und Kommunikation*, 33(1):21–25, Januar 2010.

[YV01] Haifeng Yu und Amin Vahdat. The Costs and Limits of Availability for Replicated Services. In *SOSP '01*, Chateau Lake Louise, AB, Canada, Oktober 2001.

Georg Wittenburg ist der erste Absolvent des im Jahr 2003 eingeführten Bachelor/Master-Studiengangs in Informatik an der Freien Universität Berlin. Im Anschluss an sein Studium war er von 2006 bis 2010 als wissenschaftlicher Mitarbeiter bei der Arbeitsgruppe Computer Systems & Telematics am dortigen Institut für Informatik beschäftigt. Neben seiner Lehrtätigkeit mit den Schwerpunkten Betriebssysteme und Computernetze organisierte er mehrere wissenschaftliche Workshops und absolvierte Forschungsaufenthalte an der University of Waikato in Hamilton, Neuseeland, und bei Microsoft Research Cambridge. Sein Forschungsinteresse gilt der Entwicklung von effizienten Ad-hoc- und Sensornetzen; er hält zwei Patente in diesem Feld. Derzeit arbeitet Georg Wittenburg als Post-Doktorand in der gemeinsam vom Institut National de Recherche en Informatique et en Automatique (INRIA) und dem École Polytechnique betriebenen Projektgruppe High Performance Communications (HIPERCOM) in Paris.