Henry Ristau

Inhaltlich entkoppelte Kommunikation in heterogenen, veränderlichen Netzwerktopologien

urn:nbn:de:gbv:28-diss2014-0079-9

Diese Arbeit wurde am 9. April 2013 als Dissertation zur Erlangung des akademischen Grades Doktor-Ingenieur (Dr.-Ing.) der Fakultät für Informatik und Elektrotechnik der Universität Rostock eingereicht.

Die Dissertation wurde von der Fakultät für Informatik und Elektrotechnik angenommen und am 27. November 2013 erfolgreich verteidigt.

Gutachter

Prof. Dr. rer. nat. Clemens H. Cap,
Universität Rostock

Prof. Dr.-Ing. Gero Mühl,
Universität Rostock

Prof. Dr. rer. nat. Hans-Ulrich Heiß,
Technische Universität Berlin

Bibliografische Informationen

Die deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über `http://dnb.ddb.de` abrufbar.

ISBN 978-3-86553-427-9

für
Ellinor
und
Alexandra

Inhaltsverzeichnis

1 Einleitung **1**
- 1.1 Vision . . . 3
- 1.2 Spontane Intelligente Umgebungen . . . 5
- 1.3 Ein Bahnhof als praktisches Beispiel . . . 6
- 1.4 Probleme bestehender Ansätze . . . 7
- 1.5 Beitrag der Arbeit . . . 8
- 1.6 Aufbau der Arbeit . . . 9

2 Grundlagen **11**
- 2.1 Publish/Subscribe . . . 12
  - 2.1.1 Ausdruck von Interesse . . . 13
  - 2.1.2 Verbreitung von Nachrichten . . . 15
- 2.2 Grundlegende Kommunikationsalgorithmen . . . 17
  - 2.2.1 Flooding . . . 18
  - 2.2.2 Gossiping . . . 23
- 2.3 Zusammenfassung . . . 25

3 Entkopplung - anwendungsbezogene Taxonomie für Kommunikationsverfahren **27**
- 3.1 Verwandte Arbeiten . . . 28
  - 3.1.1 Entkopplung in Raum, Zeit und Synchronisation . . . 29
  - 3.1.2 Formalisierung von Kopplungseigenschaften . . . 30
- 3.2 Dimensionen der Entkopplung . . . 31
  - 3.2.1 Räumliche Entkopplung . . . 31
  - 3.2.2 Zeitliche Entkopplung . . . 32
  - 3.2.3 Entkopplung des Produzenten . . . 34
  - 3.2.4 Entkopplung des Konsumenten . . . 35
  - 3.2.5 Inhaltliche Entkopplung . . . 37
- 3.3 Klassifizierung bestehender Kommunikationsparadigmen . . . 39
  - 3.3.1 Paketbasierte Kommunikation . . . 40
  - 3.3.2 Verbindungsorientierte Kommunikation . . . 40
  - 3.3.3 Abstraktion durch Funktionen/Methoden . . . 41

3.3.4 Nachrichtenbasierte Kommunikation 42
3.4 Anforderungen entsprechend der Problemstellung 45
3.5 Zusammenfassung 46

4 Inhaltsentkopplung durch Routing in einer Vermittlungstopologie 49
4.1 Verwandte Arbeiten 51
4.1.1 Publish/Subscribe 52
4.1.2 Sensornetze 58
4.1.3 Pervasive Computing 59
4.1.4 Zusammenfassung 63
4.2 Definition der Vermittlungstopologie 63
4.2.1 Komplexität 67
4.2.2 Arten der Datenverarbeitung 69
4.3 Vorstellung einer geeigneten Systemarchitektur 71
4.3.1 Zusammenspiel zwischen den Brokern 72
4.3.2 Aufbau des Brokers 73
4.4 Routing in einer Vermittlungstopologie 77
4.4.1 Auf Flooding basierende Ansätze 77
4.4.2 Routing ohne Ankündigungen 78
4.4.3 Routing mit Ankündigungen 78
4.5 Bewertung gefundener Pfade 79
4.5.1 Netzwerkabstraktionsschicht 80
4.5.2 Anwendungsabstraktionsschicht 81
4.5.3 Vermittlungsschicht 82
4.6 Anforderungen an die Anwendungen 83
4.6.1 Anwendungsklassen 83
4.6.2 Besonderheiten der Schnittstellen 86
4.6.3 Inhaltliche Anforderungen 87
4.7 Zusammenfassung 89

5 Algorithmen zum Routing in einer Vermittlungstopologie 91
5.1 Grundsätzliche Entscheidungen 92
5.1.1 Überblick 93
5.1.2 Notation 93
5.1.3 Verbreitung der Ankündigungen 94
5.1.4 Gültigkeitsdauer von Ankündigungen 97
5.1.5 Identifikation der Senken 99
5.1.6 Zwischenspeichern von Veröffentlichungen 99
5.2 Pfadauswahl in Phase 2 101
5.2.1 Umgang mit Ankündigungen 101

5.2.2 Abonnements, Nachrichten und Sonderfälle 102
5.2.3 Diskussion 103
5.3 Pfadauswahl in Phase 3 103
5.3.1 Umgang mit Ankündigungen 104
5.3.2 Markieren möglicher Pfade 105
5.3.3 Vorgehen bei Pfadverlust 108
5.3.4 Diskussion 110
5.4 Diskussion 111
5.4.1 Komplexitätsabschätzung der entwickelten Algorithmen 111
5.4.2 Skalierungsfähigkeit der Algorithmen 115
5.4.3 Parameter und deren Einfluss 118
5.4.4 Zuordnung zu Anwendungsklassen 120
5.5 Zusammenfassung 121

6 Simulative Evaluation **123**
6.1 Ausgangssituation 124
6.1.1 Zu überprüfende Aussagen 124
6.1.2 Bewertungskriterien 125
6.2 Methodik 129
6.2.1 Simulationsumgebung 129
6.2.2 Modellierung 130
6.2.3 Szenarien 131
6.2.4 Validierung der Simulationsumgebung 133
6.3 Untersuchung des Einflusses der Parameter 135
6.3.1 Versuch 1 – Gültigkeitsdauer der Ankündigungen 135
6.3.2 Versuch 2 – Anzahl zwischengespeicherter Nachrichten 135
6.3.3 Versuch 3 – Anzahl in Abonnements referenzierter Nachrichten . . 136
6.3.4 Schlussfolgerungen 136
6.4 Untersuchung der Algorithmen 137
6.4.1 Versuch 4 – Auftreten von Pfadfehlern 137
6.4.2 Versuch 5 – Einfluss von Mobilität 138
6.4.3 Versuch 6 – Vergleich mit Flooding 138
6.4.4 Versuch 7 – Verhalten bei Anwendungen der Klasse 2 139
6.4.5 Versuch 8 – Verhalten bei Anwendungen der Klasse 3 139
6.4.6 Versuch 9 – Verhalten bei Anwendungen der Klasse 4 140
6.4.7 Versuch 10 – Horizontale Skalierbarkeit 140
6.4.8 Versuch 11 – Vertikale Skalierbarkeit 141
6.4.9 Schlussfolgerungen 142
6.5 Versuch 12 – Untersuchung in einem realitätsnahen Szenario 145
6.5.1 Herausforderungen 145

6.5.2 Ergebnisse 147
6.6 Schwächen der Ansätze 151
6.6.1 Ein-Wege-Ansatz 151
6.6.2 Mehr-Wege-Ansatz 152
6.7 Zusammenfassung 154

7 Zusammenfassung und Ausblick 157
7.1 Ergebnisse der Arbeit 158
7.1.1 Taxonomie zur Untersuchung von Kommunikationsparadigmen . . 158
7.1.2 Konzept der Vermittlungstopologie 159
7.1.3 Entwicklung zweier geeigneter Algorithmen 162
7.1.4 Untersuchung und Vergleich der Algorithmen 164
7.2 Offene Forschungsfragen 166
7.2.1 Weitere Optimierung der vorgestellten Ansätze 166
7.2.2 Sicherheit 167
7.2.3 Persistenz 167
7.2.4 Integration bestehender Anwendungen 168
7.3 Schlusswort 168

Symbolverzeichnis 171

Literaturverzeichnis 177

A Komplexität des Flooding-Algorithmus A-1

B Versuchsprotokolle A-3
B.1 Validierung der Simulationsumgebung A-3
B.2 Versuch 1 – Gültigkeitsdauer der Ankündigungen A-7
B.3 Versuch 2 – Anzahl zwischengespeicherter Nachrichten A-13
B.4 Versuch 3 – Anzahl in Abonnements referenzierter Nachrichten A-17
B.5 Versuch 4 – Auftreten von Pfadfehlern A-22
B.6 Versuch 5 – Einfluss von Mobilität A-25
B.7 Versuch 6 – Vergleich mit Flooding A-30
B.8 Versuch 7 – Verhalten bei Anwendungen der Klasse 2 . . . . . A-34
B.9 Versuch 8 – Verhalten bei Anwendungen der Klasse 3 . . . . . A-38
B.10 Versuch 9 – Verhalten bei Anwendungen der Klasse 4 . . . . . A-42
B.11 Versuch 10 – Horizontale Skalierbarkeit A-46
B.12 Versuch 11 – Vertikale Skalierbarkeit A-52
B.13 Versuch 12 – Untersuchung in einem realitätsnahen Szenario A-58

Abbildungsverzeichnis

2.1 Kommunikation in Publish/Subscribe Systemen 13
2.2 Implosion beim Flooding 18
2.3 Ein Pfad wird nicht gefunden 19
2.4 Komplexität Erweitertes Flooding 23

3.1 Zeitliche Entkopplung 33
3.2 Entkopplung des Produzenten 35
3.3 Entkopplung des Produzenten 2 36
3.4 Entkopplung des Konsumenten 36
3.5 Entkopplung des Inhalts 38

4.1 Information Flow Graph 57
4.2 Netzwerk- und Verarbeitungstopologie 64
4.3 Zustandsbehaftete Nachrichtenverarbeitung in der Verarbeitungstopologie . 66
4.4 Vermittlungstopologie 67
4.5 Vermittlungstopologie 72
4.6 Aufbau Broker 74
4.7 Pfadmetrik in der Vermittlungstopologie 82
4.8 Anwendungstaxonomie 85

5.1 Ecke mit Nachbarn 94
5.2 Verteilungsgraphen der Ankündigungen 96
5.3 Gültigkeitsdauerverlängerung einer Ankündigung 98
5.4 Vermeidung von Mehrfachübertragungen durch bzw. bei Zwischenspeicherung 100
5.5 Broker in Phase 1 105
5.6 Broker in Phase 2 106
5.7 Zustandsbehaftete Datenverarbeitung 107
5.8 Ecke bei Pfadverlust 108
5.9 Gesamtsituation bei Pfadverlust 109

6.1 Modellierung 131
6.2 Bahnhofszenario 146

6.3 Versuch 12 – Vollständigkeit 148
6.4 Versuch 12 – Netzlast 149
6.5 Versuch 1 – Histogramm der Vollständigkeit 152
6.6 Versuch 12 – Doppeltübertragungen 153
6.7 Doppeltuebertragung 153

B.1 Validierung – Flooding A-4
B.2 Validierung – Ein-Wege-Ansatz A-5
B.3 Validierung – Mehr-Wege-Ansatz A-6
B.4 Versuch 1 – Vollständigkeit A-8
B.5 Versuch 1 – Vollständigkeit als Histogramm A-9
B.6 Versuch 1 – Phase 1 Overhead A-10
B.7 Versuch 1 – Phase 2 Overhead A-11
B.8 Versuch 1 – Pfadnutzungsdauer A-12
B.9 Versuch 2 – Vollständigkeit A-14
B.10 Versuch 2 – Netzlast A-15
B.11 Versuch 2 – Anzahl Doppeltübertragungen A-16
B.12 Versuch 3 – Vollständigkeit A-18
B.13 Versuch 3 – Netzlast A-19
B.14 Versuch 3 – Phase 2 Overhead A-20
B.15 Versuch 3 – Anzahl Doppeltübertragungen A-21
B.16 Versuch 4 – Vollständigkeit A-23
B.17 Versuch 4 – Pfadnutzungsdauer A-24
B.18 Versuch 5 – Vollständigkeit A-26
B.19 Versuch 5 – Netzlast A-27
B.20 Versuch 5 – Phase 1 Overhead A-28
B.21 Versuch 5 – Phase 2 Overhead A-29
B.22 Versuch 6 – Vollständigkeit A-31
B.23 Versuch 6 – Netzlast A-32
B.24 Versuch 6 – Netto-Latenz A-33
B.25 Versuch 7 – Vollständigkeit A-35
B.26 Versuch 7 – Netzlast A-36
B.27 Versuch 7 – Netto-Latenz A-37
B.28 Versuch 8 – Vollständigkeit A-39
B.29 Versuch 8 – Netzlast A-40
B.30 Versuch 8 – Anknüpfungslatenz A-41
B.31 Versuch 9 – Vollständigkeit A-43
B.32 Versuch 9 – Netzlast A-44
B.33 Versuch 9 – Netto-Latenz A-45
B.34 Versuch 10 – Vollständigkeit A-47

B.35 Versuch 10 – Netzlast A-48
B.36 Versuch 10 – Phase 1 Overhead A-49
B.37 Versuch 10 – Phase 2 Overhead A-50
B.38 Versuch 10 – Netto-Latenz A-51
B.39 Versuch 11 – Vollständigkeit A-53
B.40 Versuch 11 – Netzlast A-54
B.41 Versuch 11 – Phase 1 Overhead A-55
B.42 Versuch 11 – Phase 2 Overhead A-56
B.43 Versuch 11 – Netto-Latenz A-57
B.44 Versuch 12 – Vollständigkeit Bilder an stationäre Senken A-60
B.45 Versuch 12 – Vollständigkeit Bilder an mobile Senken A-61
B.46 Versuch 12 – Netzlast Bilder A-62
B.47 Versuch 12 – Phase 1 Overhead Bilder A-63
B.48 Versuch 12 – Phase 2 Overhead Bilder A-64
B.49 Versuch 12 – Netzlast Fahrplan A-65
B.50 Versuch 12 – Doppeltübertragungen A-66
B.51 Versuch 12 – Doppeltübertragungen bereinigt A-67
B.52 Versuch 12 – Netto-Latenz Bilder an stationäre Senken A-68
B.53 Versuch 12 – Netto-Latenz Bilder an mobile Senken A-69
B.54 Versuch 12 – Netto-Latenz Fahrplan an stationäre Senken A-70
B.55 Versuch 12 – Netto-Latenz Fahrplan an mobile Senken A-71
B.56 Versuch 12 – Anknüpfungslatenz Fahrplan an mobile Senken A-72

Tabellenverzeichnis

3.1 Entkopplung nach Eugster et. al. 29
3.2 Räumliche Entkopplung 32
3.3 Beispiele Inhaltlicher Entkopplung 38
3.4 Entkopplungseigenschaften von Kommunikationsparadigmen 39
3.5 Entkopplungsanforderungen spontaner intelligenter Umgebungen 46

4.1 Komplexität der Vermittlungstopologie 69
4.2 Übersicht Routingansätze in Vermittlungstopologie 79

5.1 Aufbau einer Ankündigung 95
5.2 Übersicht des Kommunikationsaufwandes 112
5.3 Übersicht des Speicherbedarfs 113
5.4 Skalierbarkeit in Bezug auf den Kommunikationsaufwand 116

6.1 Übersicht über die Versuche 126
6.2 Protokolloverhead der Algorithmen 128
6.3 Parameter des Testszenarios 133
6.4 Parameter des Versuches zur Validierung 134

B.1 Vollständige Konfiguration des Versuches zur Validierung A-3
B.2 Vollständige Konfiguration von Versuch 1 . . . . . A-7
B.3 Vollständige Konfiguration von Versuch 2 . . . . . A-13
B.4 Vollständige Konfiguration von Versuch 3 . . . . . A-17
B.5 Vollständige Konfiguration von Versuch 4 . . . . . A-22
B.6 Vollständige Konfiguration von Versuch 5 . . . . . A-25
B.7 Vollständige Konfiguration von Versuch 6 . . . . . A-30
B.8 Vollständige Konfiguration von Versuch 7 . . . . . A-34
B.9 Vollständige Konfiguration von Versuch 8 . . . . . A-38
B.10 Vollständige Konfiguration von Versuch 9 . . . . . A-42
B.11 Vollständige Konfiguration von Versuch 10 . . . . . A-46
B.12 Vollständige Konfiguration von Versuch 11 . . . . . A-52
B.13 Vollständige Konfiguration des Bahnhofszenarios A-59

Danksagung

Herzlich bedanken möchte ich mich bei Herrn Professor Cap, der mich bei dieser Arbeit betreut und unterstützt hat. In vielen Gesprächen und Emails hat er nicht mit Ideen und Kritik gespart. Auch meinem zweiten Betreuer, Herrn Professor Mühl, danke ich für die gute Beratung und die zielführende Kritik. Herrn Professor Heiß danke ich für das Gutachten und dass er sich die Zeit genommen hat, für die Verteidigung der Arbeit nach Rostock zu reisen.

Die Arbeit ist im Rahmen des Graduiertenkollegs 1424, MuSAMA, entstanden. Für die Unterstützung in Form eines Stipendiums möchte ich mich bei der Deutschen Forschungsgemeinschaft bedanken. Außerdem gilt mein Dank allen, die das Graduiertenkolleg 1424 ermöglicht haben, insbesondere Herrn Professor Kirste, dem Sprecher desselben.

Mein Dank gilt auch meinen Kollegen im Graduiertenkolleg, allen voran Maik und Christian mit denen ich ein Büro und eine Menge Spaß und Sorgen teilte. Christian, dir nochmal herzlichen Dank für das Timing deiner Verteidigung. Damit hast du mir den letzten nötigen Motivationsschub gegeben.

Den Kollegen am Lehrstuhl IuK danke ich für die vielen Diskussionen und Gespräche, das gelegentliche kritische Feedback und die zahlreichen Tipps und Ratschläge. Danke Thomas, Martin, Petra, Till, David und Kerstin.

Ein großes Dankeschön gilt meiner Frau und meiner Familie, die stets an mich geglaubt und den nötigen Druck in die richtige Richtung aufrecht erhalten haben.

Allen Wegbegleitern und Unterstützern, die ich hier nicht aufzählen konnte oder gar vergessen habe, vielen Dank!

Kapitel 1.

Einleitung

Inhalt

1.1	Vision	**3**
1.2	Spontane Intelligente Umgebungen	**5**
1.3	Ein Bahnhof als praktisches Beispiel	**6**
1.4	Probleme bestehender Ansätze	**7**
1.5	Beitrag der Arbeit	**8**
1.6	Aufbau der Arbeit	**9**

Mit der ständigen Weiterentwicklung bestehender und der Entwicklung neuer, vor allem mobiler Kommunikationstechnologien, durchdringen diese Technologien mehr und mehr unseren Alltag. Dadurch und durch die häufige organisatorische Trennung verschiedener Domänen innerhalb einzelner Kommunikationstechnologien entstehen immer komplexere und vielseitigere Netzwerktopologien zwischen verschiedenen physikalisch nicht weit voneinander entfernten Geräte. Diese Netzwerktopologien sind durch die Mobilität einzelner Geräte häufig einer ständigen Veränderung unterworfenen. Der Austausch von Informationen zwischen Geräten in einer solchen heterogenen, veränderlichen Umgebung stellt bereits eine enorme Herausforderung dar. Eine weitere Herausforderung ist die Interoperabilität zwischen einzelnen Anwendungen auf diesen Geräten. Es existiert eine ständig wachsende Vielfalt von Beschreibungssprachen und Formaten für Dokumente und andere Informationen. Die Bereitstellung von Informationen in Form einer oder mehrerer Nachrichten von einem Gerät in einer Form, wie sie die Anwendung auf diesem Gerät erzeugt, zur Benutzung auf einem oder mehreren anderen Geräten in einer Form, wie sie die betroffenen Anwendungen auf diesen Geräten verarbeiten können, in einer solchen heterogenen, veränderlichen Umgebung ist die Forschungsfrage dieser Dissertation.

Zur Illustration sei, als ein sehr einfaches Beispiel, ein typisches Drei-Personen-Büro wie das des Autors beschrieben. Es gibt eine Ethernet-Verbindung zur Anbindung der Drucker und Laptops an das Netzwerk des Instituts. Außerdem gibt es eine Ethernet-Verbindung zur Anbindung der Telefone an das Netzwerk des Rechenzentrums. Es gibt verschiedene WLAN-Accesspoints, die das WLAN des Rechenzentrums für mobile Geräte verfügbar machen. Die verwendeten Technologien, IP über Ethernet und WLAN sind prinzipiell miteinander kompatibel. Die drei beschriebenen Netzwerke sind allerdings organisatorisch und durch verschiedene Firewalls voneinander getrennt. Weiterhin benutzen meine Kollegen und ich Laptops, die zum Teil Bluetooth unterstützen. Außerdem besitzen wir Mobiltelefone, die wenigstens Bluetooth und die aktuellen mobilen Kommunikationsstandards, wie GPRS, EDGE, UMTS, etc. unterstützen. Einige der Mobiltelefone unterstützen auch WLAN, können aber nicht ohne manuellen Aufwand mit dem WLAN des Rechenzentrums kommunizieren, weil dafür eine umständliche Authentifizierung per HTTPS notwendig ist. Eine Kommunikation in einem ad-hoc WLAN mit anderen Geräten ist aber durchaus möglich. Viele der Geräte befinden sich nicht immer im Büro, sondern werden gelegentlich oder täglich daraus entfernt oder ständig umhergetragen. Die beschriebene Kommunikationslandschaft ist heterogen und die Topologie verändert sich ständig. Im Prinzip könnte jedes Gerät in dieser Kommunikationslandschaft Informationen auf wenigstens einem Pfad gegebenenfalls über dritte Geräte an jedes andere Gerät übermitteln. Trotzdem würden wir zum Beispiel ein Bild auf einem der Mobiltelefone von Hand per Bluetooth auf einen der Laptops übertragen, dort in einer Anwendung skalieren und vom Druckertreiber in PostScript transformieren lassen, um es dann auf dem Drucker auszudrucken. Die Telefonnummer eines verpassten Anrufes würden wir vom Display des Bürotelefons in unser Mobiltelefon abtippen. Obwohl die beschriebene Umgebung sehr überschaubar ist, machen wir von der Möglich-

keit der uns umgebenden Geräte, miteinander kommunizieren zu können, so gut wie keinen Gebrauch.

1.1. Vision

In einer räumlich begrenzten Umgebung sind viele verschiedenartige Geräte durch verschiedenartige Kommunikationstechnologien derart verbunden, dass eine Nachricht von jedem beliebigen Gerät durch schrittweises Weiterleiten über andere Geräte jedes beliebige andere Gerät erreichen kann. Barrieren zwischen inkompatiblen Kommunikationstechnologien werden dabei von Geräten überbrückt, die in der Lage sind, beide Technologien zu benutzen. Außerdem ist ein Teil der Geräte nicht ständig in dieser Umgebung vorhanden. Sie werden spontan in die Umgebung eingebracht, später wieder entfernt und gegebenenfalls durch andere Geräte ersetzt. Die Geräte in einer solchen Umgebung bilden eine heterogene, veränderliche Netzwerktopologie.

Bei der beschriebenen Umgebung kann es sich zum Beispiel um einem Raum oder ein Gebäude handeln. Die kommunizierenden Geräte können zum Beispiel PCs, Laptops, Mobiltelefone aber auch Sensoren oder eingebettete Systeme sein. Beispiele für die Kommunikationstechnologien sind Ethernet, WLAN, Bluetooth oder auch ZigBee.

Auf den Geräten der heterogenen, veränderlichen Netzwerktopologie sind Anwendungen aktiv, die Informationen in Form von einer oder mehreren Nachrichten veröffentlichen und/oder empfangen wollen. Sie übernehmen also die Rolle des Nachrichtenproduzenten und/oder des Nachrichtenkonsumenten. Außerdem sind auf entsprechend leistungsfähigen Geräten Anwendungen aktiv, die in der Lage sind, Nachrichten zu verarbeiten. Diese Verarbeitung kann sowohl die simple Umwandlung einer Nachricht von einem Nachrichtentyp in einen anderen oder eine komplexe Operation auf mehreren Nachrichten, die über bestimmte Eigenschaften zueinander passen, sein. Ein Beispiel für eine einfache Umwandlung ist das Konvertieren eines Word-Dokuments in ein PDF-Dokument. Ein Beispiel für eine komplexere Operation ist die Detektion von Personen auf zeitlich aufeinander folgenden Bildern einer Überwachungskamera.

Ziel einer entsprechenden Middleware ist es, Nachrichten von Produzenten an daran interessierte Konsumenten zu vermitteln und dabei wenn nötig, Verarbeiter so einzubinden, dass die Konsumenten in der Lage sind, die empfangenen Nachrichten zu verarbeiten. Dabei soll ein Produzent lediglich beschreiben müssen, welchen Typ von Nachrichten er generiert. Ein Konsument soll entweder angeben können, an welchen Nachrichtentypen er interessiert ist, oder dies anhand der Beschreibung verfügbarer Nachrichtentypen entscheiden können. Ein Verarbeiter soll anhand der Beschreibungen von verfügbaren Nachrichtentypen entscheiden können, Nachrichten welchen Typs er daraus generieren kann. Was genau dabei ein Nachrichtentyp ist, soll für die Anwendung so flexibel wie möglich zu gestalten sein. In keinem Fall soll der Produzent oder Konsument beschreiben müssen, welche Verarbeitungsschrit-

te zwischen Ihnen notwendig sind. Dies muss die Middleware selbst entscheiden können. Zwischen Produzenten und Konsumenten findet somit eine inhaltlich entkoppelte, nachrichtenbasierte Kommunikation statt.

Kriterien, die hier eine gute Middleware charakterisieren, sind:

Zuverlässigkeit Unabhängig von der momentanen Netzwerktopologie, deren Grad an Heterogenität sowie Veränderlichkeit, muss eine Middleware robust und zuverlässig Nachrichten von Produzenten an Konsumenten zustellen und verfügbare und geeignete Verarbeiter auswählen.

Transparenz Aufgrund der ständigen Veränderung der Netzwerktopologie sowie der organisatorischen und technologiebedingten Inhomogenität der verfügbaren Adressräume muss die Kommunikationsinfrastruktur so gut wie möglich vor den Anwendungen verborgen bleiben. Eine direkte Adressierung von Kommunikationspartnern über eine gültige physikalische Adresse ist ausgeschlossen.

Interoperabilität Beim Transfer von Nachrichten von Produzenten zu Konsumenten kann es notwendig sein, diese Nachrichten mehreren Verarbeitungsschritten zu unterziehen. Welche Verarbeiter und Verarbeitungsschritte dazu nötig sind, hängt unter anderem von der Umgebung der beteiligten Anwendungen und der Verfügbarkeit entsprechender Verarbeiter ab. Daher ist es sinnvoll, diese Verarbeitung sowohl vor den Produzenten als auch den Konsumenten zu verbergen.

Effizienz Die Verbreitung der Verfügbarkeit von Nachrichten für potentielle Interessenten sowie die Zustellung von Nachrichten an entsprechende Konsumenten darf die Kommunikationstechnologien nicht stärker belasten als nötig. In einer heterogenen Netzwerktopologie müssen leistungsstarke Kommunikationstechnologien vor weniger leistungsfähigen bevorzugt werden. Das gleiche gilt bei der gleichzeitigen Verfügbarkeit unterschiedlich leistungsfähiger Verarbeiter. Außerdem muss der Overhead durch unnötige Datenübertragungen und hohen Verwaltungsaufwand trotz der ständigen Veränderung der Netzwerktopologie möglichst gering gehalten werden.

Skalierbarkeit Die beschriebene Middleware soll in einer Umgebung aus Geräten, wie zum Beispiel einem Raum oder einem Gebäude arbeiten. Die Größe dieser Umgebung, die Anzahl der beteiligten Geräte sowie die Anzahl der verwendeten Kommunikationstechnologien kann variierten. Trotzdem bleibt die Umgebung abgeschlossen. Eine internetweite Skalierbarkeit steht hier nicht im Vordergrund. Von Interesse sind viel mehr die Skalierbarkeit in Bezug auf die Anzahl der kommunizierten Nachrichtentypen, der Anzahl der Konsumenten oder auch der Anzahl gleichartiger, parallel verfügbarer Verarbeiter.

Sicherheit An die Kommunikation zwischen Anwendungen in einer solchen Umgebung können verschiedenste Sicherheitsanforderungen gestellt werden. Inwiefern sich diese von den Sicherheitsanforderungen in anderen Domänen unterscheiden und wie gegebenenfalls auf Unterschiede zu reagieren ist, gehört nicht zum Kontext dieser Arbeit.

Die Definition und Gewährleistung von Sicherheitsanforderungen ist ein eigenes sehr komplexes Gebiet der Informatik.

Einfache Integrierbarkeit/Erweiterbarkeit/Austausch Aufgrund der ständigen Veränderung der Umgebung in Bezug auf die Verfügbarkeit von Geräten und Anwendungen, muss die Integration neuer Anwendungen, die Erweiterung bestehender Anwendungen oder der Austausch von Anwendungen im laufenden Betrieb der Umgebung ohne Einfluss auf die Kommunikation zwischen bestehenden Anwendungen möglich sein.

Wiederverwendbarkeit/Portabilität Wiederverwendbarkeit und Portabilität sind sehr allgemeine Anforderungen, die für jede Middleware von Bedeutung sind. Lediglich der hohe Grad an Heterogenität der Geräte unterstreicht die Wichtigkeit von Portabilität. Beide Eigenschaften sind aber Fragen der Implementierung und nicht konzeptueller Natur und werden in dieser Arbeit daher nur begrenzt betrachtet.

Die vollständige Realisierung einer in dieser Ideenskizze beschriebenen Middleware ist ein ehrgeiziges Ziel, dem diese Arbeit nicht gerecht werden kann. Viel mehr wird sie aufzeigen, dass bereits sehr einfache Grundlagen zur Realisierung dieser Vision momentan nicht verfügbar sind. Dazu werden bestehende Ansätze analysiert, notwendige Erweiterungen aufgezeigt, eine geeignete Systemarchitektur und zwei entsprechende Kommunikationsalgorithmen entwickelt und in einer simulativen Studie ausführlich untersucht und bewertet.

1.2. Spontane Intelligente Umgebungen

Hintergrund der im vorangehenden Abschnitt beschriebenen Vision ist die Idee der spontanen intelligenten Umgebung (HEIDER, 2009, S.15). Sie geht zurück auf Bestrebungen der frühen 90er Jahre (RONZANI, 2009). Dazu gehört die Idee des Ubiquitous Computing, der allgegenwärtigen, vom Benutzer nicht mehr wahrnehmbaren Computer (WEISER, 1993), und später des Pervasive Computing, des ständigen, mobilen Zugriffs des Benutzers auf für ihn relevante Informationen (HANSMANN et al., 2003, S.11). Darauf aufbauend entstanden Ambient Intelligence, die Überlegung, dass im Hintergrund verschwindende Computer völlig neuartige Bedienungskonzepte benötigen (AARTS, HARWIG und SCHUURMANS, 2002), und Smart Environments (COOK und DAS, 2005), also intelligente Umgebungen.

Die Kernthese bei intelligenten Umgebungen ist es, dass ein Benutzer in einer bestimmten Umgebung, wie zum Beispiel einem Büro oder einem Wohnhaus, ständig von vielen Geräten umgeben ist. Diese Geräte können in ihrer Gesamtheit durch gezielte Kooperation die Umgebung des Benutzers derart mit Diensten anreichern, dass sein Alltag in dieser Umgebung dadurch erleichtert wird. Grundvoraussetzung dafür ist, dass die Anwendungen auf den in der intelligenten Umgebung befindlichen Geräten miteinander kommunizieren können.

Die Idee der spontanen intelligenten Umgebung geht noch einen Schritt weiter und geht davon aus, dass eine solche Umgebung überall spontan um den Benutzer herum aus den momentan in dieser Umgebung befindlichen Geräten entstehen kann. Während eine intelli-

gente Umgebung wie ein Wohnhaus oder ein Konferenzraum noch problemlos mit einer fest installierten und konfigurierten Infrastruktur ausgestattet werden kann, ist dies bei spontanen intelligenten Umgebungen nicht mehr ohne weiteres möglich. Eine solche Umgebung kann prinzipiell überall entstehen wo Benutzer das Bedürfnis nach Unterstützung haben. Das Vorhandensein einer homogenen, organisatorisch nicht getrennten, geschickt konfigurierten Kommunikationsinfrastruktur ist hier nicht mehr gegeben.

1.3. Ein Bahnhof als praktisches Beispiel

Als anschauliches Beispiel für die Ideenskizze im Zuge der Etablierung spontaner intelligenter Umgebungen sei ein Szenario in einem Bahnhof beschrieben. Der Bahnhof besteht, angelehnt an den Rostocker Hauptbahnhof, aus drei Bereichen, in denen sich Reisende aufhalten. Auf der Vorderseite des Bahnhofs im Norden befindet sich ein größerer Gebäudekomplex mit dem Reisezentrum und Geschäften. Auf der Rückseite des Bahnhofs im Süden befindet sich ein kleineres Gebäude mit wenigen Geschäften. Dazwischen liegt der Bereich der Bahnsteige, wobei zwischen Gleis 3 und 4 noch ein kleines Verwaltungsgebäude existiert. Die nachfolgend angedeutete und im weiteren Verlauf dieser Arbeit detaillierte und simulierte Infrastruktur und das Szenario selbst bilden einen kleinen Ausschnitt einer möglichen Gesamtinfrastruktur ab und sind frei erfunden.

Die zentrale Kommunikationsinfrastruktur des Bahnhofs besteht aus verschiedenen drahtgebundenen und drahtlosen LANs, die teilweise aus organisatorischen und Sicherheitsgründen durch Firewalls voneinander getrennt sind. Im Bahnhof halten sich Reisende und Mitarbeiter auf, die die Benutzer der intelligenten Umgebung repräsentieren. Die beiden betrachteten Dienste seien die Bereitstellung des aktuellen Fahrplans für Interessierte sowie die Bereitstellung der Bilder der fünf Überwachungskameras auf den Bahnsteigen für die Mitarbeiter des Sicherheitsdienstes.

Die Fahrplaninformationen werden zentral generiert und müssen auf zwei großen Anzeigetafeln dargestellt werden. Außerdem sollen Reisende und Vorbeigehende bei Bedarf Fahrplaninformationen auf ihren mobilen Geräten anschauen können. Zentrale Herausforderungen sind hier

Transparenz/Interoperabilität Die gesamte Infrastruktur des Bahnhofs muss vor dem mobilen Gerät des Benutzers verborgen bleiben. Die Anwendung auf dem mobilen Gerät ist weder daran interessiert, auf welchem Gerät die Informationen generiert, wo sie eventuell verarbeitet wurden oder welche Formate bei der Verarbeitung eine Rolle spielen.

Einfache Integrierbarkeit/Erweiterbarkeit/Austausch Die Infrastruktur des Bahnhofs kann nicht für alle existierenden und zukünftig entwickelten Kommunikationstechnologien mobiler Endgeräte vorbereitet werden. Vielmehr ist davon auszugehen, dass im Bahnhof immer auch Geräte verfügbar sind, die eine Barriere zwischen verfügbaren

und benötigten Kommunikationstechnologien überbrücken können. Sind Informationen seitens der Infrastruktur zum Beispiel per WLAN verfügbar, so reicht rein technisch ein mobiles Gerät, dass WLAN und Bluetooth beherrscht in der Nähe, um ein Gerät über Bluetooth mit diesen Informationen zu versorgen.
Weiterhin werden ständig neue mobile Anwendungen entwickelt, die entsprechende Informationen in anderen Formaten benötigen. Dazu darf es nicht jedes Mal nötig sein, die bestehenden Fahrplaninformationen erzeugenden und benutzenden Anwendungen anzupassen.

Auf jedem der fünf Bahnsteige befindet sich eine Überwachungskamera. Die Bilder werden zentral im Verwaltungsgebäude angezeigt und gespeichert. Trotzdem soll es einem Mitarbeiter des Sicherheitsdienstes möglich sein, mit einem mobilen Gerät auf den Bahnsteigen unterwegs zu sein und die Bilder ausgewählter Kameras dort anzusehen. Zentrale Herausforderungen sind hier

Zuverlässigkeit Unabhängig von Position und Geschwindigkeit des Sicherheitsbeamten müssen die entsprechenden Bilder zuverlässig auf das mobile Gerät übertragen werden. Auch wenn der Empfangsbereich eines verfügbaren Accesspoints verlassen wird, muss die Zustellung der Bilder über andere Geräte in der Nähe sichergestellt werden.

Effizienz Aufgrund von Größe und Frequenz der Veröffentlichung von Überwachungsbildern ist es notwendig, diese möglichst effizient zu übertragen um nicht Teile der Netzwerktopologie zu überlasten.

Skalierbarkeit Die Anzahl der Überwachungskameras als Informationsproduzenten sowie die Anzahl der Sicherheitsbeamten mit mobilen Geräten darf keinen Einfluss auf die Funktionsfähigkeit des Gesamtsystems, also die erfolgreiche Übermittlung der Bilder haben.

1.4. Probleme bestehender Ansätze

Im Kontext bestehender Forschung lässt sich die Arbeit in den Bereich des Kommunikationsparadigmas Publish/Subscribe einordnen. Gründe hierfür sind einerseits die angestrebte Verbreitung von Nachrichten und Nachrichtenströmen von Produzenten zu Konsumenten, als auch die angestrebten Formen der Entkopplung zwischen Produzenten, Konsumenten, der Middleware und der Umgebung. Eine detaillierte Einordnung und Begründung für diese Einordnung gibt Kapitel 3 im späteren Verlauf dieser Arbeit.

Als Publish/Subscribe-System betrachtet muss eine Problemlösung zwei wesentliche Aspekte verbinden. Dies sind die inhaltliche Entkopplung von Produzenten und Konsumenten sowie die Anwendbarkeit in heterogenen, veränderlichen Netzwerktopologien. Bestehende Ansätze können sehr gut mit jeweils einer dieser Anforderungen umgehen. Zum Beispiel ist mit Gryphon (BANAVAR et al., 1999) ein Publish/Subscribe-System mit vollständiger inhaltlicher Entkopplung zwischen Produzenten und Konsumenten verfügbar. Dieses

zielt aber auf den Einsatz in fest konfigurierten, statischen Weitverkehrsnetzen und ist konzeptuell nicht auf den Einsatz in heterogenen, dynamischen Umgebungen übertragbar.

Mit MundoCore (AITENBICHLER, KANGASHARJU und MÜHLHAUSER, 2007) existiert eine leistungsfähige Middleware zur Realisierung von Publish/Subscribe in heterogenen, veränderlichen Netzwerktopologien mit dem Fokus auf Pervasive Computing und dem Einsatz in intelligenten Umgebungen, allerdings ist eine inhaltliche Entkopplung zwischen Produzenten und Konsumenten dort nicht vorgesehen. Eine Abbildung von Verarbeitern auf einen Konsumenten für die Ausgangsnachrichten und einen Produzenten für die daraus generierten Nachrichten ist in einem solchen System zwar theoretisch denkbar, um inhaltliche Entkopplung vorzutäuschen, allerdings ist dieses Vorgehen aus verschiedenen, in Kapitel 4 geschilderten Gründen nicht effizient.

1.5. Beitrag der Arbeit

Vier Schwerpunkte bilden den Beitrag dieser Arbeit. Diese sind die Untersuchung bestehender Kommunikationsparadigmen auf ihre Eignung zur Realisierung der Vision, die Erarbeitung eines Konzeptes zur Abbildung der Problemstellung auf ein Routingproblem in einer verteilten Vermittlungstopologie, die Entwicklung und Diskussion zweier Routingalgorithmen zur Lösung dieses Routingproblems, sowie die quantitative Analyse und der Vergleich der Routingalgorithmen durch eine simulative Evaluation.

Untersuchung von Kommunikationsparadigmen Die Fragestellung, mit welchem bestehenden Kommunikationsparadigma sich die eingangs skizzierte Middleware am besten realisieren lasst, führt zur Entwicklung einer Taxonomie, die die Einordnung existierender Kommunikationsparadigmen anhand ihrer Fähigkeit zur Entkopplung zulässt. Die vorgestellte Taxonomie besteht aus fünf Dimensionen der Kopplung bzw. Entkopplung, räumlich, zeitlich, betreffend der Anbindung der Quelle sowie der Senke an die Middleware und inhaltlich. Für jede dieser Dimensionen existieren mehrere diskrete Ausprägungen, die jeweils einen Grad der Kopplung beschreiben. Kommunikationsparadigmen können anhand dieser Taxonomie dahingehend analysiert werden, welche dieser Kopplungsgrade sie ermöglichen und welche nicht. Eine solche Analyse wird auch für die eingangs skizzierte Middleware durchgeführt und selbige anhand dieser Analyse einem geeigneten Kommunikationsparadigma zugeordnet.

Konzept der Vermittlungstopologie Zur Realisierung der Vision müssen Produzenten und Konsumenten von Nachrichten in zwei Dimensionen zueinander geführt werden, räumlich im Netzwerk und inhaltlich im Typ der übertragenen Nachrichten. In dieser Arbeit wird das Konzept einer Vermittlungstopologie vorgestellt, die einen Suchraum über diese beiden Dimensionen aufspannt und es dadurch erlaubt, mit Hilfe eines speziellen verteilten Routingalgorithmus Nachrichten inhaltlich entkoppelt vom Produzenten zum Konsumenten zu übertragen. Es wird eine Systemarchitektur vorgestellt,

die diese Vermittlungstopologie adäquat auf ein heterogenes Netzwerk von Geräten abbilden kann. Außerdem werden Anforderungen an eine Metrik zur Bewertung der Pfade in einer Vermittlungstopologie und eine beispielhafte Metrik vorgestellt.

Routingalgorithmen Zwei verteilte, inhaltsbasierte Routingalgorithmen werden vorgestellt, die in der Lage sind, die Verfügbarkeit von Nachrichtentypen in der Vermittlungstopologie zu verbreiten und geeignete Pfade zu finden, um Nachrichten von Produzenten zu interessierten Konsumenten effizient zu übertragen. Diese unterscheiden sich dahingehend, dass ein Algorithmus jeweils genau einen geeigneten Pfad auswählt, freischaltet und im Falle einer Unterbrechung einen neuen Pfad etabliert, und der zweite Algorithmus alle geeigneten Pfade ermittelt und erst beim Versenden von Nachrichten den besten verfügbaren Pfad für den Versand auswählt. Die beiden Algorithmen sowie deren Verhalten anhand bestimmter Parametereinstellungen werden ausführlich diskutiert.

Simulative Evaluation Die Ergebnisse der Diskussion werden durch Simulation in einem geeigneten Szenario überprüft und Aussagen über das zu erwartende Verhalten der Algorithmen abgeleitet. In einer Simulation des eingangs geschilderten Bahnhofszenarios wird die Funktionalität der Algorithmen in einem komplexen, realitätsnahen Szenario überprüft und anhand der Vollständigkeit der Übertragung, der dabei generierten Netzwerklast sowie der Übertragungsverzögerung der einzelnen Nachrichten miteinander verglichen.

1.6. Aufbau der Arbeit

Im folgenden Kapitel werden wesentliche Grundlagen des Publish/Subscribe-Paradigmas vorgestellt. Außerdem werden für die weitere Arbeit wichtige Kommunikationsalgorithmen, Flooding und Gossiping, sowie mögliche Erweiterungen präsentiert und auf ihre Eignung in heterogenen, dynamischen Umgebungen hin untersucht.

Kapitel 3 stellt die Taxonomie für Kommunikationsparadigmen vor. Dazu werden eingangs verwandte Arbeiten diskutiert. Anschließend werden die Dimensionen der Entkopplung erklärt. Darauf folgend werden existierende Kommunikationsparadigmen entsprechend der Taxonomie eingeordnet und darauf basierend ihre Eignung zur Realisierung der skizzierten Middleware beurteilt.

Den Kern der Arbeit bilden Kapitel 4 und 5. In ersterem wird die Vermittlungstopologie definiert. Dazu werden eingangs verwandte Arbeiten untersucht und im Anschluss die Vermittlungstopologie selbst erklärt. Darauf folgend werden eine geeignete Systemarchitektur vorgestellt, die in der Lage ist, die Vermittlungstopologie auf ein Netzwerk aus Brokern abzubilden. Im Anschluss werden verschiedene grundlegende Konzepte zum Routing von Nachrichten auf ihre Eignung untersucht, eine geeignete Metrik zu Bewertung gefundener Pfade vorgestellt, sowie Anforderungen an die Anwendungen diskutiert.

In Kapitel 5 werden zwei geeignete Routingalgorithmen zum Routing in der Vermittlungstopologie vorgestellt. Dabei werden eingangs grundsätzliche Designentscheidungen und im Anschluss die jeweils für die einzelnen Algorithmen speziellen Erweiterungen erklärt. Daran schließt sich eine ausführliche Diskussion der Komplexitätseigenschaften, Skalierungsfähigkeit und der erwartete Einfluss der einzelnen Parameter an.

Die Evaluation der beiden Algorithmen in verschiedenen Szenarien bildet den Inhalt von Kapitel 6. Nach der Vorstellung der zu prüfenden Aussagen werden die quantitativen Bewertungskriterien vorgestellt, gefolgt von einer kurzen Vorstellung der Modellierung und der Simulationsumgebung sowie der beiden Szenarien. Darauf folgend werden die Ergebnisse der einzelnen Simulationen vorgestellt und deren Bedeutung diskutiert.

Den Abschluss der Arbeit bildet Kapitel 7 mit der Zusammenfassung und Bewertung der Ergebnisse der Arbeit sowie einem Ausblick auf offene Forschungsfragen.

Kapitel 2.

Grundlagen

Inhalt

2.1 Publish/Subscribe **12**
2.1.1 Ausdruck von Interesse 13
2.1.2 Verbreitung von Nachrichten 15
2.2 Grundlegende Kommunikationsalgorithmen **17**
2.2.1 Flooding 18
2.2.2 Gossiping 23
2.3 Zusammenfassung **25**

In diesem Kapitel werden die wesentlichen Grundlagen der Arbeit erklärt und analysiert.

Der erste Abschnitt befasst sich mit dem Publish/Subscribe-Paradigma, dass im weiteren Verlauf der Arbeit eine wichtige Rolle spielen wird. Die verschiedenen Mechanismen zur Filterung und Auswahl von Nachrichten werden vorgestellt und bewertet. Außerdem werden die grundsätzlich möglichen Vorgehensweisen zur Verteilung von Nachrichten in Publish/Subscribe-Systemen erklärt und bewertet. Weitere Kommunikationsparadigmen werden später bei ihrer Klassifizierung im Abschnitt 3.3 nur kurz vorgestellt.

Im Anschluss werden grundlegende Kommunikationsalgorithmen, die zur Verbreitung von Informationen in heterogenen, veränderlichen Netzwerktopologien geeignet sind, vorgestellt, bewertet und deren Anwendbarkeit und Nutzen insbesondere im Rahmen dieser Arbeit kritisch hinterfragt.

2.1. Publish/Subscribe

Publish/Subscribe ist ein Kommunikationsparadigma zur Realisierung ereignisgesteuerter Systeme. Es beinhaltet drei Parteien, den **Produzenten**, den **Konsumenten** und den **Vermittler**[1]. Das zentrale Element in einem Publish/Subscribe System ist das Ereignis. Ein Ereignis beschreibt eine Zustandsänderung im Produzenten, die dieser publiziert. Ereignisse werden in Form von Nachrichten beschrieben und diese Nachrichten werden im Publish/Subscribe-System zwischen Produzenten und Konsumenten vermittelt.

Das Format und der Aufbau von Nachrichten sind durch die Implementierung vorgegeben oder werden vollständig den Anwendungen überlassen. Häufig verwandte Nachrichtenformate sind Name-Wert-Paare, Objekte oder semistrukturierte Daten wie zum Beispiel XML (MÜHL, FIEGE und PIETZUCH, 2006, S.11).

Der Ablauf der Kommunikation in Publish/Subscribe-Systemen wird in Abbildung 2.1 beschrieben. Konsumenten registrieren sich beim Vermittler mit ihrem Interesse. Dieser Vorgang wird als Abonnement oder auch Subscription bezeichnet. Ein Produzent registriert ein Ereignis und generiert eine Nachricht. Diese übergibt er an den Vermittler, der Prozess der Veröffentlichung oder Publication. Der Vermittler stellt die Nachricht anschließend an alle bei ihm registrierten Konsumenten, die Interesse an dieser Nachricht haben, zu. Dieser letzte Vorgang nennt sich Benachrichtigung oder Notification. Daraufhin verarbeitet der Konsument die Nachricht.

Aus diesem Ablauf ergeben sich zwei wesentliche Anforderungen für Publish/Subscribe-Systeme respektive deren Vermittler, zum Einen der Ausdruck von Interesse sowie der Abgleich von Interesse und Nachrichten und zum Anderen die Verbreitung von Nachrichten von den Produzenten zu den Konsumenten. Die verschiedenen Ansätze zur Realisierung dieser Anforderungen eignen sich zur Kategorisierung von Publish/Subscribe-Systemen und

[1] Der Vermittler wird in der Literatur als Event Notification Service, Event Service oder Event Broker bezeichnet

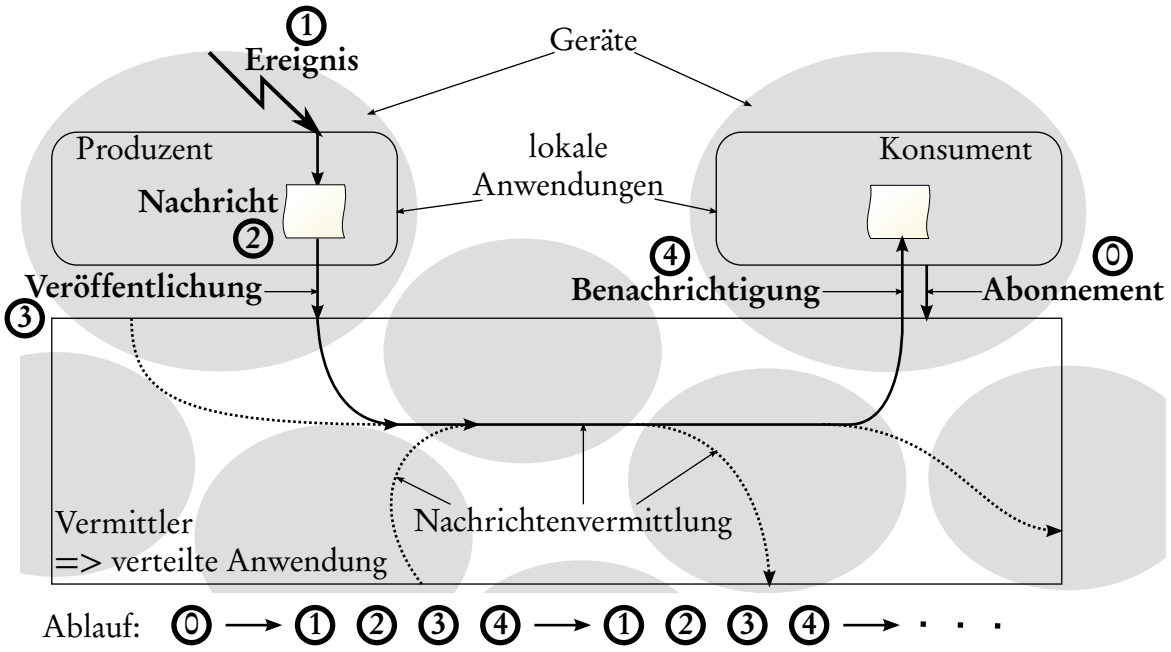


Abbildung 2.1.:
Grundsätzlicher Ablauf der Kommunikation in Publish/Subscribe-Systemen.

werden im Folgenden näher erläutert. Anschließend wird erläutert, welche Probleme mit der Realisierung inhaltlicher Entkopplung mit Hilfe des Publish/Subscribe Paradigmas verbunden sind.

2.1.1. Ausdruck von Interesse

Der Konsument registriert beim Vermittler sein Interesse in Form eines Filters, der beschreibt, an welchen Nachrichten der Konsument interessiert ist. Nur die auf den Filter passenden Nachrichten werden anschließend an diesen Konsumenten vermittelt. Die verwendeten Filtermechanismen sind unterschiedlich ausdrucksstark und haben Einfluss auf die Verteilungsmechanismen im Publish/Subscribe-System.

Kanalbasierte Filterung

Dies ist der ausdrucksschwächste Filtermechanismus. Vom Vermittler werden Kanäle verwaltet. Produzenten können Kanäle erzeugen und ihre Nachrichten in diese Kanäle veröffentlichen. Alle Konsumenten, die einen Kanal abonniert haben, empfangen alle in diesem Kanal veröffentlichten Nachrichten. Ein Publish/Subscribe System mit kanalbasierter Filterung ist zum Beispiel SCRIBE (CASTRO et al., 2002).

Der wesentliche Nachteil kanalbasierter Filterung ist die geringe Ausdrucksstärke. Ein Konsument, der einen Kanal abonniert, empfängt alle darin veröffentlichten Nachrichten.

Die Entscheidung, in welchem Kanal eine Nachricht veröffentlicht wird, liegt aber auf der Seite des Nachrichtenproduzenten. Wenn eine Nachricht in mehrere Kanäle passt und entsprechend in mehreren Kanälen veröffentlicht wird, entspricht das dem Mehrfachversand dieser Nachricht und ist entsprechend ineffizient.

Themenbasierte Filterung

Bei der themenbasierten Filterung versieht der Produzent einer Nachricht diese mit einem Thema, repräsentiert durch eine gegebenenfalls strukturierte Zeichenkette. Zum Beispiel könnte ein Temperaturwert gemessen im Institut für Informatik in Raum 213 mit dem Thema `/UniRo/IEF/AES21/R213/Sensordaten/Temperatur` versehen sein.

Themenbasierte Filterung ist ausdrucksstärker als kanalbasierte Filterung, da der Konsument durch Platzhalter Filter auf den gegebenenfalls strukturierten Themen definieren und somit die Menge der empfangenen Nachrichten anhand ihrer Themen eingrenzen kann. Zum Beispiel können mit `/UniRo/IEF/AES21/R213/Sensordaten/*` alle Sensordaten aus Raum 213 empfangen werden. Ein Beispiel für themenbasierte Filterung wird von OKI et al. (1993) beschrieben.

Die Ausdrucksstärke themenbasierter Filterung ist dahingehend begrenzt, dass verschachtelte Hierarchien nicht adäquat abgebildet werden können. Möchte der Konsument aus dem vorangehenden Beispiel alle Temperaturwerte aus der Fakultät abonnieren, wäre zur Realisierung des Filters `/Sensordaten/Temperatur/UniRo/IEF/*` eine andere Hierarchie nötig. Einerseits skaliert die Struktur der Themen hier schlecht weil auf diese Weise sehr schnell sehr viele Themen nötig werden, andererseits müsste ein Produzent eine Nachricht viele Male veröffentlichen, um sie jedem passenden Thema zuzuordnen.

Typbasierte Filterung

Die Abbildung mehrerer verschachtelter Hierarchien wird über die typbasierte Filterung möglich. Hierbei werden Nachrichten von Produzenten einem oder mehreren Nachrichtentypen zugeordnet. Nachrichtentypen sind außerdem verschachtelbar. Von BATES et al. (1998) werden die Typen zum Beispiel als Interfaces und die Verschachtelung über Vererbung realisiert.

Inhaltsbasierte Filterung

Der ausdrucksstärkste Filtermechanismus ist die inhaltsbasierte Filterung. Nachrichten werden ohne Zusatzinformationen veröffentlicht und die Auswahl findet über Filter statt, die eine Filterung direkt auf dem Inhalt der Nachricht durchführen. Die Ausdrucksstärke der Filter hängt nur noch vom verwendeten Nachrichtenformat und entsprechend dem Filtertyp ab. Beispiele sind Templates auf Zeichenketten (CUGOLA, DI NITTO und FUGGETTA,

2001), Filter aus Vergleichsoperationen auf Name-Wert-Paaren (SEGALL et al., 2000), XPath-basierende Filter auf XML-Dokumenten (ALTINEL und FRANKLIN, 2000) oder ausführbarer Programmcode auf beliebigen Daten (EUGSTER und GUERRAOUI, 2001).

2.1.2. Verbreitung von Nachrichten

Konzeptuell nimmt der Vermittler eine zentrale Rolle ein und entkoppelt die Kommunikation zwischen Produzenten und Konsumenten. In der Umsetzung hat eine schrittweise Verteilung der Vermittler-Rolle stattgefunden. Die einzelnen Instanzen des Vermittlers auf verschiedenen physikalischen Geräten werden nachfolgend als **Broker** bezeichnet. Um ein Netzwerk aus Brokern, die zusammengenommen wie ein zentraler Vermittler agieren sollen, zu realisieren, sind verschiedene Strategien etabliert (MÜHL, FIEGE und PIETZUCH, 2006, S.22f.):

Verbreitung aller Nachrichten

Alle Nachrichten werden an alle Broker verteilt. Jeder Broker leitet entsprechend der lokalen Registrierungen gefilterte Nachrichten an seine Konsumenten weiter. Dieses Vorgehen ist nur in sehr speziellen Situationen effizient, zum Beispiel wenn die Daten sehr weniger Produzenten an sehr viele Konsumenten verteilt werden sollen und eine effiziente Verteilungsstrategie existiert. In den meisten Fällen ist die Verbreitung aller Nachrichten an alle Broker ineffizient und kann zur Überlastung des Netzwerkes und der Broker führen.

Verbreitung allen Interesses

Da ein Konsument potentiell selten sein Interesse ändert aber viele Nachrichten empfängt, wird bei dieser Strategie das Interesse der registrierten Konsumenten in Form von Filtern an alle Broker verteilt. Filter beschreiben dabei, an welchen Nachrichten ein Broker interessiert ist. Daraus ergibt sich für jeden Broker eine Art Routingtabelle, in der beschrieben ist, welche Nachrichten an welche benachbarten Broker weiterzuleiten sind. Eine Nachricht wird anschließend nur an die benachbarten Broker weitergeleitet, von denen zuvor Interesse an den Inhalten der Nachricht empfangen wurde. Dieser Vorgang wird als **inhaltsbasiertes Routing** bezeichnet.

Hieraus ergeben sich zwei Probleme. Erstens muss jeder Filter aus der Routingtabelle auf jede einzelne Nachricht, die ein Broker von einem Produzenten oder anderen Broker empfängt, angewandt und anhand des Ergebnisses entschieden werden, ob diese Nachricht weiterzuleiten ist. Und zweitens können diese Routingtabellen sehr groß werden, also sehr viele Filter enthalten.

Um diesen Skalierungsproblemen zu begegnen, gibt es verschiedene Verfahren, die Anzahl der zu verwaltenden und weiterzuleitenden Filter weiter einzuschränken (MÜHL et al., 2002):

Identitätsbasiertes Routing - ein Filter wird nicht an einen Nachbarbroker weitergeleitet, an den schon einmal ein Filter weitergeleitet wurde, der auf die gleichen Nachrichten passt,

Überdeckungsbasiertes Routing - ein Filter wird nicht an einen benachbarten Broker weitergeleitet, wenn er nur auf eine Teilmenge der Nachrichten passt, die auf einen schon an diesen Nachbarn weitergeleiteten Filter passen[2] (CARZANIGA, ROSENBLUM und WOLF, 2001),

Zusammenführendes Routing - um die Anzahl der weitergeleiteten Filter weiter zu senken, wird anstelle mehrerer Filter ein zusammengeführter Filter, ein neuer Filter für eine Obermenge der auf die ursprünglichen Filter passenden Nachrichten, generiert und weitergeleitet. Dies kann sowohl ein vollkommener zusammengeführter Filter, der exakt auf alle Nachrichten passt, auf die zuvor die einzelnen Filter gepasst haben als auch ein unvollkommener zusammengeführter Filter sein. (MÜHL, 2002).

Weiterhin wurden spezielle Datenstrukturen definiert, um die entstehenden Routingtabellen möglichst effizient durchsuchen zu können (TARKOMA, 2006).

Verbreitung von Ankündigungen

Da beim inhaltsbasierten Routing von Nachrichten für jede Nachricht anhand der Routingtabelle aus Interesse entschieden werden muss, an welche Nachbarbroker diese Nachricht weitergeleitet werden muss, ist es sinnvoll, diese Routingtabellen so kurz wie möglich zu halten. Die Nutzung von Ankündigungen macht sich hierzu den Effekt zu Nutze, dass in einem Publish/Subscribe System mit hoher Wahrscheinlichkeit mehr Konsumenten für einen Nachrichtentyp existieren als Produzenten.

Der Verbreitung von Interesse wird hierbei eine Verbreitung von Ankündigungen vorgeschaltet. Für Ankündigungen werden die gleichen Filter benutzt wie zur Beschreibung von Interesse, allerdings wird hier nicht das Interesse eines Konsumenten, sondern das Angebot eines Produzenten beschrieben. Die Ankündigungen werden nach den gleichen Verbreitungsmechanismen verteilt wie das Interesse im vorangehenden Abschnitt. Das Interesse wird daraufhin wie ein Abonnement nur noch an Broker verbreitet, von denen Ankündigungen für die entsprechenden Inhalte empfangen wurden.

Durch dieses Vorgehen entstehen zwei verschiedene Routingtabellen in jedem Broker. Eine Routingtabelle entsteht aus den Ankündigungen aller Produzenten und wird genutzt, um die Verbreitung von Abonnements dahingehend einzuschränken, dass Abonnements nur dorthin versandt werden, wo die entsprechenden Nachrichten auch veröffentlicht werden. Bereits diese erste Routingtabelle kann deutlich kürzer werden als die Routingtabelle im vorhergehenden Abschnitt wenn im System deutlich weniger Produzenten als Konsumenten vorhanden sind.

[2]Dazu wird eine Überdeckungsrelation zwischen zwei Filtern definiert und die Filter werden vor dem Weiterleiten anhand dieser Überdeckungsrelation verglichen.

Die zweite Routingtabelle entsteht nun aus den verbreiteten Abonnements und wird durch die zuvor beschriebene gezieltere Verbreitung der Abonnements deutlich kürzer als wenn, wie im letzten Abschnitt, alles Interesse an jeden Broker verbreitet wird. Anhand der zweiten Routingtabelle werden dann die Nachrichten im System verbreitet.

Aufgrund der Entkopplung zwischen Produzenten und Konsumenten durch ein Netz aus Brokern, die sich über inhaltsbasiertes Routing koordinieren, sowie die hohe Ausdrucksstärke inhaltsbasierter Filterung eignet sich das Paradigma Publish/Subscribe hervorragend für die Nutzung in mobilen Umgebungen (HUANG und GARCIA-MOLINA, 2004).

2.2. Grundlegende Kommunikationsalgorithmen

In diesem Abschnitt werden grundlegende Kommunikationsalgorithmen, die im späteren Verlauf der Arbeit eine Rolle spielen, erklärt, deren Vor- und Nachteile diskutiert, sowie ihr Nutzen im Rahmen dieser Arbeit bewertet. Im Einzelnen sind dies Flooding, klassisches Gossiping und eine angepasste Form des Gossiping.

Folgende, sehr allgemeine Annahmen bezüglich der Netzwerktopologie liegen den Ausführungen zu Grunde:

- Zwei benachbarte Kommunikationsteilnehmer, kurz Nachbarn, sind mit einer Kommunikationstechnologie direkt verbunden. Sie können demzufolge Nachrichten ohne das Zutun eines dritten Kommunikationsteilnehmers austauschen.
- Alle derartigen Verbindungen im Netzwerk sind bidirektional, wobei jede Richtung mit einem eigenen Kostenfaktor entsprechend einer Metrik bewertet wird. Die Kosten für beide Richtungen einer bidirektionalen Verbindung können unterschiedlich sein. [3]
- Als Netzwerktopologie wird zu einem bestimmten Zeitpunkt, von einem bestimmten Kommunikationsteilnehmer aus betrachtet, die Menge aller anderen, über jeweils wenigstens einen zusammenhängenden Pfad aus Kommunikationsverbindungen erreichbaren, Kommunikationsteilnehmer bezeichnet.

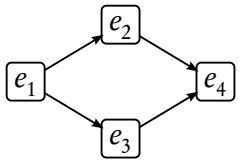
Diese Annahmen sind bewusst so allgemein formuliert, da sie für möglichst viele drahtgebundene wie drahtlose Kommunikationstechnologien gültig sein sollen.

Bei der Diskussion der Algorithmen wird die Netzwerktopologie demzufolge als attributierter, symmetrischer[4], zusammenhängender, schleifenfreier, gerichteter Graph

[3] Sind in einem Netzwerk Kommunikationsverbindungen vorhanden, die nur in eine Richtung nutzbar sind, so gibt es zwei Möglichkeiten: Entweder sie sind durch bestehende Techniken auf einfache Weise in bidirektionale Verbindungen umzuwandeln, zum Beispiel indem zwei Verbindungen zwischen zwei Geräten aufgebaut werden, und können somit als bidirektionale Verbindung abstrahiert werden oder sie werden nicht als Verbindungen betrachtet, da der Aufwand, sie nutzen zu können, die Komplexität eines entsprechenden Kommunikationsalgorithmus so stark erhöht, dass der Mehrwert der Nutzung einer solchen Verbindung fragwürdig ist.

[4] symmetrisch in Bezug auf das Vorhandensein jeweils entgegengesetzt gerichteter Kanten zwischen zwei Knoten, nicht darauf, dass diese mit demselben Kostenfaktor belegt sind.

Abbildung 2.2:
Implosion beim Flooding: Wird eine Nachrichtenkopie von Ecke e_1 an ihre Nachbarn e_2 und e_3 versandt und beide haben Ecke e_4 als Nachbarn, so erhält e_4 die Kopie doppelt. Eine der Übertragungen ist redundant und Ressourcen werden vergeudet.



$G = (E, K, w)$ betrachtet. Jeder Kommunikationsteilnehmer wird durch eine Ecke $e \in E$ im Graphen abgebildet. Die Kommunikationsverbindungen zwischen benachbarten Kommunikationsteilnehmern $e \in E$ und $f \in E$ werden durch jeweils zwei Kanten aus $K \subset E \times E$, nämlich (e, f) und (f, e) abgebildet. Die beiden Kanten bilden zusammen eine bidirektionale Kommunikationsverbindung ab. Die Anzahl der Ecken wird mit $|E|$ und die Anzahl der Kanten mit $|K|$ bezeichnet. Die Attributfunktion w bildet jede Kante $k \in K$ auf ein Kantengewicht $w(k)$ ab. Dieses repräsentiert die Kommunikationskosten entlang einer Kante. Durch die, durch die bidirektionale Kommunikation entstehende Symmetrie des Graphen gilt $|K| = 2n$, wobei n die Anzahl der bidirektionalen Kommunikationsverbindungen bezeichnet. Die Menge der benachbarten Ecken für eine Ecke $e \in E$ wird mit N_e bezeichnet.

2.2.1. Flooding

Ziel des Flooding-Algorithmus ist es, eine Nachricht, die auf einem Kommunikationsteilnehmer gespeichert ist, möglichst schnell unter allen Teilnehmern in einem Kommunikationsnetzwerk zu verbreiten. Dazu wird von jedem Teilnehmer, nachdem er eine Kopie dieser Nachricht zum ersten Mal empfangen hat, an jeden seiner Nachbarn, außer den Absender der empfangenen Kopie, eine weitere Kopie der Nachricht versandt. Weitere empfangene Nachrichtenkopien werden ignoriert.

Das Flooding hat zwei Vorteile: Es garantiert, dass eine Kopie der Nachricht bei jeder momentan erreichbaren Ecke ankommt, wenn das Netzwerk durch den Versand der Nachrichtenkopien nicht überlastet wird und dadurch einzelne Verbindungen ausfallen. Außerdem ist es ein beinahe zustandsloser Algorithmus. Eine Ecke muss sich lediglich für eine relativ kurze Zeit merken, ob sie schon einmal eine Kopie einer bestimmten Nachricht erhalten hat oder nicht.

Der wesentliche Nachteil beim Flooding ist die Implosion (HEINZELMAN, KULIK und BALAKRISHNAN, 1999). Nachrichtenkopien werden häufiger übertragen als notwendig, um alle Ecken zu erreichen. Abbildung 2.2 skizziert dieses Problem anhand eines einfachen Beispiels.

Flooding zur Pfadsuche im Netzwerk

Die Implosion kann auch sinnvoll genutzt werden. Anhand des Beispiels in Abbildung 2.2 soll das Vorgehen kurz beschrieben werden. Wenn in den Nachrichtenkopien ein Kostenmaß

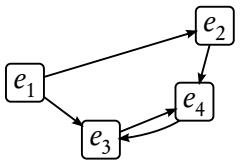


Abbildung 2.3:
Wenn der kostengünstigste Pfad von e_1 zu e_2 der Pfad über e_3 und e_4 ist, so würde dieser Pfad durch Flooding mit der durch die Pfeile skizzierten Nachrichtenverteilung nicht gefunden werden. Man beachte, dass in der Skizze nicht der Graph dargestellt ist, sondern die Übertragungswege der Nachrichtenkopien. Der zu Grunde liegende Graph ist symmetrisch und ein Pfad e_1, e_3, e_4, e_2 existiert selbstverständlich.

für die jeweilige Verbindung von e_1 gespeichert ist, so erkennt e_4, wenn es die zweite Kopie der Nachricht erhält, dass es wenigstens zwei mögliche Pfade von e_1 gibt und welcher der beiden Pfade von e_1 kostengünstiger ist.

Dieses Vorgehen findet den global kostengünstigsten Pfad aber nur dann, wenn das Netz **synchron** ist, also die Zeit für die Übertragung der Nachrichtenkopien auf einem Pfad mit dessen Kosten in einem annähernd linearen Verhältnis zueinander steht. Im Beispiel in Abbildung 2.3 wären die Pfadkosten $w(e_1, e_2)$ sehr hoch, während die Nachrichtenkopien auf diesem Pfad aber sehr schnell zugestellt werden. Das entsprechende Netzwerk wäre nicht synchron und der kostengünstigste Pfad würde nicht gefunden. Bereits für ein kabelgebundenes Netzwerk mit konstanten Pfadkosten von 1 ist die Annahme, es wäre synchron, nicht realistisch (AWERBUCH und GALLAGER, 1987).

Eine Alternative ist eine einfache verteilte Umsetzung des Bellmann-Ford-Algorithmus (KLEINBERG und TARDOS, 2006, S.292ff.), nachfolgend als **erweitertes Flooding** bezeichnet. Die Funktionsweise ist identisch zum Flooding, nur dass sich jede Ecke das beste bisher erhaltene Pfadmaß merkt. Sobald sie eine Kopie mit einem besseren Pfadmaß erhält, wird ebenfalls eine Kopie an alle Nachbarn außer deren Absender versandt (AWERBUCH und GALLAGER, 1987).

Beide Formen des Flooding haben den Nachteil, dass aus Sicht eines Kommunikationsteilnehmers zu keinem Zeitpunkt bekannt ist, ob der bisher ermittelte kürzeste Pfad bereits der global kürzeste Pfad ist oder ob noch weitere Nachrichtenkopien zu erwarten sind.

Theoretische Komplexitätsbetrachtung

Die Komplexität des Flooding lässt sich anhand der Anzahl der Kommunikationsteilnehmer und der Kommunikationsverbindungen ausdrücken. Die Anzahl der benötigten Nachrichtenkopien ergibt sich dabei als $t = |K| - |E| + 1 = 2n - |E| + 1$. Details zur Herleitung finden sich im Anhang A.

Für einen zusammenhängenden Graphen mit minimaler Kantenzahl $n = |E| - 1$ ergibt sich $t = |E| - 1$. t liegt somit in der Komplexitätsklasse $t = \Omega(|E|)$. Für einen vollständigen Graphen mit maximaler Kantenzahl $n = \frac{1}{2}|E|(|E| - 1)$ ergibt sich $t = (|E| - 1)^2$. t liegt also in der Komplexitätsklasse $t = O(|E|^2)$. Je nach Beschaffenheit der Netzwerktopologie ist die

Komplexität der Anzahl zu versendender Nachrichtenkopien beim Flooding somit linear bis quadratisch in $|E|$, also in Bezug auf die Anzahl der Kommunikationsteilnehmer.

Die Komplexität des erweiterten Flooding hängt ebenfalls von der Topologie aber zusätzlich von der Synchronizität[5] des Netzwerkes, also dem Zusammenhang zwischen den Kantengewichten $w(k)$ und der tatsächlichen Übertragungszeit einer Nachrichtenkopie auf dieser Kante, ab. Es lassen sich hier rechnerisch ebenfalls untere und obere Grenzen bestimmen. Im besten Fall ist das Netzwerk vollständig synchron und für das erweiterte Flooding werden keine weiteren Nachrichtenkopien benötigt. Die Komplexität fällt also mit der des normalen Flooding zusammen: $t = \Omega(|E|)$.

Der ungünstigste Fall ist der, dass in einem vollständig verbundenen Graphen jede Ecke eine Nachrichtenkopie zuerst auf dem Pfad mit den höchsten Pfadkosten erhält und dann sukzessive auf dem jeweils am wenigsten besseren Pfad. In diesem Fall müsste sie jede erhaltene Nachrichtenkopie auch entsprechend der Vorschrift weiterleiten. Es wird folgend angenommen, dass sich für jeden Graphen wenigstens eine Belegung von Kantengewichten finden lässt, so dass dieser ungünstigste Fall eintritt. Dies wird im Rahmen dieser Arbeit nicht bewiesen, da es für die Ermittlung einer oberen Grenze nicht von Bedeutung ist. Sollte sich eine derart ungünstige Belegung von Kosten und Übertragungszeiten nicht für jeden Graphen finden lassen, so wäre die ermittelte Komplexitätsklasse noch immer eine obere Grenze. Für den Fall der ungünstigsten Belegung lässt sich die Anzahl der benötigten Nachrichtenkopien wie folgt bestimmen:

Es seien $q \in E$ die Quelle, also die Ecke an der die Nachricht eingespeist wird, $|P(q,e)|$ die Anzahl der möglichen Pfade[6] zwischen $q \in E$ und $e \in E$, also auch die Anzahl der Nachrichtenkopien, die e erhält, sowie $|N_e|$ die Anzahl der Nachbarn der Ecke e. Daraus ergibt für die Anzahl der insgesamt benötigten Nachrichtenkopien als

$$t(q) = |N_q| + \sum_{e \in E \setminus \{q\}} |P(q,e)|(|N_e| - 1).$$

Für den betrachteten vollständig verbundenen Graphen gilt $|N_q| = |N_e| = |E| - 1$. Außerdem lässt sich $|P(q,e)|$ über die möglichen Kombinationen der anderen Ecken herleiten. Es gibt genau einen Pfad der Länge 1 direkt von q nach e, $|E| - 2$ Pfade der Länge 2 über jede andere Ecke außer q und e. Kombinatorisch handelt es sich um eine Auswahl mit Beachtung der Reihenfolge ohne Zurücklegen von i aus $|E| - 2$ Ecken für alle i von 0 bis $|E| - 2$. Die entsprechende Formel für $|E| \geq 2$ ist:

$$|P(q,e)| = \sum_{i=0}^{|E|-2} \frac{(|E|-2)!}{(|E|-2-i)!}$$

[5] Vergleiche die Beschreibung eines synchronen Netzwerkes im Abschnitt 2.2.1

[6] Hier ist die Anzahl der Pfade gemeint, die der Flooding-Algorithmus benutzen kann, also die jeden Knoten außer $q \in E$ und $e \in E$ höchstens einmal enthalten.

Daraus ergibt sich[7] für t:

$$
\begin{aligned}
t(q) &= (|E|-1)+\sum_{e\in E\setminus\{q\}}\Big[\Big(\sum_{i=0}^{|E|-2}\frac{(|E|-2)!}{(|E|-2-i)!}\Big)(|E|-2)\Big] \text{ für } |E|\geq 2\\
\Leftrightarrow t &= (|E|-1)+(|E|-2)(|E|-1)!\sum_{i=0}^{|E|-2}\frac{1}{(|E|-2-i)!}\\
&\quad \text{mit } \sum_{i=0}^{|E|-2}\frac{1}{(|E|-2-i)!}<\mathrm{e}\\
t &< (|E|-1)+(|E|-2)(|E|-1)!\mathrm{e}\\
&< (|E|-1)+|E|(|E|-1)!\mathrm{e}\\
&= (|E|-1)+\mathrm{e}|E|!\\
&\quad \text{mit } |E|!>|E|-1 \text{ für } |E|\geq 2\\
&< (\mathrm{e}+1)|E|!\\
&= c|E|! \quad \text{mit konstantem } c=\mathrm{e}+1\\
t &= O(|E|!)
\end{aligned}
$$

Die Komplexität für den ungünstigsten Fall erweiterten Floodings ist somit $t = O(|E|!)$.

In der Praxis sind vollständig verbundene Graphen die Ausnahme und die Annahme, dass alle Nachrichtenkopien alle Ecken in umgekehrter Reihenfolge der Pfadgüte erreichen, ist ebenfalls nicht realistisch. Aber die enorme Verschlechterung der Komplexität lässt erahnen, dass erweitertes Flooding in der Praxis nicht sinnvoll nutzbar ist. Dem gegenüber steht die Aussage von AWERBUCH und GALLAGER (1987), dass eine starke Asynchronizität[8] fast nie vorkommt und somit nie mehr als wenige Nachrichtenkopien pro Kante übertragen werden. Diese ist insbesondere für drahtlose Netze aufgrund der gegenseitigen Beeinflussung der Übertragungen nah beieinander gelegener Geräte durch den geteilten Zugriff auf das Medium wahrscheinlich nicht haltbar.

Simulative Evaluierung der Komplexität

Eine starke Asynchronizität bei drahtlosen Netzwerken, verbunden mit einer deutlichen Verschlechterung der Komplexität erweiterten Floodings wird nachfolgend in einer einfachen Simulation gezeigt. Gemessen wird die Anzahl der von allen Geräten tatsächlich empfangenen Kopien der zu verteilenden Nachricht mit einer Länge von 26 Byte in einem drahtlosen Netz. Dazu sind auf einer quadratischen Fläche mit der Kantenlänge 200m x stationäre Geräte zufällig verteilt. Diese kommunizieren über IEEE802.11 WLAN im ad-hoc-Modus

[7] Man beachte den Unterschied zwischen e (Ecke) und e (eulersche Zahl).
[8] Vergleiche die Beschreibung eines synchronen Netzwerkes im Abschnitt 2.2.1

mit einer Geschwindigkeit von 11Mb/s. Die Eingangsempfindlichkeit des Empfängers ist dabei so konfiguriert, dass sich eine Reichweite von 300m ergibt. Aufgrund des maximalen Abstandes zweier Geräte von 283m (Diagonale) bilden die Kommunikationsverbindungen immer einem vollständigen Graphen.

An einem der Geräte wird die zu verteilende Nachricht eingespeist und entsprechend dem erweiterten Flooding-Algorithmus verbreitet. Dabei wird die Anzahl der Geräte ermittelt, die Kopien der Nachricht erhalten, die Anzahl der von allen Geräten empfangenen Nachrichtenkopien, die auch bei einem einfachen Flooding empfangen würden, sowie die Anzahl der Nachrichtenkopien, die zusätzlich durch das erweiterte Flooding empfangen wurden. Abbildung 2.4 zeigt das Ergebnis für $x = 5, 10, 15, \ldots, 45$ und 50 Geräte.

Das Experiment wurde pro gewählter Gerätezahl mit zehn verschiedenen zufälligen Topologien durchgeführt. In jeder Topologie wurden zehn Nachrichten mit ausreichend zeitlichem Abstand[9] eingespeist. Jede eingespeiste Nachricht entspricht einem schmalen Balken in der Abbildung. Die zehn Balken pro zufälliger Topologie sind gruppiert. Man sieht deutlich, dass sich die Anzahl der für das erweiterte Flooding nötigen Nachrichtenkopien mit jeder verteilten Nachricht ändert. Dies liegt darin begründet, dass sich sowohl die Verbindungskosten zwischen den Geräten durch das Verteilen der Nachrichten selbst verändern[10], als auch die Übertragungszeiten der Nachrichtenkopien auf derselben Verbindung durch Interferenz dritter Geräte beeinflusst wird.

Die Anzahl der für das einfache Flooding nötigen Nachrichtenkopien sollte lediglich von der Topologie, genauer der Anzahl der Kanten im Graphen abhängen. Dies sieht man in den Ergebnissen mit bis zu 30 Geräten ebenfalls sehr deutlich. Die Anzahl der für das einfache Flooding benötigten Nachrichtenkopien entspricht exakt der theoretisch vorhergesagten. Angefangen bei 35 Geräten ändert sich dieses Verhalten. Das Netz wird durch das erweiterte Flooding so stark belastet, dass nicht mehr alle Nachrichten zugestellt werden können.

Es ist insgesamt deutlich zu erkennen, dass die Aussage von AWERBUCH und GALLAGER (1987), dass eine starke Asynchronizität fast nie vorkommt und somit nie mehr als wenige Nachrichtenkopien pro Kante übertragen werden, für drahtlose Netze so nicht stimmt. Je nach Anzahl der Geräte und Grad der Vermaschung benötigt das erweiterte Flooding ein Vielfaches an Nachrichtenkopien gegenüber dem normalen Flooding und führt sehr schnell zur Überlastung einzelner Verbindungen.

Weiterhin hängt die Anzahl der benötigten Nachrichtenkopien neben der Anzahl der Kanten im Graphen auch von der Belegung der Kantengewichte ab. Diese verändert sich sehr viel häufiger als die Anzahl der Kanten. Damit ist das Verhalten des erweiterten Floodings in Bezug auf die entstehende Netzwerklast auch sehr viel schwerer vorherzusagen als beim

[9] Der Abstand ist so gewählt, dass eine weitere Nachricht erst eingespeist wird, wenn die letzte Nachricht unter allen Geräten verteilt ist und keine Kommunikation mehr stattfindet.

[10] Die Verbindungskosten berechnen sich aus der Roundtriptime der bisher versandten Nachrichten und verändert sich somit durch das Versenden von Nachrichten.

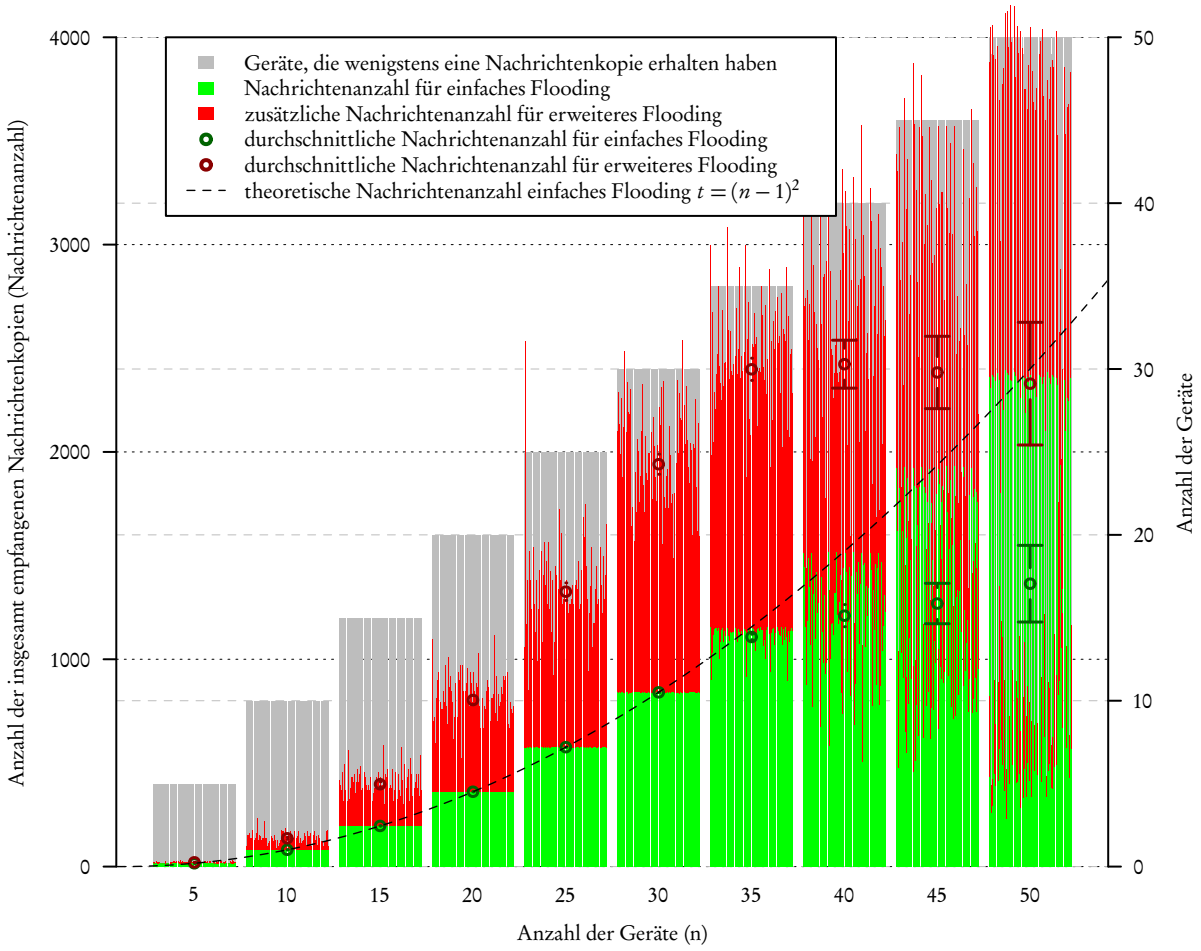


Abbildung 2.4.:
Simulation der Komplexität des erweiterten Floodings. Die rechte Y-Achse beschreibt lediglich die grauen Balken. Die einzelnen grünen und roten Balken sind sehr schmal. Jeweils 10 Balken beschreiben die zehn Nachrichten in einem Experiment und sind durch einen kleinen Abstand von den nächsten 10 getrennt. Die Durchschnitte beziehen sich auf alle Experimente pro auf der X-Achse notierter Konfiguration und sind mit 95% Konfidenzintervallen versehen.

einfachen Flooding, dessen Komplexität in Abhängigkeit von der Anzahl der Geräte sehr klar abschätzbar ist.

2.2.2. Gossiping

Gossiping ist ein Verfahren, das ursprünglich zur Verteilung von Datenbankupdates unter allen Geräten in einem Netzwerk entworfen wurde (DEMERS et al., 1987). Im Folgenden wird es als **klassisches Gossiping** bezeichnet. Es funktioniert wie folgt: Erhält eine Ecke eine Kopie der Nachricht, so wählt sie zufällig einen ihre Nachbarn aus und übermittelt eine

weitere Kopie der Nachricht an diesen. Dieser Nachbar antwortet mit der Information, ob dies die erste Kopie der Nachricht ist, die er empfangen hat, oder ob ihm die Nachricht schon bekannt war. Das Auswählen von Nachbarn und übermitteln einer Kopie wird regelmäßig fortgesetzt, bis entweder eine Kopie an jeden Nachbarn versandt wurde, oder die Anzahl der Nachbarn, die die Nachricht bereits kannten, einen Schwellwert übersteigt.

Im Unterschied zum Flooding hat das klassische Gossiping den Vorteil, dass das Netz nicht augenblicklich durch eine Flut von Nachrichten belastet wird. Stattdessen wird die Nachricht wie ein Gerücht zwischen Personen langsam unter allen Geräten verbreitet. Nach einer gewissen Laufzeit haben alle Geräte mit sehr hoher Wahrscheinlichkeit eine Kopie der Nachricht erhalten.

Eignung zur Pfadsuche im Netzwerk

Theoretisch lassen sich auch per klassischem Gossiping Pfade in einem Netzwerk finden. Allerdings spiegelt der Algorithmus aufgrund der Pausen zwischen den Nachrichtenübermittlungen an andere Nachbarn eher eine Art rekursiver Tiefensuche im Netzwerk wieder, bei der so lange zufällig weiter ins Netz vorgestoßen wird, bis ein Zyklus entstanden ist. Dieser wird ignoriert und nach einer Pause wird von jedem Punkt auf dem bisher gefundenen Pfad wieder per Tiefensuche vorgestoßen bis ein Zyklus entstanden ist.

Es ist davon auszugehen, dass nach einiger Zeit ein erster Pfad etabliert werden kann, der aber aufgrund der Tiefensuche-Eigenschaft wahrscheinlich nicht kostengünstig ist. Erst nach einigen Pausenzeiten entstehen mehr Pfade und die Wahrscheinlichkeit steigt, dass ein kostengünstigerer Pfad darunter ist.

Der größte Nachteil des klassischem Gossiping in Bezug auf die Suche nach Pfaden in einem Netzwerk ist somit die hohe zeitliche Verzögerung zwischen der Initiierung der Pfadsuche durch die Quelle und dem Zustandekommen eines geeigneten und dann eines kostengünstigen Pfades bis zur Senke.

Gossiping in Ad-Hoc Netzwerken

Speziell für den Einsatz in ad-hoc Netzen haben sich andere Ansätze des Gossiping etabliert (HAAS, HALPERN und LI, 2006). Hierbei wird die Nachrichtenkopie ähnlich dem Flooding in einer Breitensuche an mehrere Nachbarn verteilt. Allerdings wird entweder die Anzahl der Nachbarn, an die eine weitere Kopie versandt wird, die Tatsache ob eine Kopie an einen bestimmten Nachbarn versandt wird oder die Wahrscheinlichkeit ob eine Kopie an alle oder keinen Nachbarn versandt wird, anhand einer Wahrscheinlichkeitsfunktion bestimmt. In diese Wahrscheinlichkeitsfunktion können Parameter wie zum Beispiel die bisherige Pfadlänge, die Anzahl der Nachbarn der aktuellen Ecke oder der Ecke, von dem die Nachrichtenkopie empfangen wurde, eingehen.

Die Grundidee beim Gossiping ist es, durch die Nutzung einer Wahrscheinlichkeitsfunktion die Implosion gegenüber dem Flooding zu verringern. Trotzdem soll ein Kommunikati-

onspartner mit geringer zeitlicher Verzögerung gefunden werden. In HAAS, HALPERN und LI (2006) wird das bimodale Verhalten des Gossiping festgestellt. Entweder die Nachricht stirbt schnell aus weil die Weiterleitungswahrscheinlichkeit zu gering war, oder die Nachricht wird mit hoher Wahrscheinlichkeit an sehr viele Geräte übertragen. Verschiedene regelmäßige Topologien werden untersucht und je nach Wahrscheinlichkeitsfunktion eine Grenze für das bimodale Verhalten von 0,6 bis 0,75 festgestellt.

Durch das Gossiping lassen sich zur Suche eines Pfades zwischen Quell- und Zielknoten bis zu 35% der Nachrichten sparen (HAAS, HALPERN und LI, 2006). Allerdings werden durch die gesparten Nachrichten auch weniger mögliche Pfade gefunden aus denen der beste verfügbare Pfad ausgewählt werden kann. Das kann dann im späteren Verlauf der Kommunikation mehr Nachrichtenübertragungen bzw. Nachrichtenübertragungen auf Pfaden mit schlechteren Kosten zur Folge haben.

2.3. Zusammenfassung

In diesem Kapitel wurden Publish/Subscribe als Kommunikationsparadigma erklärt und verschiedene grundlegende Kommunikationsalgorithmen vorgestellt und analysiert. Die wichtigsten Erkenntnisse werden nachfolgend zusammengefasst.

Die mächtigste Form, Interesse in Publish/Subscribe-Systemen auszudrücken, ist die inhaltsbasierte Filterung. Unterstützt ein Publish/Subscribe-System inhaltsbasierte Filterung, so kann jeder andere Filtermechanismus darauf abgebildet werden.

Die Verbreitung von Interesse und Ankündigungen in Publish/Subscribe-Systemen können durch identitätsbasiertes, überdeckungsbasiertes sowie zusammenführendes Routing weiter optimiert werden. Eine Umsetzung von inhaltsbasiertem Routing ohne diese Optimierungen stellt somit den ungünstigsten Fall in Bezug auf Netzwerklast und Skalierbarkeit dar.

Erweitertes Flooding erzeugt im Gegensatz zum einfachen Flooding eine unkalkulierbar größere Belastung der Netzwerkinfrastruktur, die mit sehr hoher Wahrscheinlichkeit den Vorteil, alle möglichen Pfade zwischen zwei Ecken zu finden, nicht aufwiegt. Das einfache Flooding ist demzufolge zur Verbreitung von Nachrichten in einer Topologie vorzuziehen.

Durch Gossiping-Verfahren kann die Belastung der Netzwerkinfrastruktur gegenüber dem Flooding weiter gesenkt werden, da der Umfang der Implosion verringert wird. Insbesondere in speziellen, stark vermaschten Topologien kann hiermit die erzeugte Last stark vermindert werden. Die Nutzung von Flooding als Basis für einen entsprechenden Algorithmus stellt also auch hier den ungünstigsten, allgemeinsten Fall dar.

Kapitel 3.

Entkopplung - anwendungsbezogene Taxonomie für Kommunikationsverfahren

Inhalt

3.1 Verwandte Arbeiten **28**
3.1.1 Entkopplung in Raum, Zeit und Synchronisation 29
3.1.2 Formalisierung von Kopplungseigenschaften 30
3.2 Dimensionen der Entkopplung **31**
3.2.1 Räumliche Entkopplung 31
3.2.2 Zeitliche Entkopplung 32
3.2.3 Entkopplung des Produzenten 34
3.2.4 Entkopplung des Konsumenten 35
3.2.5 Inhaltliche Entkopplung 37
3.3 Klassifizierung bestehender Kommunikationsparadigmen **39**
3.3.1 Paketbasierte Kommunikation 40
3.3.2 Verbindungsorientierte Kommunikation 40
3.3.3 Abstraktion durch Funktionen/Methoden 41
3.3.4 Nachrichtenbasierte Kommunikation 42
3.4 Anforderungen entsprechend der Problemstellung **45**
3.5 Zusammenfassung **46**

Um zu beurteilen, welche Kommunikationsparadigmen sowie deren Implementierungen sich zur Realisierung des Datenaustausches zwischen Anwendungen in einer gegebenen Problemstellung eignen, ist es sinnvoll, diese nach anwendungsbezogenen Kriterien zu bewerten. Ein Kriterium, dass insbesondere für Anwendungen in heterogenen, veränderlichen Netzwerktopologien von großer Bedeutung ist, ist die Kopplung bzw. Entkopplung.

Je enger Anwendungen an entfernte Anwendungen, Services oder Middleware-Komponenten gekoppelt sind, desto größer ist die Abhängigkeit von einer stabilen Kommunikationsverbindung zwischen einzelnen Geräten. Diese ist aber gerade in heterogenen, veränderlichen Netzwerktopologien nicht gegeben.

Auch in Bezug auf den Datenaustausch ist eine Entkopplung von Anwendungen sinnvoll um Interoperabilität sicherzustellen und die spontane Interaktion verschiedener Anwendungen in einer gemeinsamen Umgebung zu erleichtern bzw. zu ermöglichen.

Bestehende Taxonomien zur Unterteilung von Kommunikationsparadigmen und deren Implementierungen anhand derer Kopplungseigenschaften aus Anwendungssicht sind zum Teil widersprüchlich und nicht ausreichend. Daher wird in diesem Kapitel eine neue Taxonomie vorgestellt, existierende Kommunikationsparadigmen sowie die Problemstellung der Arbeit anhand dieser Taxonomie beurteilt und dadurch das für die Lösung des Problems am besten geeignete Kommunikationsparadigma bestimmt.

3.1. Verwandte Arbeiten

Auslöser für das Aufkommen von Taxonomien, die die Entkopplung von Kommunikationsparadigmen betrachten, ist die Unzulänglichkeit des Begriffes der Synchronisation. Synchronisation in der Interprozesskommunikation bezeichnet ein Warten eines Prozesses auf ein Ereignis, dass von einem anderen Prozess ausgelöst wird und dient der Gewährleistung der Kausalität nebenläufiger Prozesse. In der zwischenmenschlichen Kommunikation bezeichnet Synchronisation, dass beide Kommunikationspartner gleichzeitig verfügbar sind während sie miteinander kommunizieren. So gewährleistet ein Telefonat zum Beispiel eine synchrone Kommunikation zwischen entfernten Personen, während der Austausch per Email eine asynchrone Kommunikation zulässt.

In der Kommunikation auf Anwendungsebene ist der Begriff Synchronisation nicht klar definiert. Sowohl die gleichzeitige Verfügbarkeit beider Kommunikationspartner, als auch das blockierende Warten aufeinander spielen hier eine Rolle. Diese beiden und andere Formen der Kopplung zwischen Anwendungen wurden in EUGSTER et al. (2003) getrennt voneinander betrachtet. Eine Formalisierung und Erweiterung wurde in ALDRED et al. (2005) und ALDRED et al. (2009) durchgeführt.

Kommunikationsparadigma	Raum	Zeit	Synchronisation
Message Passing	Nein	Nein	Produzent
RPC/RMI	Nein	Nein	Produzent
Asynchrones RPC/RMI	Nein	Nein	Ja
Benachrichtigung (Observer-Pattern)	Nein	Nein	Ja
Tuple Spaces	Ja	Ja	Produzent
Message Queueing (Pull)	Ja	Ja	Produzent
Publish/Subscribe	Ja	Ja	Ja

Tabelle 3.1.:
Untersuchung der Kommunikationsparadigmen Message Passing (TANENBAUM und STEEN, 2007, S.142ff), RPC/RMI (PETERSON und DAVIE, 2007, S.411ff), Observer-Pattern (GAMMA et al., 1995, S.293ff), Tuple Spaces (GELERNTER, 1985), Message Queueing (TANENBAUM und STEEN, 2007, S.145ff) sowie Publish/Subscribe auf ihre Fähigkeit zur Gewährleistung von Entkopplung nach EUGSTER et al. (2003).

3.1.1. Entkopplung in Raum, Zeit und Synchronisation

Als Taxonomie zur Differenzierung von Kommunikationsparadigmen wird in EUGSTER et al. (2003) die Fähigkeit einer Middleware zur Realisierung von Entkopplung zwischen Anwendungen vorgeschlagen. Dabei wird zwischen den Dimensionen Raum, Zeit und Synchronisation unterschieden. Die räumliche und zeitliche Entkopplung werden nicht weiter untergliedert. Sie beschreiben lediglich, dass die Kommunikationspartner einander nicht kennen bzw. nicht gleichzeitig verfügbar sein müssen.

Synchronisationsentkopplung wird in die Seite des Produzenten und die des Konsumenten untergliedert, wobei jeweils die Entkopplung des Kontrollflusses betrachtet wird. Der Produzent gilt als entkoppelt, wenn er nicht-blockierend senden kann und der Konsument gilt als entkoppelt, wenn er nicht-blockierend über die Verfügbarkeit neuer Informationen benachrichtigt wird.

Entsprechend dieser Taxonomie werden die untersuchten Kommunikationsparadigmen, wie in Tabelle 3.1 dargestellt, nach ihrer Fähigkeit zur Gewährleistung von Entkopplung eingeordnet. Mit Hilfe dieser Einordnung wird argumentiert, dass Publish/Subscribe das einzige Kommunikationsparadigma ist, dass sowohl Entkopplung in Raum, Zeit als auch Synchronisation ermöglicht.

Eine detaillierte Differenzierung der Entkopplungsdimensionen ist in dieser Taxonomie nicht möglich. So sind einige der Einordnungen nicht direkt nachvollziehbar. Zum Beispiel wird die Identität der Kommunikationspartner bei RMI durchaus vor den Anwendungen verborgen. Die Abstraktion der Kommunikation als Methodenaufruf erlaubt keine Identifizierung des Kommunikationspartners und das vorhergehende Registrieren bzw. Nach-

schlagen des entfernten Objektes geschieht über eine dritte Instanz, die RMI Registry (SUN MICROSYSTEMS, INC.).

Auch eine Unterscheidung zwischen den sehr unterschiedlichen zeitlichen Entkopplungsformen zwischen zum Beispiel Tuple Spaces (GELERNTER, 1985), die ähnlich einer Datenbank Nachrichten permanent speichern, und Publish/Subscribe, bei dem lediglich die Übertragung an alle registrierten Konsumenten sichergestellt wird, ist nicht möglich.

Zuletzt ist die Entkopplungsdimension der Synchronisation als Entkopplung des Kontrollflusses grundsätzlich zu hinterfragen. Geht es nur um den Kontrollfluss einer kommunizierenden Anwendung, so kann dieser auch ohne Middleware jederzeit durch die Nutzung eines weiteren Threads vom Kommunikationsaufwand entkoppelt werden. Die Entkopplung des Kontrollflusses ist also nur eine Frage der Implementierung, und zwar der Implementierung der kommunizierenden Anwendung.

Es wird bei der Synchronisationsentkopplung des Konsumenten die Fähigkeit der Middleware zur Benachrichtigung des Konsumenten bei Verfügbarkeit einer Information genannt. Hier könnte man argumentieren, dass sich eine Benachrichtigung durch die Middleware von einem zusätzlichen Thread, der blockierendes Lesen in eine Benachrichtigung umwandelt, unterscheidet. Allerdings ist eine Benachrichtigung durch die Middleware nur eine Frage der Implementierung und nicht des Kommunikationsparadigmas. So sind zum Beispiel bestimmte Implementierungen von Tuple Spaces durchaus in der Lage, eine Anwendung über die Verfügbarkeit eines gewünschten Tupels per Callback zu informieren (FREEMAN, HUPFER und ARNOLD, 1999).

3.1.2. Formalisierung von Kopplungseigenschaften

In ALDRED et al. (2005) werden dieselben Dimensionen der Kopplung und Entkopplung, Raum, Zeit und Kontrollfluss auf Seiten des Produzenten und des Konsumenten, mit Hilfe von Colored Petri Nets formal definiert. Anschließend werden hauptsächlich konkrete Implementierungen verschiedener Kommunikationsparadigmen, aber auch Paradigmen selbst auf ihre Fähigkeiten zur Gewährleistung von Kopplung und Entkopplung in den jeweiligen Dimensionen hin untersucht und kategorisiert.

Auffällig sind bei der Kategorisierung signifikante Unterschiede zwischen der Einordnung einer Implementierung eines Kommunikationsparadigmas und der Einordnung desselben Paradigmas in EUGSTER et al. (2003). So wird in ALDRED et al. (2005) zum Beispiel MPI (MESSAGE PASSING INTERFACE FORUM) als Standardisierung des Message Passing eine Synchronisationsentkopplung auf Seite des Produzenten und des Konsumenten bescheinigt, JavaSpaces (FREEMAN, HUPFER und ARNOLD, 1999) als Implementierung von Tuple Spaces eine Entkopplung ausschließlich auf der Seite des Konsumenten, sowie CORBA (OBJECT MANAGEMENT GROUP) als Implementierung des RPC/RMI-Konzeptes die Fähigkeit zur zeitlichen wie räumlichen Entkopplung.

Diese widersprüchlichen Ergebnisse untermauern die These, dass die Fähigkeit einer Middleware zur Entkopplung des Kontrollflusses als Bewertungskriterium für Kommunikationsparadigmen nicht sinnvoll nutzbar ist.

In ALDRED et al. (2009) werden die Dimensionen der Kopplung/Entkopplung aufgegeben und hin zu Kommunikationspattern entwickelt. Pattern sind dabei die bereits definierten, zeitliche Kopplung, zeitliche Entkopplung, blockierendes Senden, nicht blockierendes Senden, blockierendes Empfangen, nicht blockierendes Empfangen sowie räumliche Kopplung.

Die räumliche Entkopplung wird in zwei Pattern aufgelöst, kanalbasierte räumliche Entkopplung und themenbasierte räumliche Entkopplung, wobei sich die Kanäle auf eine 1:1-Übertragung zwischen einander unbekannten Kommunikationspartnern beziehen und die Themen auf eine 1:n-Übertragung.

Zu diesen neun Basispattern kommen noch jeweils ein Pattern zur Beschreibung von Multicast und Aggregation sowie sechs Pattern zur Beschreibung von Anfrage-Antwort-Varianten, wie zum Beispiel eine Bestätigung vor bzw. nach der Verarbeitung.

Die zusätzlichen Pattern beziehen sich sehr stark auf nachrichtenbasierte Kommunikation. Es werden auch nur noch Implementierungen nachrichtenbasierter Kommunikationsparadigmen analysiert. Der Grad an Spezialisierung widerstrebt einer Nutzung als allgemeingültige Taxonomie, löst aber die im vorhergehenden Abschnitt genannten Probleme nicht. Lediglich die Untergliederung von räumlicher Entkopplung in 1:1 und 1:n ermöglicht eine etwas exaktere Einordnung.

3.2. Dimensionen der Entkopplung

In dieser Arbeit wird Entkopplung in fünf voneinander weitestgehend unabhängigen Dimensionen betrachtet. Diese sind Raum, Zeit, Produzent, Konsument und Inhalt. Sie werden nachfolgend detailliert beschrieben.

3.2.1. Räumliche Entkopplung

Räumliche Entkopplung beschreibt, inwiefern einer kommunizierenden Anwendung die Identität und die Anzahl der Kommunikationspartner bekannt ist. Dies bedeutet für das Versenden von Daten von einem Produzenten zu einem oder mehreren Konsumenten, dass

1. dem Produzenten die Identität der Konsumenten bekannt ist oder nicht,
2. dem Produzenten die Anzahl der Konsumenten bekannt ist oder nicht sowie
3. dem Konsumenten die Identität des Produzenten bekannt ist oder nicht.

Diese Eigenschaften können beliebig kombiniert werden, so dass acht Zustände der räumlichen Kopplung bzw. Entkopplung entstehen, die in Tabelle 3.2 dargestellt sind.

Während jede der acht Kombinationen realisierbar ist, sind nicht alle sinnvoll. Zustand (a) repräsentiert die **enge räumliche Kopplung**, bei der Produzent und Konsument die Iden-

Zustand	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
Produzent kennt Identität der Konsumenten	✓	✓	✓	✓	✗	✗	✗	✗
Produzent kennt Anzahl der Konsumenten	✓	✓	✗	✗	✓	✓	✗	✗
Konsumenten kennen Identität der Produzenten	✓	✗	✓	✗	✓	✗	✓	✗

Tabelle 3.2.:
Die acht Zustände der räumliche Entkopplung. Dabei wird für jeden Zustand jede der drei Eigenschaften mit ja (✓) oder nein (✗) belegt.

tität des anderen kennen und dem Produzenten bekannt ist, dass er die Daten an normalerweise genau einen Konsumenten übermittelt. Zustand (h) stellt die **vollständige räumliche Entkopplung** zwischen Produzent und Konsument dar. Der Produzent kennt keine Adressen oder andersartige Identifikationen der Konsumenten, auch nicht deren Anzahl und die Konsumenten kennen nicht die Identität des Produzenten. Eine Zustellung der Daten erfolgt auf Basis derer Inhalte oder zusätzlicher Informationen, wie etwa einer Kanal- oder einer Gruppenbezeichnung.

Wesentliche Zustände teilweiser Entkopplung sind (f) und (g). Bei Zustand (g) kennt lediglich der Konsument die Identität des Produzenten, wie es beim klassischen **Multicast** üblich ist. Der Produzent kennt hier nur eine Multicast-Adresse und weiß auch nicht, welche und wie viele Empfänger sich dahinter verbergen. Zustand (f) repräsentiert eine **anonyme Unicast** Kommunikation wie sie zum Beispiel beim Message Queueing realisiert wird. Produzent und Konsument kommunizieren über einen beiden bekannten Vermittler, in diesem Fall die Message Queue.

Die Zustände (b) und (e) haben praktische Bedeutung bei der anonymen Client-Server-Kommunikation, bei der der Client dem Server nicht bekannt ist. Die **anonyme Anfrage** ist Zustand (b) zuzuordnen und die dazugehörige **anonyme Antwort** Zustand (e). Ein anderes Beispiel für Zustand (e) ist Anycast, bei dem Daten an genau einen Kommunikationspartner aus einer Gruppe nicht einzeln bekannter Geräte versandt werden.

Die Zustände (c) und (d) sind von fragwürdiger Relevanz. Die Ursache dafür ist, dass der Produzent zwar die Identitäten der Konsumenten kennen würde, nicht aber deren Anzahl. Eine Middleware, die Daten an eine mengenmäßig nicht kontrollierbare Auswahl aus einzeln identifizierten Empfängern zustellt, ist im Prinzip vorstellbar aber vermutlich nicht sinnvoll einsetzbar.

3.2.2. Zeitliche Entkopplung

Zeitliche Entkopplung bezieht sich auf die zeitliche Verfügbarkeit von Produzent, Konsument und Übertragungskanal. Drei Zustände zeitlicher Kopplung bzw. Entkopplung werden unterschieden. Sie sind in Abbildung 3.1 dargestellt.

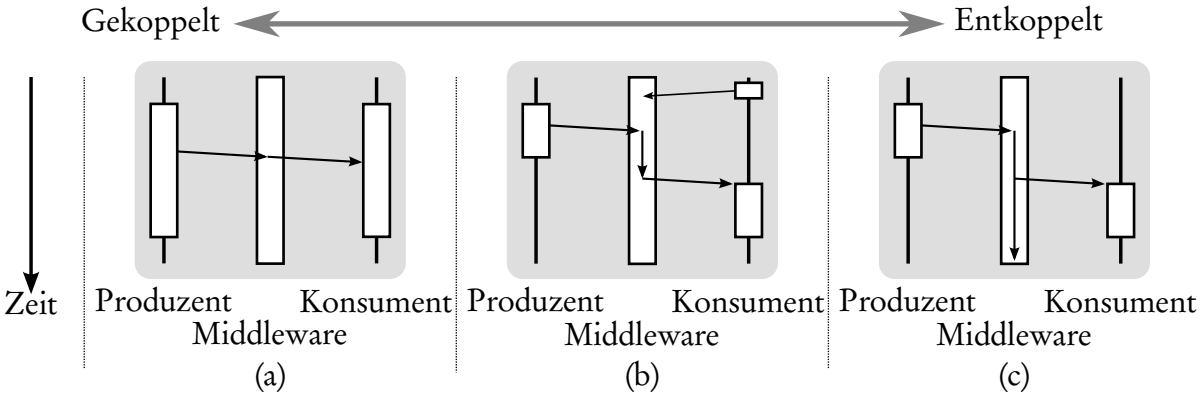


Abbildung 3.1.:
Darstellung der drei Formen zeitlicher Kopplung als Nachrichtenfluss in Sequenzdiagrammen: (a) zeitliche Kopplung, (b) gegenwärtige zeitliche Entkopplung und (c) zukünftige zeitliche Entkopplung.

Zeitliche Kopplung bedeutete, dass eine Datenübertragung zwischen Produzent und Konsument nur dann zustande kommt, wenn Produzent, Konsument und Übertragungskanal gleichzeitig verfügbar sind. Dies wird auch als synchrone Übertragung bezeichnet. Wesentliche Vorteile der zeitlichen Kopplung sind die damit verbundene geringe Übertragungsverzögerung sowie die Möglichkeit der Synchronisation zwischen entfernten Anwendungen.

Das Gegenteil ist die zeitliche Entkopplung. Hier müssen Produzent, Konsument und Übertragungskanal nicht zur gleichen Zeit verfügbar sein. Die Daten für die Übertragung werden in einer Middleware zwischengespeichert, so dass eine Zustellung auch dann noch gewährleistet werden kann, wenn der Produzent schon nicht mehr verfügbar ist. In dem Fall, dass eine Middleware die Identifikation des Konsumenten übernimmt, unterteilt sich die zeitliche Entkopplung in zwei Formen.

Gegenwärtige zeitliche Entkopplung bedeutet, dass Daten von einem Produzenten an alle der Middleware momentan bekannten Konsumenten zugestellt werden unabhängig davon, wann ein Übertragungskanal und die entsprechenden Konsumenten verfügbar sind. Hierfür genügt eine gezielte Zwischenspeicherung der Daten in der Middleware zum Zweck der Zustellung an die ihr momentan bekannten Konsumenten. Nach erfolgreicher Zustellung können die Daten gefahrlos gelöscht werden. Vorteil dieser Form der zeitlichen Entkopplung ist die Unabhängigkeit von der gleichzeitigen Verfügbarkeit von Produzent, Übertragungskanal und Konsumenten wobei gleichzeitig ein definierter Endzustand erreicht wird, in dem die Daten als übertragen gelten und die Middleware von deren Übertragung wieder entlastet ist.

Zukünftige zeitliche Entkopplung bedeutet, dass Daten von einem Produzenten an alle der Middleware in der Zukunft bekannt werdenden Konsumenten übertragen werden.

Hierzu müssen die Daten dauerhaft gespeichert werden. Um eine Auslastung der Speicherkapazität der Middleware vorzubeugen, muss über zusätzliche Anwendungslogik sichergestellt werden, dass die Daten aus der Middleware wieder entfernt werden. Dies kann durch gezieltes Löschen durch den Produzenten oder einen Konsumenten passieren oder durch das Versehen mit einer zeitlich limitierten Gültigkeit beim Versand. Der Vorteil dieser Form der zeitlichen Entkopplung ist, dass die Daten selbst dann übertragen werden, wenn ein Konsument zum Zeitpunkt des Versandes noch nicht bekannt ist.

Die Bezeichnung „bekannt sein“ bezieht sich im Zusammenhang mit der zeitlichen Entkopplung nicht darauf, dass die Middleware weiß, wie der entsprechende Konsument zu erreichen ist, sondern lediglich darauf, dass er existiert. Aus Sicht des Konsumenten stellt sich der Unterschied zwischen gegenwärtiger und zukünftiger zeitlicher Entkopplung wie folgt dar: Bei ersterer meldet sich der Konsument erstmalig an einer Middleware an und kann prinzipiell nur Daten empfangen, die danach von einem Produzenten veröffentlicht werden. Bei der zukünftigen zeitlichen Entkopplung erhält der Konsument nach der ersten Anmeldung alle Daten, die seinem Interesse entsprechen, auch wenn sie in der Vergangenheit veröffentlicht wurden. Wie weit zurück diese Vergangenheit reicht, ist dabei implementationsspezifisch.

3.2.3. Entkopplung des Produzenten

Eine weitere Form der Kopplung ist die Art, wie der Produzent an eine Middleware angebunden ist. Hier spielt allerdings die genaue Schnittstelle keine signifikante Rolle. Stellt die Middleware lediglich einen blockierenden Funktionsaufruf zur Verfügung, so kann dieser durch die Implementierung eines einfachen Threads problemlos in einen nicht-blockierenden Aufruf umgewandelt werden. Die Abbildung eines nicht-blockierenden Funktionsaufrufs auf einen blockierenden ist ebenso möglich.

Von Bedeutung für die Anwendung ist viel mehr die Art der Zusicherung, ob Daten übermittelt werden oder nicht. Abbildung 3.2 stellt die vier Möglichkeiten dar. Bei der **Kopplung** des Produzenten an die Middleware gibt diese keine Zusicherung bezüglich der erfolgreichen Übertragung von Daten. Die Anwendung muss also in jedem Fall auf die Antwort der Middleware warten um auf eine erfolgreiche oder fehlerhafte Übertragung reagieren zu können.

Die **partielle Kopplung** des Produzenten an die Middleware kann entweder **mit positiver oder negativer Zusicherung** erfolgen. Die Middleware sichert in diesem Fall zu, dass Daten erfolgreich bzw. nicht übertragen werden. Die Anwendung wird nur im jeweiligen Fehlerfall benachrichtigt, um entsprechend reagieren zu können.

Eine **vollständige Entkopplung** des Produzenten von der Middleware wird häufig als Fire and Forget (JOSUTTIS, 2007, S. 297) bezeichnet. Die zu übertragenden Daten werden der Middleware übergeben und diese gibt keinerlei Rückmeldung. Dabei muss die Middleware klar definieren, welche Art der Zusicherung sie gibt. Möglichkeiten sind best effort, also die Übertragung soweit dies möglich ist, oder eine definiert sichere Übertragung. Damit

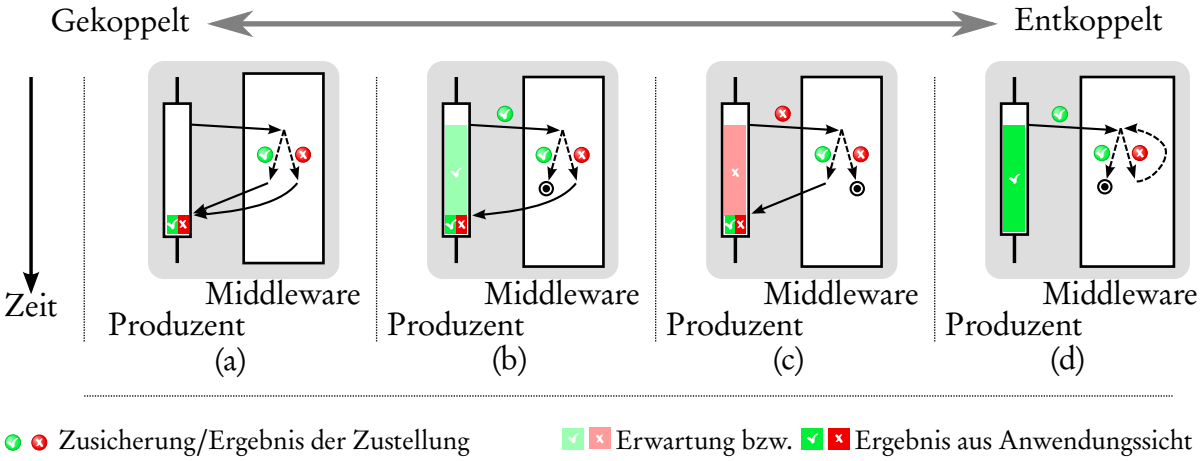


Abbildung 3.2.:
Die vier Zustände der Kopplung eines Produzenten an eine Middleware als Mischung aus Sequenz- und Zustandsdiagramm: (a) enge Kopplung, (b/c) partielle Kopplung mit positiver bzw. negativer Zusicherung und (d) vollständige Entkopplung.

ist gemeint, dass die Middleware die Rahmenbedingungen klar definiert unter denen sie die Übertragung der Daten sicherstellt.

Da in verwandten Arbeiten die Kopplung des Produzenten an die Middleware über die Frage, ob ein blockierender (b) oder nicht-blockierender ($\not{b}$) Funktionsaufruf angeboten wird, entschieden wird, muss an dieser Stelle noch einmal auf den bedeutenden Unterschied eingegangen werden. Die Art des Funktionsaufrufs ist allein eine Frage der lokalen Implementierung der Middleware. Und sie kann durch die lokale Implementierung der Anwendung (z.Bsp. ein weiterer Thread) leicht verändert ($b \rightarrow \not{b}$ / $\not{b} \rightarrow b$) werden.

An welchem Ort die Middleware allerdings die Zustellung der Daten sicherstellt, ist eine Frage der Implementierung bzw. des Konzeptes der Middleware. So ist es zum Beispiel nicht möglich, eine vollständig entkoppelte Middleware lokal auf eine gekoppelte abzubilden, weil die Information, ob die Daten letztlich wirklich zugestellt wurden, lokal zu keinem Zeitpunkt vorliegt. Andersherum ist es nicht effizient, eine gekoppelte Middleware in einer Umgebung mit instabilem Rückkanal einzusetzen, indem lokal durch wiederholtes Versuchen die Übertragung so lange wiederholt wird, bis sie erfolgreich bestätigt wurde. Abbildung 3.3 veranschaulicht diesen Sachverhalt.

3.2.4. Entkopplung des Konsumenten

Ähnlich wie bei der Entkopplung des Produzenten spielt bei der Anbindung des Konsumenten an die Middleware die Art des Funktionsaufrufs, ob blockierend oder nicht, keine Rolle, da beides leicht aufeinander abbildbar ist. Vielmehr wird der Grad der Kopplung hier

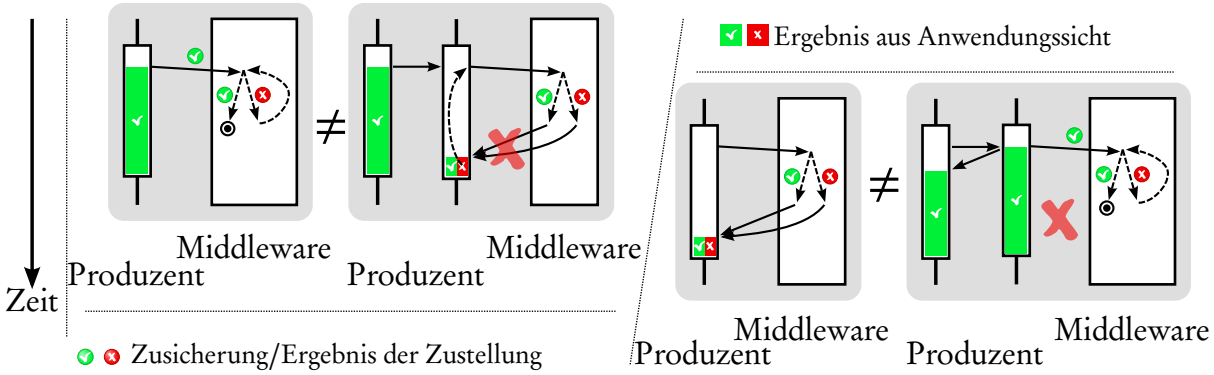


Abbildung 3.3.:
Kopplung und Entkopplung des Produzenten in Bezug auf die Middleware lassen sich durch Modifikation des Produzent nicht adäquat aufeinander abbilden. Die Abbildung als Mischung aus Sequenz- und Zustandsdiagramm zeigt eine Aufteilung des Produzenten in 2 Threads als Versuch einer Abbildung. In beiden Fällen ist der Unterschied der von der Middleware vorgesehene bzw. nicht vorgesehene Rückkanal (rote Kreuze).

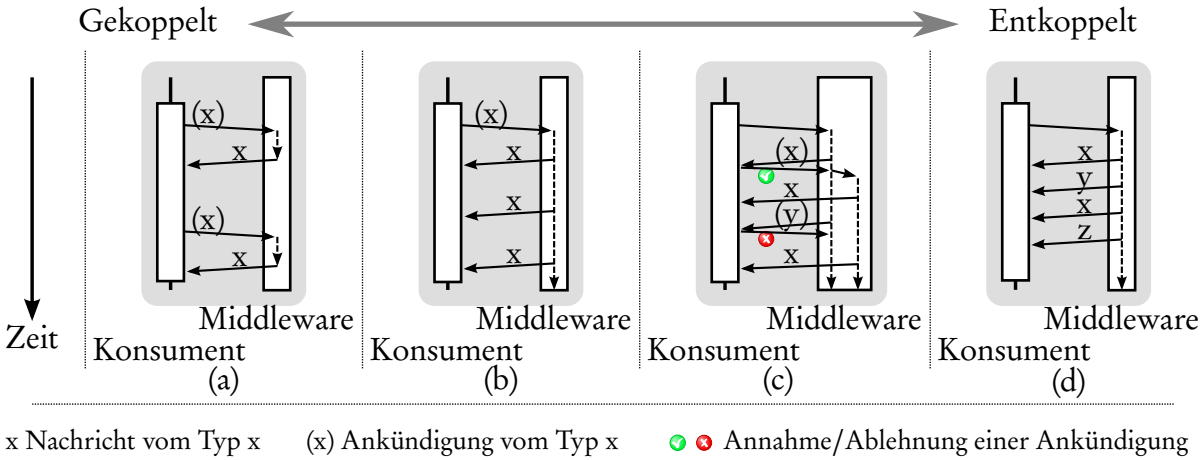


Abbildung 3.4.:
Die vier Zustände der Entkopplung des Konsumenten von der Middleware als Sequenzdiagramme: Anfrage-Antwort (a), kontinuierliche Anfrage (b), Registrierung mit Rückfrage (c) sowie einfache Registrierung (d).

dadurch bestimmt, wie feingranular Daten bei der Middleware angefordert werden müssen. Abbildung 3.4 gibt einen Überblick.

Die stärkste Form der Kopplung ist **Anfrage-Antwort**. Es wird gezielt nach einer oder mehreren Informationen gefragt und die entsprechenden Daten werden gegebenenfalls sobald sie verfügbar sind geliefert. Eine Anfrage liefert die zur Zeit der Anfrage auf die Anfrage

passenden Informationen. Für danach veröffentlichte Informationen muss der Konsument erneut anfragen.

Eine etwas schwächere Form der Kopplung ist die **kontinuierliche Anfrage**. Der Konsument registriert sich für einen Informationstyp und die entsprechenden Informationen werden, wann immer sie verfügbar werden, so lange an den Konsument geliefert, bis dieser die Registrierung beendet. Der Konsument beschreibt die benötigten Informationen und erhält dann genau alle diese Informationen.

Wiederum etwas schwächer an die Middleware gekoppelt ist der Konsument bei der **Registrierung mit Rückfrage**. Hier registriert er sich bei der Middleware mit oder auch ohne Einschränkung seines Interesses. Er wird dann über die Verfügbarkeit von passenden Informationen benachrichtigt und kann anschließend auswählen, ob diese Informationen geliefert werden sollen, oder nicht.

Die schwächste Form der Kopplung, hier als **einfache Registrierung** bezeichnet, ist die Registrierung bei der Middleware ohne weitere Einschränkungen und der anschließende Empfang aller Informationen.

Wie Daten bei der Anfrage beschrieben bzw. als Informationstypen durch die Middleware angekündigt werden, obliegt der Definition der Middleware bzw. deren Schnittstellen. Möglichkeiten sind zum Beispiel die Adresse des Produzenten, Gruppenbezeichner, Themen, Strukturinformationen oder inhaltliche Beschreibungen der Daten.

3.2.5. Inhaltliche Entkopplung

Inhaltliche Entkopplung bezeichnet die Entkopplung zwischen Produzent und Konsument auf Ebene des Nachrichteninhalts. Dabei kann zwischen drei Zuständen unterschieden werden. Diese sind in Abbildung 3.5 dargestellt. Beispiele für die drei Zustände werden in Tabelle 3.3 angegeben.

Inhaltliche Kopplung beschreibt, dass keine Manipulation von Daten durch die Middleware durchgeführt wird. Der Konsument erhält die Daten genau so, wie sie der Produzent versandt hat.

Teilweise inhaltliche Entkopplung beschreibt, dass entweder der Produzent zusammen mit den Daten eine Verarbeitungsvorschrift veröffentlicht, anhand derer diese durch die Middleware verarbeitet werden, damit sie von verfügbaren Konsumenten verstanden werden können oder, dass der Konsument diese Verarbeitungsvorschrift zusammen mit seiner Anfrage an die Middleware bereitstellt, so dass die Daten vor der Zustellung an ihn von der Middleware entsprechend verarbeitet werden.

Vollständige inhaltliche Entkopplung ist dann gegeben, wenn der Produzent eine Information x veröffentlicht und der Konsument eine Information y erhält und die Middleware eigenständig dafür Sorge trägt, dass die Daten auf dem Weg vom Produzent zum Konsument entsprechend umgewandelt werden. Dies kann entweder in der Middleware selbst passieren. Dazu ist es nötig, dass die Middleware den Inhalt der Daten versteht und demzufolge in

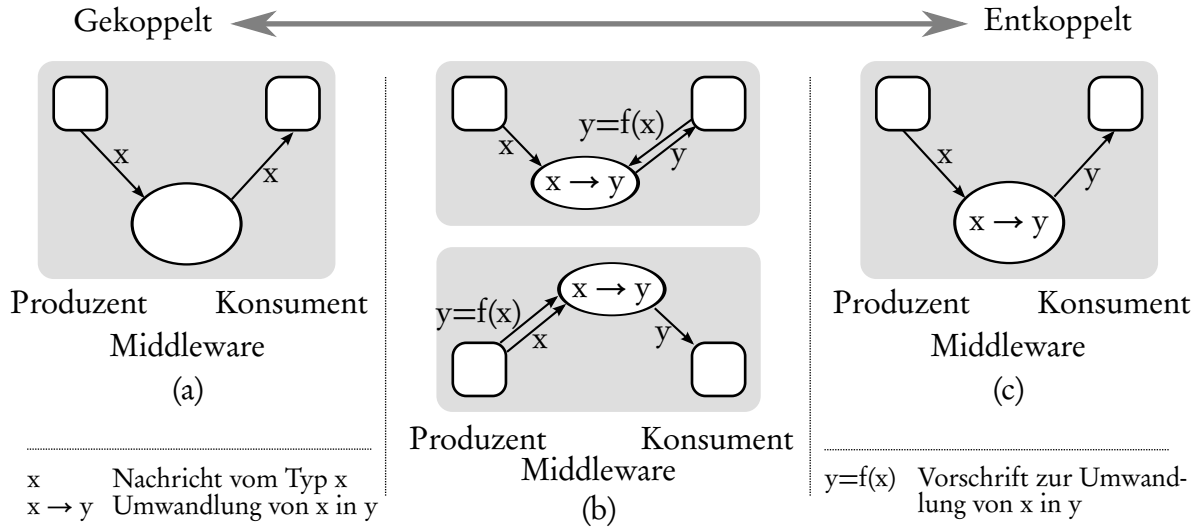


Abbildung 3.5.:
Die drei Zustände inhaltlicher Entkopplung: Kopplung (a), teilweise Entkopplung (b), vollständige Entkopplung (c). Dabei repräsentieren x und y Nachrichten vom jeweiligen Typ. „y=f(x)“ stellt eine Verarbeitungsvorschrift für die Umwandlung dar und „x → y“ repräsentiert die Durchführung der Umwandlung einer Nachricht vom Typ x in den Typ y.

der Lage ist, ihn zu manipulieren. Die zweite Möglichkeit ist die Einbindung externer Anwendungen, die eine Umwandlung der Daten gewährleisten. Hierzu ist es nicht zwangsläufig nötig, dass die Middleware den Inhalt der Daten versteht. Es muss lediglich ein Mechanismus zur Auswahl einer im Einzelfall geeigneten externen Anwendung existieren.

Veröffentlichte Information (Produzent)	Anfrage (Konsument)	Grad inhaltlicher Kopplung
Temperatur in C	Temperatur in C	gekoppelt
Temperatur in C	Temperatur in K = f(C)	teilweise entkoppelt
Temperatur in C	Temperatur in K	vollständig entkoppelt
Temperatur in C	Durchschnittstemperatur in C über 10 min liegt über 400	teilweise entkoppelt
Temperatur in C	Feueralarm	vollständig entkoppelt

Tabelle 3.3.: Beispiele für die verschiedenen Zustände Inhaltlicher Entkopplung.

Verfahren	Abstraktion	Raum								Zeit			Produzent				Konsument				Inhalt		
		a	b	c	d	e	f	g	h	a	b	c	a	b	c	d	a	b	c	d	a	b	c
verbindungsloser Unicast	Paket	✓	O							✓						✓				✓	✓		
Anycast	Paket					✓	O			✓						✓				✓	✓		
Multicast	Paket							✓	O	✓						✓				✓	✓		
verbindungsorientierter Unicast	Verbindung	✓	O			O	O			✓				✓						✓	✓		
RPC	Funktion	✓	✓			✓	✓			✓			✓			*	*	✓			✓	O	O
RMI / Object-Broker	Methode	✓	✓			✓	✓			✓			✓			*	*	✓			✓	O	O
Message Queueing	Nachricht						✓		O			✓	✓	✓	✓	✓				✓	✓	O	O
Publish/Subscribe	Nachricht								✓	✓			✓	✓	✓	✓		✓	O	✓	✓	✓	✓
Tuple Space	Nachricht						O		✓			✓	✓	✓	✓	✓	✓	O					✓

Tabelle 3.4.:
Charakterisierung verschiedener Kommunikationsparadigmen nach den unterstützten Graden an Entkopplung. Dabei wird unterschieden zwischen ist im Kommunikationsparadigma vorgesehen (✓) und kann ohne das Paradigma zu verletzen realisiert werden (O). Felder ohne Markierung bedeuten, dass eine Form der Entkopplung nicht implementiert werden kann, ohne das Paradigma zu verletzen. Abweichende Entkopplungseigenschaften für einen eventuellen Rückkanal sind durch (*) gekennzeichnet.
Dabei bedeuten Raum: a - enge Kopplung, b - anonyme Anfrage, e - anonyme Antwort, f - anonymer Unicast, g - Multicast, h - vollständige Entkopplung; Zeit: a - Kopplung, b - gegenwärtige Entkopplung, c - zukünftige Entkopplung; Produzent: a - Kopplung, b/c partielle Kopplung mit positiver/negativer Zusicherung, d - Entkopplung; Konsument: a - Anfrage-Antwort, b - kontinuierliche Anfrage, c - Registrierung mit Rückfrage, d - einfache Registrierung; Inhalt: a - Kopplung, b - teilweise Entkopplung, c - vollständige Entkopplung.

3.3. Klassifizierung bestehender Kommunikationsparadigmen

Nachfolgend werden existierende Kommunikationsparadigmen auf ihre Kopplungseigenschaften hin untersucht. Da sich basierend auf der Art der Abstraktion der Datenübertragung verschiedene Besonderheiten ergeben, wird diese zur Gruppierung der einzelnen Paradigmen in den folgenden Abschnitten benutzt. Tabelle 3.4 gibt einen Überblick über die Ergebnisse der Untersuchung.

3.3.1. Paketbasierte Kommunikation

Der Grad der Abstraktion bei paketbasierten Verfahren ist sehr gering. Es werden Datenpakete vom Produzent an einen oder mehrere Konsumenten versandt. Eine Zwischenspeicherung zum Zweck der zeitlichen Entkopplung findet nicht statt. Sind Produzent, Übertragungskanal und Konsument nicht gleichzeitig verfügbar, so wird das Paket nicht übertragen. Eine Benachrichtigung des Produzenten findet nicht statt. Somit ist der Produzent vollständig von der Middleware entkoppelt. Aufgrund der fehlenden zeitlichen Entkopplung erfolgt die Übertragung nach dem best-effort-Prinzip.

Ein Konsument registriert sich ohne zusätzliche Informationen und empfängt alle Pakete, die an ihn versandt wurden. Eine vorherige Einschränkung ist nicht möglich. Eine inhaltliche Entkopplung ist nicht möglich. Die Daten kommen im gleichen Zustand an, wie sie versandt wurde.

Die räumliche Entkopplung hängt vom gewählten Verfahren ab. Der verbindungslose Unicast ist stark räumlich gekoppelt. Beim Anycast und Multicast kann der Produzent die Konsumenten nicht eindeutig identifizieren. Bei ersterem wird eine Identifikation verwandt, hinter der sich beliebig viele Konsumenten verbergen können. Lediglich die Übertragung an genau einen wird durchzuführen versucht. Bei zweiterem wird eine spezielle Multicast-Adresse verwendet, unter der sich eine dem Produzenten unbekannte Menge an Konsumenten registrieren und die versandten Pakete empfangen können. Ein dem Konsument nicht bekannter Produzent ist in allen Fällen möglich, aber im Prinzip nicht vorgesehen. Er ist vielmehr ein Nebeneffekt von Verfahren wie Network Address Translation.

Ein Beispiele für paketbasierte Verfahren ist das UDP-Protokoll (POSTEL, 1980). Je nach verwendeter Zieladresse werden verbindungsloser Unicast, Anycast und Multicast unterstützt.

3.3.2. Verbindungsorientierte Kommunikation

Verbindungsorientierte Verfahren abstrahieren von der paketbasierten Datenübertragung, indem sie einer Anwendung eine stabile Verbindung zu einer anderen Anwendung bereitstellen, über die Daten versandt und empfangen werden können. Da eine solche Verbindung bidirektional ist, sind beide Anwendungen sowohl Produzent als auch Konsument. Die Kopplungseigenschaften sind nicht richtungsabhängig.

Aufgrund der Abstraktion als Verbindung, die momentan besteht, geht mit verbindungsorientierten Verfahren immer eine enge zeitliche Kopplung einher. Der Produzent sendet Daten an den Konsument und es wird sichergestellt, dass diese Daten sicher ankommen, solange die Verbindung besteht. Ein Abbruch der Verbindung wird dem Produzent mitgeteilt. Er ist also mit positiver Zusicherung partiell an die Middleware gekoppelt. Auf Seite des Konsumenten werden alle Daten empfangen, die vom Produzent über die Verbindung ver-

sandt werden. Eine Einschränkung ist nicht möglich. Eine Inhaltliche Entkopplung ist nicht vorgesehen.

Normalerweise stellen verbindungsorientierte Verfahren eine enge räumliche Kopplung zwischen Produzent und Konsument sicher. Es gibt spezielle Ausnahmen. Analog zu den verbindungslosen Verfahren kann die Identität des Produzenten zum Beispiel durch Network Address Translation vor dem Konsument verborgen werden. Außerdem ist in sehr stabilen Netzen oder mit Erweiterungen bezüglich der Verbindungsabsicherung (PARTRIDGE, MENDEZ und MILLIKEN, 1993) auch ein Anycast möglich, so dass die Identität des Konsumenten vor dem Produzent verborgen bleibt. Einer Kombination beider Ausnahmen steht ebenfalls nichts im Weg.

Klassischer Vertreter der verbindungsorientierten Verfahren ist das TCP-Protokoll (POSTEL, 1981). Es gibt aber auch leistungsfähigere Protokolle, wie zum Beispiel SCTP (STEWART, 2007), dass die Übertragung mehrerer paralleler Datenströme zulässt sowie mehrere Identitäten pro Anwendung handhaben kann. Die Kopplungseigenschaften unterscheiden sich aber nicht.

3.3.3. Abstraktion durch Funktionen/Methoden

Ein weiterer Abstraktionsschritt ist das Verbergen der Kommunikation hinter einer Prozedur oder Methode, die von einer Anwendung aufgerufen wird. Das entsprechende Konzept ist der Remote Procedure Call (RPC) (TANENBAUM und STEEN, 2007, S.125ff).

Während bei paketbasierter und verbindungsorientierter Kommunikation die Anwendung alle Daten selbst auf die Übertragung vorbereiten musste, werden hier Serialisierung, Kodierung und Übertragung der Parameter und Rückgabewerte von der Middleware übernommen. Auch hier handelt es sich um eine bidirektionale Datenübertragung. Die Parameter werden von der aufrufenden zur ausführenden Anwendung übertragen und anschließend die Rückgabewerte von der ausführenden zur aufrufenden Anwendung. Die Unterschiede in Bezug auf die Kopplungseigenschaften zwischen beiden Übertragungsrichtungen sind minimal. Eingangs wird angenommen, die aufrufende Anwendung ist der Produzent und die ausführende Anwendung ist der Konsument.

Zeitlich sind Produzent und Konsument eng aneinander gekoppelt. Zwischen Funktionsaufruf und Bereitstehen der Rückgabewerte muss sowohl der Konsument, als auch der Übertragungskanal verfügbar sein, um die Funktion auszuführen bzw. die Parameter und Rückgabewerte zu übertragen.

Eine räumliche Entkopplung ist in Bezug auf die Identitäten von Produzent und Konsument gegenüber dem Kommunikationspartner möglich. Hier kann ohne das Paradigma zu verletzen die Vermittlung von Produzent und Konsument, sowie die Übertragung der Daten zum Beispiel über einen zentralen Vermittler abgewickelt werden.

Eine inhaltliche Entkopplung zwischen Produzent und Konsument ist im Prinzip nicht vorgesehen. Allerdings können eine Konvertierung von Datentypen oder eine Anpassung

der Daten an unterschiedliche Frameworks oder Programmiersprachen durchaus notwendig sein und können zum Beispiel ebenfalls über einen zentralen Vermittler realisiert werden.

Der Produzent ist eng an die Middleware gekoppelt. Die Funktion oder Methode wird aufgerufen und ein Erfolg oder Misserfolg der transparenten Datenübertragung ist erst bekannt, wenn ein Rückgabewert oder Fehler zur Verfügung steht. Selbst bei asynchronen RPC-Systemen (TANENBAUM und STEEN, 2007, S.134ff) ist eine erfolgreiche Übertragung für den Produzenten erst ersichtlich, wenn die Annahme der Anfrage vom Server bestätigt wird. Der Konsument registriert sich bei der Middleware und bekommt im Anschluss klar definierte Daten in Form von Funktions- oder Methodenparametern zugesandt. Dieses Verhalten entspricht der kontinuierlichen Anfrage.

Betrachtet man die Übertragung des Rückgabewertes mit der ausführenden Anwendung als Produzent und der aufrufenden Anwendung als Konsument, so ergibt sich ein anderes Bild. Da der Produzent nach der Ausführung der Funktion oder Methode die Rückgabewerte einfach an die Middleware übergibt, muss die erfolgreiche Übertragung von der Middleware sichergestellt werden. Der Produzent ist also in diesem Fall vollständig entkoppelt. Für den Konsument erfolgt der Empfang der Rückgabewerte als Antwort auf seinen ursprünglichen Funktions- oder Methodenaufruf. Die Entkopplungseigenschaft für den Konsument ist also Anfrage-Antwort.

Beispiele für RPC-basierende Kommunikationssysteme sind Sun-RPC (SUN MICROSYSTEMS, 1988), DCE RPC (THE OPEN GROUP), DCOM (MICROSOFT CORPORATION), Java-RMI (SUN MICROSYSTEMS, INC.), CORBA (OBJECT MANAGEMENT GROUP) und XML-RPC (DAVE WINER).

3.3.4. Nachrichtenbasierte Kommunikation

Während bei den zuvor beschriebenen Arten der Abstraktion das Ziel im Vordergrund lag, die Tatsache, dass Kommunikation mit einer anderen Anwendung stattfindet, schrittweise stärker vor den Anwendungen zu verbergen, werden bei der nachrichtenbasierten Kommunikation die Nebeneffekte von Kommunikation vor den Anwendungen verborgen. Hauptaugenmerk liegt auf dem Verbergen der räumlichen und zeitlichen Kopplung von Nachrichtenproduzent und -konsument.

Die zu übertragenden Informationen werden als eine Einheit, die Nachricht, zusammengefasst und mit Hilfe einer Middleware versandt. Die Identitäten der Konsumenten sind dem Produzent unbekannt genau wie die Identität des Produzenten vor den Konsumenten verborgen bleibt. Eine weitere zentrale Eigenschaft ist die Unidirektionalität. Eine Nachricht wird von einem Produzent an einen oder mehrere Konsumenten übertragen, ohne dass diese direkt darauf antworten können.

Zur Diskussion der konkreten Eigenschaften werden die drei wesentlichen Konzepte zur Bereitstellung nachrichtenbasierter Kommunikation einzeln untersucht.

Publish/Subscribe

Publish/Subscribe als Kommunikationsparadigma sowie entsprechende Implementierungsmöglichkeiten wurden in Abschnitt 2.1 ausführlich beschrieben. Publish/Subscribe gewährleistet gegenwärtige zeitliche Entkopplung, wobei eine veröffentlichte Nachricht an alle Konsumenten versandt wird, die im Moment der Veröffentlichung bei der Middleware registriert sind, unabhängig davon, ob sie gerade verfügbar sind oder nicht.

In Publish/Subscribe Systemen sind die Identitäten und die Anzahl der Konsumenten für den Produzent sowie die Identität des Produzenten für die Konsumenten unbekannt. Die Anbindung des Produzenten ist implementierungsspezifisch. Für den Konsument ist der Normalfall, dass er sich mit einem Filter, seinem Interesse, an der Middleware registriert und anschließend entsprechende Nachrichten empfängt. Es ist auch denkbar, dass auf diese Weise alle Nachrichten empfangen werden.

In Publish/Subscribe-Systemen, die auf inhaltsbasiertem Routing mit Ankündigungen basieren, ist es sogar denkbar, dass die Middleware der Anwendung einen Überblick über verfügbare Nachrichten geben und somit den Entkopplungsgrad Registrierung mit Rückfrage gewährleisten kann.

Publish/Subscribe-Systeme mit inhaltlicher Kopplung und teilweise inhaltlicher Entkopplung sind weit verbreitet. Auch vollständige inhaltliche Entkopplung ist möglich, ohne das Publish/Subscribe-Paradigma zu verletzen. Zahlreiche Beispiele für Publish/Subscribe-Systeme wurde bereits in Abschnitt 2.1 genannt und weitere Beispiele finden sich im Abschnitt 4.1.1.

Message Queueing

Das zentrale Konzept beim Message Queueing ist die Message Queue, eine Warteschlange für Nachrichten. Produzenten können Nachrichten an eine Warteschlange senden und Konsumenten diese aus einer Warteschlange abholen. Wenn eine neue Nachricht an eine Warteschlange angehängt wird, so ist noch nicht bekannt, welcher Konsument diese Nachricht wann abholen wird. Dieses Verhalten entspricht zukünftiger zeitlicher Entkopplung. Es besteht die Möglichkeit, dass die Nachricht niemals abgeholt wird. Um dem Volllaufen einer Warteschlange vorzubeugen, erlauben viele Message Queueing Systeme, Nachrichten mit einer maximalen Gültigkeit zu versehen.

Normalerweise wird eine Nachricht von genau einem Konsument aus der Warteschlange abgeholt, dem Produzent ist also die 1:1-Semantik bekannt. Es lässt sich aber über eine peek-Operation, bei der eine Nachricht am Kopf der Warteschlange angesehen wird, ohne sie abzuholen, auch eine 1:n-Semantik realisieren, bei der dem Produzent nicht mehr bekannt ist, wie viele Konsumenten ihre Nachricht empfangen.

Die Art der Kopplung des Produzenten an die Middleware ist prinzipbedingt nicht eingeschränkt, hängt also allein von deren Implementierung ab. Der Konsument hat keine Möglichkeit zu bestimmen, welche Art von Nachrichten er empfängt. Er meldet sich bei einer

Warteschlange an und holt dort Nachrichten ab. Erst lokal kann deren Inhalt ausgewertet und entsprechend reagiert werden. Eine inhaltliche Entkopplung von Produzent und Konsument ist nicht vorgesehen aber möglich, ohne das Paradigma zu verletzen.

Ein Beispiel für ein reines Message Queueing System ist MSMQ (MICROSOFT CORPORATION). Aufgrund vieler Ähnlichkeiten insbesondere zwischen themenbasiertem Publish/Subscribe und Message Queueing, gibt es zahlreiche Implementierungen, die Message Queueing und themenbasiertes Publish/Subscribe verbinden, wie zum Beispiel IBM WebSphere MQ (IBM) oder Java Message Service (SUN MICROSYSTEMS, INC.), welches sowohl aus einer eigenen Referenzimplementierung, als auch einer Schnittstelle zur Anbindung anderer Message Queueing und Publish/Subscribe Systeme an die Java-Plattform besteht.

Tuple Space

Ein Tuple Space ist konzeptionell ein verteilter assoziativer Speicher (GELERNTER, 1985). Dabei ist die Speicherung der Nachrichten im Tuple Space nicht Hauptaugenmerk sondern viel mehr Mittel zum Zweck der zukünftigen zeitlich entkoppelten Übertragung.

Nachrichten werden im Tuple Space durch Tupel repräsentiert. Ein Tupel besteht aus typisierten Atomen. Der Tuple Space definiert drei Operationen, write, read und take. Write benötigt die Tuple-Struktur und dessen Inhalt und schreibt das Tupel in den Tuple Space. Read liefert eine Kopie eines Tupels aus dem Tuple Space, dass anhand seiner Struktur und eventuell Teilen seines Inhalts gefunden wird. Take ist eine atomare Operation aus Read und dem Entfernen des gelesenen Tupels aus dem Tuple Space.

Aufgrund der beliebigen Kombination aus read und take ist die Anzahl der Konsumenten für einen Produzent eigentlich nicht bekannt. Da durch die take-Operation das Tupel aber aus dem Tupel Space entfernt und somit nicht an weitere Konsumenten übermittelt werden kann, lässt sich anwendungsseitig mit einer Tuple Space Middleware auch eine 1:x-Übertragung mit durch den Produzent definiertem x realisieren. Voraussetzung dafür ist allerdings, dass sich die Konsumenten an die notwendige Übereinkunft halten.

Die Kopplung eines Produzenten an den Tuple Space ist wiederum eine Frage der Implementierung und konzeptuell nicht eingeschränkt. Der Konsument entnimmt Tupel aus dem Tuple Space nach dem Prinzip Anfrage-Antwort. Eine Erweiterung von Tuple Spaces hin zur Registrierung von Konsumenten für einen beschriebenen Typ von Tupel und der Benachrichtigung bzw. Übermittlung entsprechender Tupel ist möglich ohne das Kommunikationsparadigma zu verletzen.

Aufgrund der vollständigen Abbildung der Kommunikation auf das Konzept eines zentralen Nachrichtenspeichers ist eine inhaltliche Kopplung zwischen Produzent und Konsument nicht möglich. Es gibt seitens der Middleware keine Möglichkeit sicherzustellen, dass ein Tupel wirklich von einem bestimmten Produzent stammt oder nicht schon mehrfach durch Anwendungen aus dem Tuple Space entfernt, verarbeitet, und wieder in den Tuple Space eingefügt wurde.

Beispiele für Implementierungen des Tuple Space Paradigmas sind TSpaces (IBM) und JavaSpaces (FREEMAN, HUPFER und ARNOLD, 1999).

3.4. Anforderungen entsprechend der Problemstellung

In 1.1 wurden verschiedene Anforderungen an eine gute Middleware aufgeführt. Für die Entkopplungseigenschaften der Middleware sind insbesondere diejenigen Anforderungen wichtig, die sich auf die Interaktion zwischen Anwendungen sowie zwischen Anwendungen und Middleware beziehen. Diese sind Zuverlässigkeit, Transparenz und Interoperabilität sowie die einfache Integrierbarkeit und der Austausch im laufenden Betrieb.

Um Transparenz in einer unbekannten und veränderlichen Umgebung gewährleisten zu können ist es sinnvoll, Anwendungen so gut wie möglich räumlich voneinander zu entkoppeln. Wenn von vornherein angenommen wird, dass die Anzahl und Identität der Kommunikationspartner einer Anwendung unbekannt sind, dann lässt sich eine Veränderung dieser beiden Parameter am effektivsten vor der Anwendung verbergen.

Zur Gewährleistung der Zuverlässigkeit der Kommunikation zwischen Anwendungen ist eine gegenwärtige zeitliche Entkopplung unbedingt notwendig, da in einer veränderlichen heterogenen Umgebung nicht gewährleistet ist, dass beide Kommunikationspartner und der Übertragungskanal gleichzeitig verfügbar sind.

Eine maximale Entkopplung des Produzenten von der Middleware ist ebenfalls wünschenswert. Sie entbindet den Produzenten von der Notwendigkeit sicherzustellen, dass seine Veröffentlichungen erfolgreich bei der Middleware und den Kommunikationspartnern ankommen.

Zur Gewährleistung von Interoperabilität zwischen Anwendungen ist eine maximale inhaltliche Entkopplung sinnvoll. Nur wenn die Middleware in der Lage ist, notwendige Datenverarbeitung automatisch durchzuführen, kann die Interoperabilität zwischen Anwendungen mit unterschiedlichen Kommunikationsanforderungen ohne deren Anpassung gewährleistet werden.

Aufgrund der vorrangigen Kommunikation von Informationen, die einer stetigen Aktualisierung durch neuere Informationen unterliegen, ist für den Konsument die Kopplungsform kontinuierliche Anfrage sinnvoll. Stellt der Konsument eine Schnittstelle zum Benutzer zur Verfügung, kann es von Bedeutung sein, dem Nutzer eine Auswahl im System verfügbarer Informationen zu liefern. Dies ist sehr effizient mit der Kopplungsform Registrierung mit Rückfrage möglich, da der Konsument hier über alle im System verfügbaren Informationstypen informiert wird, ohne selbst alle veröffentlichten Informationen auch empfangen zu müssen.

Zum Zweck der einfachen Integrierbarkeit neuer Anwendungen sowie dem Austausch von Anwendungen im laufenden Betrieb kann es sinnvoll sein, die zeitliche Entkopplung dahingehend zu erweitern, dass Informationen nicht nur an alle bekannten Konsumenten

Verfahren	Abstraktion	Raum								Zeit			Produzent				Konsument				Inhalt		
		a	b	c	d	e	f	g	h	a	b	c	a	b	c	d	a	b	c	d	a	b	c
Publish/Subscribe	Nachricht							✓		✓			✓	✓	✓	✓		✓	O	✓	✓	✓	✓
Tuple Space	Nachricht						O	✓			✓		✓	✓	✓	✓	✓	O					✓
Anforderungen								✓		✓	?					✓		✓	✓				✓

Tabelle 3.5.:
Die abgeleiteten Kommunikationsanforderungen spontaner intelligenter Umgebungen im Vergleich mit den beiden geeignetsten Kommunikationsparadigmen. Zusätzlich zur Notation der Übersichtstabelle 3.4 wird durch (?) dargestellt, dass eine eingeschränkte Unterstützung zukünftiger zeitliche Entkopplung wünschenswert ist.
Dabei bedeuten Raum: f - anonymer Unicast, h - vollständige Entkopplung; Zeit: b - gegenwärtige Entkopplung, c - zukünftige Entkopplung; Produzent: a - Kopplung, b/c partielle Kopplung mit positiver/negativer Zusicherung, d - Entkopplung; Konsument: a - Anfrage-Antwort, b - kontinuierliche Anfrage, c - Registrierung mit Rückfrage, d - einfache Registrierung; Inhalt: a - Kopplung, b - teilweise Entkopplung, c - vollständige Entkopplung.

vermittelt werden, sondern auch an alle zukünftig bekannt werdenden. Im Unterschied zur zukünftigen zeitlichen Entkopplung gilt dieses Verhalten nicht für alle Informationen, sondern nur für die jeweils aktuellsten eines Informationstyps, also zum Beispiel die zuletzt veröffentlichten Nachrichten in einem Nachrichtenstrom.

In Tabelle 3.5 werden die zuvor beschriebenen Anforderungen den Fähigkeiten der Kommunikationsparadigmen Publish/Subscribe und Tuple Space gegenübergestellt. Man erkennt deutlich, dass Publish/Subscribe das ähnlichste Kommunikationsparadigma ist. Lediglich eine Zwischenspeicherung aktueller Veröffentlichungen für zukünftig interessierte Konsumenten ist eine sinnvolle Erweiterung, um die Integrierbarkeit von Anwendungen und den unterbrechungsfreien Betrieb bei deren Austausch zu optimieren.

3.5. Zusammenfassung

Die Fähigkeit zur Entkopplung kommunizierender Anwendungen voneinander und von der sie umgebenden Netzwerktopologie ist insbesondere in heterogenen, veränderlichen Netzwerktopologien eine wichtige Eigenschaft einer Middleware. Um die durch die Problemstellung gegebenen Anforderungen mit den Möglichkeiten verschiedener Kommunikationsparadigmen zu vergleichen ist eine Taxonomie der Kopplung bzw. Entkopplung sinnvoll.

Da bestehende Taxonomien hier nur unzureichend und mit zum Teil widersprüchlichen Ergebnissen differenzieren, wurde eine neue Taxonomie vorgestellt. Sie unterscheidet zwischen fünf Dimensionen der Entkopplung: Raum, Zeit, zwischen Produzent und Middleware, zwischen Konsument und Middleware sowie inhaltlich.

Anhand dieser Taxonomie lassen sich bestehende Kommunikationsparadigmen anhand ihrer Kopplungs- und Entkopplungseigenschaften einordnen und mit den Anforderungen der Problemstellung vergleichen. Die Problemstellung erfordert eine Middleware, die folgende Formen der Entkopplung gewährleisten kann:

- vollständige räumliche Entkopplung
- gegenwärtige zeitliche Entkopplung für alle übertragenen Nachrichten und zukünftige zeitliche Entkopplung für die jeweils aktuellste Nachricht eines Nachrichtenstromes
- vollständige Entkopplung des Produzenten von der Middleware
- kontinuierliche Anfragen und Anfragen mit Rückfrage für den Konsumenten
- vollständige inhaltliche Entkopplung

Als Kommunikationsparadigma zur Realisierung einer solchen Middleware ist Publish/Subscribe am besten geeignet, weil es als einziges diese Formen der Entkopplung gewährleisten kann, ohne das Paradigma zu verletzen.

Im nächsten Kapitel wird daher ein Konzept vorgestellt, mit dem es gelingt eine Publish/Subscribe-basierende Middleware zu realisieren, die in der Lage ist, das eingangs geschilderte Problem zu lösen.

Kapitel 4.

Inhaltsentkopplung durch Routing in einer Vermittlungstopologie

Inhalt

4.1 Verwandte Arbeiten **51**
4.1.1 Publish/Subscribe 52
4.1.2 Sensornetze 58
4.1.3 Pervasive Computing 59
4.1.4 Zusammenfassung 63
4.2 Definition der Vermittlungstopologie **63**
4.2.1 Komplexität 67
4.2.2 Arten der Datenverarbeitung 69
4.3 Vorstellung einer geeigneten Systemarchitektur **71**
4.3.1 Zusammenspiel zwischen den Brokern 72
4.3.2 Aufbau des Brokers 73
4.4 Routing in einer Vermittlungstopologie **77**
4.4.1 Auf Flooding basierende Ansätze 77
4.4.2 Routing ohne Ankündigungen 78
4.4.3 Routing mit Ankündigungen 78
4.5 Bewertung gefundener Pfade **79**
4.5.1 Netzwerkabstraktionsschicht 80
4.5.2 Anwendungsabstraktionsschicht 81
4.5.3 Vermittlungsschicht 82
4.6 Anforderungen an die Anwendungen **83**
4.6.1 Anwendungsklassen 83
4.6.2 Besonderheiten der Schnittstellen 86
4.6.3 Inhaltliche Anforderungen 87
4.7 Zusammenfassung **89**

Entsprechend der Ergebnisse des vorhergehenden Kapitels können die durch die Problemstellung gegebenen Anforderungen durch eine Publish/Subscribe-basierende Middleware erfüllt werden, die

- räumliche Entkopplung,
- gegenwärtige und eingeschränkt auch zukünftige zeitliche Entkopplung,
- Entkopplung des Produzenten durch Fire and Forget,
- Anbindung des Konsumenten durch kontinuierliche Anfragen und Registrierung mit Rückfrage, sowie
- vollständige inhaltliche Entkopplung

gewährleistet. Um dabei inhaltliche Entkopplung gewährleisten zu können, müssen Anwendungen existieren, die Nachrichten von Produzenten oder bereits verarbeitete Nachrichten weiterverarbeiten können, so dass sie für weitere Konsumenten interessant werden. Diese Anwendungen werden im Folgenden als Verarbeiter bezeichnet. Besondere Herausforderungen dabei sind

Zuordnung, das Interesse von Konsumenten und die Nachrichten von Produzenten zusammenzuführen auch wenn diese erst durch einen oder mehrere Verarbeitungsschritte zueinander passen,

Einbindung, automatisch die richtigen Verarbeiter in den Kommunikationspfad einzubinden, so dass eine Nachricht am Ende auf das Interesse eines Konsumenten passt,

Effektivität, unnötige Nachrichtenverarbeitungsschritte, nach denen die Nachricht für keine Konsumenten mehr interessant ist, zu vermeiden, und

Effizienz, Verarbeiter effizient einzusetzen auch wenn mehrere Verarbeiter für den gleichen Verarbeitungsschritt verfügbar sind.

Eine scheinbar unmittelbare Lösung für dieses Problem ist die Abbildung eines Verarbeiters auf einen Konsumenten für die Ausgangsnachrichten und einen Produzenten für die verarbeiteten Nachrichten. Diese Abbildung würde die Nutzung eines existierenden Publish/Subscribe-Systems ohne inhaltliche Entkopplung erlauben. Für die Middleware geht dadurch der Zusammenhang zwischen diesem Konsumenten und dem dazugehörigen Produzenten verloren.

Der Ansatz ist allerdings nicht effektiv. Sobald der Verarbeiter sein Interesse an das Publish/Subscribe-System in Form eines Abonnements übertragen hat, werden ihm entsprechende Nachrichten zugestellt, die er dann in verarbeiteter Form wieder veröffentlichen muss. Dabei ist insbesondere aufgrund der räumlichen Entkopplung zwischen Anwendungen durch das Publish/Subscribe-System für den Verarbeiter nicht erkennbar, ob überhaupt Konsumenten oder weitere Verarbeiter mit Interesse an diesen Nachrichten existieren.

Weiterhin ist der Ansatz nicht effizient. Sobald mehrere Verarbeiter für einen Nachrichtenverarbeitungsschritt existieren, müssen diese die Ausgangsnachrichten abonnieren, bekommen diese zugestellt und müssen die verarbeiteten Nachrichten wieder veröffentlichen.

Eine Abstimmung eines aktiven Verarbeiters oder eine Aufteilung der Nachrichten auf die Verarbeiter ist aufgrund der gewährleisteten räumlichen Entkopplung zwischen Anwendungen durch das Publish/Subscribe-System nicht möglich.

Um alle genannten Anforderungen erfüllen zu können, ist es somit notwendig, dass die Rolle des Verarbeiters durch die Middleware vorgesehen und dessen Einbindung in die Nachrichtenverteilung durch die Middleware koordiniert wird.

In diesem Abschnitt wird ein Konzept vorgestellt, mit dem sich ein Publish/Subscribe-System mit den eingangs beschriebenen Anforderungen realisieren lässt. Dazu wird zuerst ein Überblick über verwandte Arbeiten gegeben. Diese werden mit Fokus auf die Verwendbarkeit in heterogenen und veränderlichen Umgebungen sowie auf ihre Möglichkeiten zur Realisierung vollständiger inhaltlicher Entkopplung diskutiert.

Anschließend werden ein Konzept und eine geeignete Systemarchitektur vorgestellt, die eine Abstraktion von der inhaltlichen Entkopplung und der Heterogenität der Netzwerktopologie aus Sicht des Routings zulassen und dadurch das Finden von Kommunikationspartnern und geeigneten Pfaden zwischen diesen auf Routingalgorithmen mit speziellen Anforderungen abbildet. Anschließend werden verschiedene inhaltsbasierte Routing-Strategien auf ihre Eignung hinsichtlich dieser Anforderungen diskutiert. Darauf folgt die Diskussion der grundsätzlichen Anforderungen und die beispielhafte Realisierung einer Routingmetrik in einem solchen Umfeld.

Den Abschluss bildet eine Diskussion der Anwendungsschicht. Der Fokus liegt dabei auf der Unterteilung der Anwendungen hinsichtlich einfacher Dienstgüteanforderungen, den notwendigen Schnittstellen zur Middleware sowie der Diskussion der Eingliederung bestehender Anwendungen.

4.1. Verwandte Arbeiten

Im folgenden Abschnitt werden verwandte Arbeiten vorgestellt und diskutiert. Dabei liegt der Fokus auf Publish/Subscribe-Systemen mit inhaltsbasiertem Routing. Insbesondere werden diese im Hinblick auf ihre Eignung in heterogenen, veränderlichen Netzwerktopologien und ihre Fähigkeit zur Gewährleistung von inhaltlicher Entkopplung untersucht.

Ein Bereich mit ähnlichen Eigenschaften, insbesondere in Bezug auf den Bedarf nach infrastrukturfreien Routingverfahren, sind Sensornetze. Verwandte Arbeiten aus diesem Bereich werden im Anschluss mit Fokus auf die speziellen Anforderungen der Problemstellung diskutiert.

Auch im Bereich des Pervasive Computing und der intelligenten Umgebungen gibt es verschiedene Vorarbeiten. Auch diese werden unter dem Fokus der Verwendbarkeit in heterogenen, veränderlichen Topologien sowie der Gewährleistung der notwendigen Entkopplungseigenschaften diskutiert.

4.1.1. Publish/Subscribe

SIENA

Eines der ersten verteilten Publish/Subscribe Systeme, zwischen dessen Brokern inhaltsbasiertes Routing eingesetzt wurde, ist SIENA (Scalable Internet Event Notification Architectures) (CARZANIGA, ROSENBLUM und WOLF, 2000; CARZANIGA, ROSENBLUM und WOLF, 2001). SIENA zielt auf den internetweiten Einsatz und setzt den Fokus auf Skalierbarkeit und Effizienz. Es werden hierarchische Client/Server-Topologien sowie azyklische und zyklische Peer-to-Peer-Topologien unterstützt.

Als Nachrichten kommen bei SIENA nicht typisierte Mengen typisierter Attribute zum Einsatz. Abonnements werden als Konjunktion aus binären Operationen auf Attributen realisiert. Ankündigungen werden ebenfalls unterstützt, unterscheiden sich von den Abonnements aber dadurch, dass binäre Operatoren auf dem gleichen Attribut disjunkt verknüpft werden.

Die Nachrichtenübermittlung bei SIENA erfolgt nach dem best-effort Prinzip unter der Prämisse, dass die Nachrichtenfilterung so früh wie möglich und die Nachrichtenreplikation so spät wie möglich im Kommunikationsverlauf erfolgen soll, um Bandbreite zu sparen.

Neben einfachen Filtern können auch so genannte Pattern benutzt werden. Sie bestehen aus mehreren Filtern, die auf aufeinander folgende Nachrichten passen müssen, damit diese an einen Konsumenten zugestellt werden. Eine Aggregation oder Verarbeitung von Nachrichten auf Basis dieser Pattern ist allerdings nicht vorgesehen.

SIENA ermöglicht entkoppelte Kommunikation zwischen Produzenten und Konsumenten, wobei inhaltliche Entkopplung überhaupt nicht vorgesehen ist. Mit dem Fokus auf IP-basierenden Weitverkehrsnetze ist SIENA für den Einsatz in heterogenen, veränderlichen Umgebungen nicht geeignet. Ähnliche Eigenschaften haben auch Elvin4 (SEGALL et al., 2000) und JEDI (CUGOLA, DI NITTO und FUGGETTA, 2001).

Rebeca

Die Unterstützung von Mobilität wurde zuerst in REBECA realisiert (FIEGE et al., 2003). Dazu werden die Broker in drei Arten bezüglich ihrer Aufgaben im Brokernetzwerk unterschieden. Lokale Broker befinden sich auf denselben Geräten wie Produzenten und Konsumenten und sorgen für die Abstraktion dieser vom Netzwerk. Grenzbroker befinden sich am Rand des Brokernetzwerks und verwalten einen oder mehrere Lokale Broker. Innere Broker verbinden lediglich Grenzbroker und andere innere Broker miteinander.

In REBECA wird zwischen physischer und logischer Mobilität unterschieden. Physische Mobilität bezieht sich auf die Systemebene und bedeutet, dass sich für einen lokalen Broker durch die Veränderung seiner Position in der Netzwerktopologie der Anknüpfungspunkt an das Brokernetzwerk, also der zuständige Grenzbroker, ändert. Dieses Roaming wird in REBECA durch spezielle Algorithmen für die Konsumenten transparent realisiert.

In Publish/Subscribe Systemen sind weiterhin mobile Konsumenten denkbar, deren Interesse an Nachrichten von ihrer aktuellen Position abhängt. Dabei kann eine Positionsänderung das Interesse verändern, ohne dass dazu Roaming zu einem anderen Grenzbroker nötig ist. Dieser Prozess wird in REBECA als logische Mobilität respektive Mobilität auf Anwendungsebene adressiert und durch die Definition positionsabhängiger Filter und entsprechender Routingalgorithmen realisiert.

Trotz der Unterstützung mobiler Konsumenten und damit mobiler lokaler Broker basiert REBECA auf einem relativ statischen, homogenen Netzwerk aus inneren und Grenzbrokern. Auch die Nutzung von Peer-to-Peer-Overlays zur Koordination des Brokernetzwerkes, wie sie für REBECA durch TERPSTRA et al. (2003) realisiert wird, ist für heterogene, veränderliche Topologien nur sehr begrenzt geeignet.

Unterstützung von Ad-hoc Topologien

Die Unterstützung von homogenen und heterogenen Ad-hoc Topologien durch Publish/Subscribe Systeme wurde mehrfach erforscht. STEAM (MEIER und CAHILL, 2002) basiert auf Gruppenkommunikation zwischen nah beieinander liegenden Geräten und erlaubt die Zustellung von Nachrichten basierend auf Nähe des Konsumenten zum Produzenten.

Mit GREEN (SIVAHARAN, BLAIR und COULSON, 2005) wurde ein Publish/Subscribe System vorgestellt, dass sehr schnelle und effiziente Rekonfiguration unterstützt und somit besonders für heterogene Umgebungen geeignet ist. Durch eine um Plugins erweiterbare Systemarchitektur ist das System aufgrund des geringen Speicherverbrauchs auch auf ressourcenarmen mobilen Geräten einsetzbar. Andere Publish/Subscribe Systeme mit speziellem Fokus auf Rekonfigurierbarkeit sind DREAM (LECLERCQ, QUÉMA und STEFANI, 2004) und REDS (CUGOLA und PICCO, 2006).

Von PETROVIC, MUTHUSAMY und JACOBSEN (2005) wird der Einsatz von einfachem inhaltsbasiertem Routing mit regelmäßigen Ankündigungen in Ad-hoc-Topologien untersucht. Durch die regelmäßige Verbreitung der Ankündigungen wird auf Veränderungen in der Topologie reagiert. Außerdem werden zwei Erweiterungen vorgestellt, die die Anzahl der erfolgreich zugestellten Nachrichten deutlich steigern können.

Die erste, CBR-FT, geht davon aus, dass ein Konsument sich zwischen zwei Ankündigungen nicht zu weit von seiner vorherigen, bekannten Position entfernt hat. Somit wird, wenn eine Nachricht nicht mehr zugestellt werden kann, der Konsument über die Weiterleitung an alle Nachbarn über wenige Hops gesucht. Die Netzwerklast wird dadurch deutlich erhöht, aber der Anteil erfolgreich zugestellter Nachrichten kann enorm gesteigert werden. Die Möglichkeit, dass ein Pfad bereits weit vom Konsumenten entfernt zerstört wird, und dieser durch die lokale Suche nicht mehr gefunden wird, wird nicht näher diskutiert. Daher kann nicht ausgeschlossen werden, dass CBR-FT nur in den untersuchten Szenarien derart erfolgreich ist.

Die zweite Erweiterung, RAFT-CBR, nutzt das seltene bis einmalige Fluten von Ankündigungen und den anschließenden Versand von Interesse nur zum Finden geeigneter Konsumenten. Anschließend werden alle Nachrichten per darunter liegendem Routingprotokoll direkt versandt. Dieser Ansatz ist in heterogenen Netzen nicht umsetzbar.

Außerdem wird von PETROVIC, MUTHUSAMY und JACOBSEN (2005) überdeckungsbasiertes und zusammenführendes Routing[1] für den Einsatz in Ad-hoc-Topologien adaptiert. Es wird gezeigt, dass durch überdeckungsbasiertes Routing die Netzwerklast mit steigender Konsumentenzahl ohne signifikanten Einfluss auf den Anteil erfolgreich zugestellter Nachrichten signifikant verringert werden kann. Zusammenführendes Routing erhöht gegenüber überdeckungsbasiertem Routing die Netzwerklast um etwa 30%, erhöht aber den Anteil erfolgreich zugestellter Nachrichten um etwa 10%.

Ein sehr interessanter probabilistischer Ansatz wird von COSTA und PICCO (2005) verfolgt. Interesse wird hier durch Fluten an alle Nachbarbroker verbreitet, die nicht weiter entfernt sind als ein vorgegebener Interessenhorizont von zum Beispiel einem Hop. Die zu verbreitenden Nachrichten werden anschließend entweder an Nachbarbroker verschickt, von denen bekannt ist, dass sie Interesse an dieser Nachricht haben, oder eine zufällig ausgewählte Anzahl von Nachbarn deren Anzahl durch eine Nachrichtenverbreitungsschwelle definiert ist. Einen anderer probabilistischer Ansatz wird von BALDONI et al. (2005) beschrieben.

Einige der hier vorgestellten Verfahren sind in heterogenen, veränderlichen Topologien einsetzbar. Eine inhaltliche Entkopplung ist jedoch bei keinem der Verfahren vorgesehen.

Event-Stream Processing

In vielen Szenarien werden von den Produzenten regelmäßig Ereignisse generiert und als Nachrichtenströme veröffentlicht. Das Event-Stream Processing (ESP) erlaubt den Konsumenten, Anfragen auf diese Nachrichtenströme zu stellen, die eine Bearbeitung der darin enthaltenen Nachrichten einschließen. Ein Beispiel ist die Aggregation mehrerer Datenströme zu einem einzigen.

Von EISENHAUER, BUSTAMANTE und SCHWAN (2001) wird mit ECho ein kanalbasiertes Publish/Subscribe-System vorgestellt, dass die Definition von Filtern durch Konsumenten erlaubt anhand derer die Inhalte von Kanälen manipuliert und dadurch abgeleitete Kanäle erstellt werden. Die Nachrichten aus den abgeleiteten Kanälen werden anschließend an die Konsumenten zugestellt. Hauptaugenmerk von ECho ist die hinsichtlich der erzeugten Netzwerklast effiziente Ausführung der Filter, indem diese so weit wie möglich zu den Produzenten durchgereicht und somit die Verarbeitung nah an den Produzenten durchgeführt wird.

Mit JECho (CHEN, SCHWAN und ZHOU, 2003) wird dieses kanalbasierte Konzept weiterentwickelt. Die Filter werden durch so genannte Modulatoren ersetzt, Java-Objekte, die beliebige Operationen auf Nachrichten durchführen können. Diese werden auf einem belie-

[1] siehe dazu Abschnitt 2.1.2 auf Seite 15.

bigen Broker ausgeführt, wobei ebenfalls auf eine Ausführung nah am Produzenten geachtet wird. Wie bei ECho wird davon ausgegangen, dass Nachrichten durch die Verarbeitung immer kleiner werden und durch die Nähe zum Produzenten Bandbreite gespart wird.

Außerdem ermöglicht JECho mit so genannten opportunistischen Kanälen die Mobilität von Konsumenten. Opportunistische Kanäle realisieren Mobilität auf Anwendungsebene, benötigen aber zwingend eine homogene Infrastruktur wie Mobile-IP (JOHNSON, PERKINS und ARKKO, 2004), die auf Netzwerkebene Mobilität unterstützt.

Beim ESP werden die Nachrichten in den Nachrichtenströmen anhand von vom Konsumenten definierten Anfragen verarbeitet. Damit erlaubt das ESP eine teilweise inhaltliche Entkopplung auf Seite des Produzenten. Eine vollständige inhaltliche Entkopplung ist nicht möglich.

Composite Event Detection

Composite Event Detection (CED), das Erkennen komplexer Ereignisse, kommt aus dem Bereich der aktiven Datenbanken und den dort verwendeten ECA-Regeln (DAYAL et al., 1988). ECA steht für Event Condition Action und bedeutet, dass als Reaktion auf definierte Ereignisse der Zustand der Datenbank anhand einer Reihe definierter Bedingungen geprüft wird. Wenn der Zustand den Bedingungen genügt, wird eine definierte Aktion in der Datenbank durchgeführt.

Da primitive Ereignisse häufig auftreten, und somit die Bedingungen häufig überprüft werden müssen, wurden verschiedene Ansätze zur Definition und Erkennung komplexer zusammengesetzter Ereignisse entwickelt (GATZIU und DITTRICH, 1994; CHAKRAVARTHY und MISHRA, 1994; CHAKRAVARTHY et al., 1994) um aktive Datenbanken effizienter arbeiten zu lassen.

Da in Publish/Subscribe-Systemen häufig ebenfalls primitive Ereignisse auftreten und in Form von Nachrichten kommuniziert werden, hat die CED auch hier Einzug gehalten. Es wurden verschiedene Formalismen zur Beschreibung zusammengesetzter Ereignisse entwickelt. Beispiele für solche Formalismen sind XML-basierende Beschreibungssprachen (XU, MA und HUANG, 2005) oder die Verwendung von Zustandsautomaten, die dazu um ein Zeitmodell und Parametrisierbarkeit erweitert wurden (PIETZUCH, SHAND und BACON, 2003).

Im Unterschied zu einfachem ESP ist die Erkennung zusammengesetzter Ereignisse aus primitiven Ereignissen häufig zustandsabhängig. Das heißt, die Reihenfolge, in der die primitiven Ereignisse auftreten müssen, um ein zusammengesetztes Ereignis auszulösen, kann vorgegeben werden. Die grundsätzliche Vorgehensweise, dass der Konsument die auftretenden zusammengesetzten Ereignisse beschreiben muss und dadurch nur eine teilweise inhaltliche Entkopplung gewährleistet wird, unterscheidet sich nicht.

Distributed Stream-Processing Systems

Die Übergänge zwischen ESP und CED sind fließend. Auch in Nachrichtenströmen können zusammengesetzte Ereignisse anhand zustandsabhängiger Anfragen gefunden und zugestellt werden. Dies wird von einer ganzen Reihe von Systemen unterstützt. Der Begriff Distributed Stream-Processing Systems (DSPS) hat sich hier etabliert.

Beispiele für derartige Systeme sind die auf Overlay Netzwerken basierenden Systeme Medusa (CHERNIACK et al., 2003), PIER (HUEBSCH et al., 2003) und SAND (AHMAD et al., 2005). Borealis (ABADI et al., 2005) unterstützt die Revision von Ergebnissen und kontextabhängige Anfragen. Cayuga (DEMERS et al., 2006) basiert auf nichtdeterministischen Zustandsautomaten. GATES (CHEN, REDDY und AGRAWAL, 2004) nutzt eine Grid-Infrastruktur als Kommunikationsbasis.

Diesen Systemen ist gemein, dass sie aufgrund ihres klar definierten schichtenweisen Aufbaus und ihrem Fokus auf Skalierbarkeit und den Einsatz in Weitverkehrsnetzen nicht für heterogene, veränderliche Topologien geeignet sind.

Ein anderer Ansatz zur Verarbeitung von Nachrichtenströmen im Netzwerk wird von SRIVASTAVA, MUNAGALA und WIDOM (2005) mit dem Konzept des Operator Placement verfolgt. Ursprünglich für den Einsatz in Sensornetzen entwickelt, ist die Grundidee, eine Anfrage aus konjunktiv verknüpften Filtern und Joins von einem Konsumenten ausgehend im Netz zu verbreiten. Anschließend werden die Filter so in dem sich ergebenden hierarchischen Graphen verteilt, dass die zu verarbeitenden Nachrichtenströme an optimalen Positionen aggregiert werden. Optimierungskriterien sind hier die Belastung der begrenzt leistungsfähigen Sensorknoten sowie die benutzte Bandbreite im Netzwerk.

Von PIETZUCH et al. (2006) wird das Operator Placement für den Einsatz in anderen Netzwerktopologien dahingehend weiterentwickelt, dass zur Optimierung der Routen eine Metrik aus Datenrate und Latenz verwendet wird. Es sollen unabhängig von der Netzwerktopologie gute Platzierungen für Operatoren und damit gute Pfade im Netz gefunden werden.

Genau wie bei CED und ESP Systemen ist eine vollständige inhaltliche Entkopplung bei DSPSs nicht vorgesehen. Der Ursprung aus dem Datenbankbereich ist hier klar erkennbar. Der Konsument möchte Informationen empfangen, also muss er beschreiben können, aus welchen Primitiven diese Informationen durch welche Operationen generiert werden können.

Gryphon

Das einzige dem Autor bekannte Publish/Subscribe-System, dass neben der Entkopplung in Raum und Zeit sowie von Produzent und Konsument auch vollständige inhaltliche Entkopplung ermöglicht, ist Gryphon (BANAVAR et al., 1999). Es ermöglicht inhaltsbasiertes Publish/Subscribe durch ein Netzwerk aus Brokern. Nachrichtenströme werden als Schemata und Interesse als Prädikate bezeichnet. Außerdem werden in Gryphon alte Nachrichten

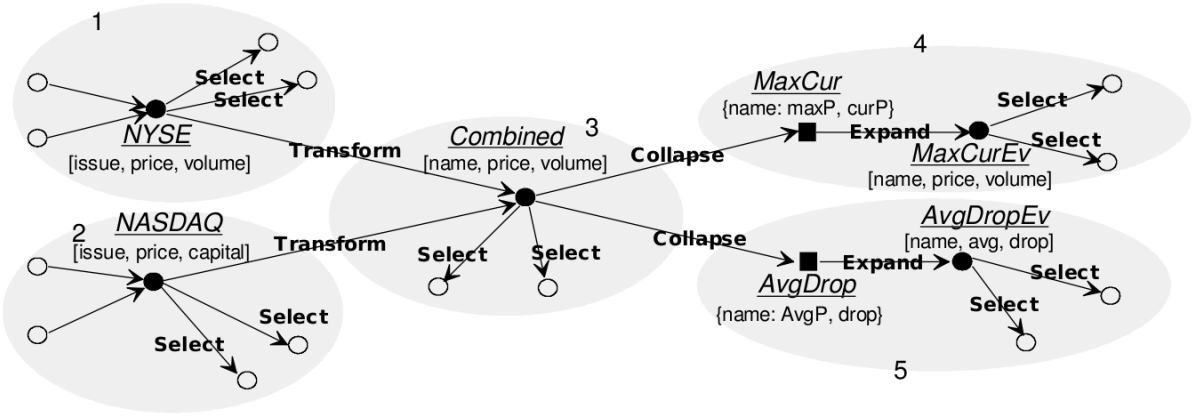


Abbildung 4.1.:
Beispiel für einen Information Flow Graph (Quelle: BANAVAR et al. (1999)).

aus einem Nachrichtenstrom konzeptuell als History bezeichnet. Gryphon unterscheidet zwischen zustandsloser und zustandsbehafteter Nachrichtenverarbeitung.

Das zentrale Konzept ist ein Information Flow Graph (IFG). Er enthält alle Ereignisströme im System. Ein IFG besteht aus zwei Komponenten, Information Spaces (IS) und Datenflüsse. Jeder IS beschreibt einen Typ von Nachrichten und ihm ist ein Schema zugeordnet, also ein kanalartiger Nachrichtenstrom mit Nachrichten dieses Typs. Jedem IS ist ein zentraler Knoten zugeordnet, der diesen Nachrichtenstrom repräsentiert.

Datenflüsse fließen zwischen den Knoten des IFG. Es gibt vier Arten von Datenflüssen, wobei jede mit einer Operation verbunden ist. `Select` verbindet die zentralen Knoten der IS mit Konsumenten, die Nachrichten mit dem entsprechenden Schema empfangen möchten, und erlaubt eine Auswahl von Nachrichten auf Basis inhaltsbasierter Filter, den so genannten Prädikaten.

`Transform` erlaubt die zustandslose Umwandlung von Nachrichten von einem Schema in ein anderes und verbindet daher die zentralen Knoten verschiedener IS. `Collapse` erzeugt aus einer History einen Zustand und verbindet daher einen zentralen Knoten eines IS mit einem Zustandsknoten eines anderen IS. `Expand` erzeugt eine Nachricht wenn sich ein Zustand verändert und verbindet somit den Zustandsknoten eines IS mit dessen zentralem Knoten. Durch die Kombination aus Collapse und Expand wird zustandsabhängige Nachrichtenverarbeitung ermöglicht. Abbildung 4.1 zeigt beispielhaft einen IFG.

Der IFG repräsentiert den gesamten möglichen Datenfluss im System und muss zur Realisierung des Systems auf eine Netzwerktopologie abgebildet werden. Dazu soll ein inhaltsbasiertes Routingsystem genutzt werden. Die Auswahl der Daten (`Select`) soll dicht am Produzenten stattfinden weil dabei die zu übertragende Datenmenge verkleinert wird. Die Umwandlung (`Transform`) soll dicht am Konsumenten stattfinden weil dabei Informationen verloren gehen. Es werden zwei Möglichkeiten aufgezeigt - Umordnen und neu Schreiben -

wie ein IFG entsprechend umgewandelt werden kann, um diesen Anforderungen gerecht zu werden.

Außerdem wird aufgezeigt, wie die Operation `Expand` abhängig von der Verfügbarkeit der Konsumenten optimiert werden kann. Anstelle aller erfolgten Zustandsübergänge seitdem der Konsument letztmalig verfügbar war, wird ihm, wenn er wieder verfügbar ist, nur die ökonomischste äquivalente Ereignisfolge, die zum aktuellen Zustand führt, übermittelt.

Obwohl Gryphon in allen Bereichen ausreichend entkoppelte Kommunikation ermöglicht, kommt es für den Einsatz in heterogenen, veränderlichen Umgebungen nicht in Frage. Die Erzeugung des IFG und dessen Abbildung auf eine Netzwerktopologie erfordert globales Wissen über alle Nachrichtenströme im gesamten System und stellt eine sehr statische Konfiguration des Systems dar. In einer eingangs unbekannten und sich häufig verändernden Topologie kann dieses Vorgehen nicht zum Erfolg führen.

4.1.2. Sensornetze

In drahtlosen Sensornetzen (POTTIE und KAISER, 2000) spielt entkoppelte Kommunikation ebenfalls eine große Rolle. Die Ursache hierfür liegt in der ebenfalls eingangs häufig unbekannten und durch den Ausfall von Sensorknoten ständigen Veränderungen unterliegenden Topologie sowie dem Fehlen einer globalen Infrastruktur. Dadurch ist adressbasierte Kommunikation für Sensornetze ineffizient und andere, entkoppelte Kommunikationsmechanismen haben sich etabliert.

Wesentliche Unterschiede von Sensornetzen sind eine oft geografische Verteilung der Sensorknoten, die für das Finden von Kommunikationspfaden ausgenutzt werden kann (GOLDIN et al., 2004), eine homogene drahtlose Kommunikationsinfrastruktur mit einem oft hohen Grad an Vermaschung, die den lokalen Austausch von Nachbarinformationen und lokales Clustering zulässt (HEINZELMAN, CHANDRAKASAN und BALAKRISHNAN, 2000), sowie eine Fokussierung auf geringen Ressourcenverbrauch und Energieeffizienz.

Mit dem Operator Placement wurde bereits ein aus Sensornetzen stammendes Konzept erläutert. Ein weiterer Ansatz, der auf heterogene, veränderliche Netze sehr gut übertragbar ist, ist Directed Diffusion (INTANAGONWIWAT, GOVINDAN und ESTRIN, 2000). Ähnlich zu inhaltsbasierten Routingansätzen für Publish/Subscribe Systeme wird als erster Schritt Interesse im Netzwerk verbreitet. Interesse wird als Filter auf den Nachrichten, die hier durch Name-Wert-Paare implementiert werden, realisiert. Durch die Verbreitung des Interesses durch Flooding entstehen als Gradienten bezeichnete Pfade in Richtung des Konsumenten.

Das besondere am Directed Diffusion Ansatz ist die Pfadauswahl. Dazu wird davon ausgegangen, dass Nachrichten als Ströme von den jeweiligen Produzenten veröffentlicht werden. Ist Interesse an Nachrichten vorhanden, wird der Nachrichtenstrom mit einer geringen Frequenz auf verschiedenen Pfaden in Richtung Konsument geschickt. Der Konsument arbeitet

mit positiver und negativer Verstärkung[2], die in Richtung Produzent übertragen werden, um die Nachrichtenfrequenz auf Pfaden zu erhöhen oder zu verringern. Dadurch ist eine flexible Reaktion auf bestimmte, insbesondere den Energie- und Verfügbarkeitsanforderungen in Sensornetzen entsprechende Veränderungen im Netzwerk möglich.

Von HEIDEMANN et al. (2001) wird Directed Diffusion um Verarbeitungsanweisungen in Filtern erweitert. Diese Filter, im Wesentlichen Signalverarbeitung und Aggregation, werden automatisch im Netz verteilt und erlauben eine Datenverarbeitung während der Übertragung. Allerdings werden die Filter mit der Anfrage vom Konsumenten ausgehend eingespeist und ermöglichen somit nur eine teilweise inhaltliche Entkopplung von Produzent und Konsument.

4.1.3. Pervasive Computing

Im Umfeld des Pervasive Computing gibt es auch Middleware-Projekte, die entkoppelte Kommunikation in verschiedenen Dimensionen gewährleisten. Einige davon werden im Folgenden vorgestellt.

Solar

Solar (CHEN, LI und KOTZ, 2008) stellt eine Publish/Subscribe artige Middleware für Pervasive Computing zur Verfügung. Sensoren bilden die Produzenten und veröffentlichen Nachrichtenströme. Anwendungen bilden die Konsumenten und sind an Nachrichten interessiert. Diese können sie über Operatoren definieren. Als Datenformate für Nachrichten werden beispielhaft Binärdaten, serialisierte Objekte, Attribut-Wert-Paare und XML-Dokumente genannt. Operatoren sind eigenständige Datenverarbeitungskomponenten. Operatoren werden von den Konsumenten über eine XML-basierte Beschreibungssprache parametrisiert und verknüpft. Über die Verknüpfung von Operatoren entsteht ein so genannter Operatorengraph. Die Nachrichtenströme werden als Kanäle zwischen Produzenten, Operatoren und Konsumenten realisiert.

In einem so genannten Planet-Overlay, einem Overlay-Netzwerk, in diesem Fall Pastry (ROWSTRON und DRUSCHEL, 2001), dass eine Reihe von Rechnern, die so genannten Planeten verbindet, werden die Operatoren ausgebracht und bilden dort den Operatorengraph ab. In dem dadurch entstehenden Netz aus Kanälen von den Produzenten in den Operatorengraph, durch die Operatoren hindurch bis zu den Konsumenten, werden die Nachrichtenströme über Multicast-Mechanismen ausgeliefert.

In seiner grundsätzlichen Funktionsweise ähnelt der Aufbau von Solar den IFGs von Gryphon. Der Operatorengraph ist weniger komplex und weniger statisch, muss allerdings auch in einer zuverlässigen, vollverbundenen Infrastruktur, wie zum Beispiel einem Overlay-Netzwerk, abgebildet werden. Damit eignet sich Solar nicht für heterogene, veränderliche

[2] im Ansatz als Reinforcement bezeichnet

Topologien. Weiterhin werden die Operatoren von den Anwendungen, also den Konsumenten definiert. Daher wird nur teilweise inhaltliche Entkopplung gewährleistet.

MundoCore

MundoCore (AITENBICHLER, KANGASHARJU und MÜHLHAUSER, 2007) als Teil des Mundo-Projektes (AITENBICHLER, 2006) stellt eine Plattform für die Kommunikation in Intelligenten Umgebungen zur Verfügung. Dazu wird ein Schichtenmodell definiert, dass aus folgenden Schichten besteht:

Anwendungsschicht - Sie enthält die Anwendungen.

Sprachanbindungsschicht (Language Binding) - Anwendungsdaten werden in Nachrichten verpackt.

Inhaltsbasierte Vermittlungsschicht (Brokering) - Nachrichten werden anhand inhaltlicher Kriterien vermittelt.

Adressbasierte Vermittlungsschicht (Routing) - Nachrichten werden anhand von Geräteidentifikatoren zu nicht benachbarten Geräten versandt.

Transportschicht - die Kommunikation mit benachbarten Geräten wird sichergestellt und unterschiedliche Kommunikationstechnologien werden vereinheitlicht.

Dieses Schichtenmodell wird in einem Mikrokernel modular abgebildet. Dazu gibt es für jede Schicht verschiedene Module aus denen vielfältige Kommunikations-Stacks gebildet werden können. Beispiele sind in der Transportschicht die Unterstützung von IP, Bluetooth und Infrarot, in der adressbasierten Vermittlungsschicht Nachbar-zu-Nachbar-Vermittlung, die Nutzung hierarchischer Strukturen oder strukturierter Overlays sowie in der inhaltsbasierten Vermittlungsschicht kanalbasierte oder inhaltsbasierte Vermittlung.

Inhaltsbasiertes Publish/Subscribe wird vom Modul für inhaltsbasierte Vermittlung in der inhaltsbasierten Vermittlungsschicht realisiert. Es verlangt den Versand von Ankündigungen und gleicht diese mit dem vorhandenen Interesse der Konsumenten ab. Für Ankündigungen sowie Interesse werden konjunktive Filter verwendet. Je nach Wahl des darunter liegenden Moduls zur adressbasierten Vermittlung funktioniert die inhaltsbasierte Vermittlung in strukturierten Overlay-Topologien genau wie mit strukturfreier Nachbar-zu-Nachbar-Kommunikation.

MundoCore ist sehr vielseitig und unter anderem für den Einsatz in heterogenen, veränderlichen Umgebungen geeignet. Eine Entkopplung in Raum, Zeit sowie von Produzent und Konsument ist möglich genau wie inhaltsbasierte Kommunikation zwischen Nachrichtenproduzenten und -konsumenten. Allerdings ist bei MundoCore keine inhaltliche Entkopplung irgendeiner Art vorgesehen.

METEOR

Das zentrale Konzept von METEOR (JIANG et al., 2004; JIANG et al., 2008) ist das Assoziative Rendezvous in so genannten Rendezvous Points (RPs). METEOR definiert Nachrichten als Tripel (Kopf, Aktion, Daten). Der Kopf enthält das Profil, eine Menge von Attributen, die die Nachricht inhaltlich beschreiben sowie eine Gültigkeit. Die Daten sind der Anwendung überlassen. Die Aktion beschreibt, was passieren soll, wenn die Nachricht im RP auf eine andere Nachricht trifft, und die beiden Profile zueinander passen. Es wird bei den Profilen nicht zwischen Ankündigungen, Nachrichten und Interesse unterschieden. Die Semantik ist hier den Anwendungen überlassen.

Mögliche Aktionen sind das `Speichern` der versandten Nachricht im und das `Abrufen` auf die versandte Nachricht passender Nachrichten aus dem RP, das Anfordern einer `Datenbenachrichtigung`, wenn im RP eine passende `Interessenbenachrichtigung` enthalten ist und umgekehrt sowie das `Löschen` einer Datenbenachrichtigung respektive Interessenbenachrichtigung.

Ob es sich um eine Daten- oder Interessenbenachrichtigung handelt, ist vom Absender der Nachricht abhängig. Ein Produzent versendet neben Nachrichten mit der Aktion `Speichern` Nachrichten mit der Aktion `Interessenbenachrichtigung` und erhält dann Interessenbenachrichtigungen vom RP. Ein Konsument versendet entsprechend Nachrichten mit den Aktionen `Abrufen` und `Datenbenachrichtigung`. So ist sichergestellt, dass immer Produzenten mit Konsumenten kommunizieren.

Alle Nachrichten werden im RP entsprechend ihrer Gültigkeit gespeichert und mit anderen Nachrichten abgeglichen. Danach werden sie aus dem RP entfernt. Rendezvous Points sind in drei Schichten realisiert.

Assoziatives Rendezvous - hier werden die Nachrichten auf Basis ihrer Profile verglichen und bei Treffern die entsprechenden Aktionen ausgeführt,

Inhaltsbasiertes Routing - zwischen den Rendezvous Points werden Nachrichten inhaltsbasiert versandt,

Overlay Netzwerk - auf Netzwerkebene sind die Rendezvous Points mit einem strukturieren Peer-to-Peer-Overlay verbunden.

Als Peer-to-Peer-Overlay kommt Chord (STOICA et al., 2001) zum Einsatz.

Das Konzept von METEOR ist sehr vielseitig und erlaubt die Einbindung transparenter Nachrichtenverarbeitung über Anwendungen, die sich als Produzent und Konsument am RP anmelden und anschließend Nachrichten umwandeln. Damit ähnelt METEOR sehr dem Konzept der Tuple Spaces. Durch die Verteilung der RPs über ein Peer-to-Peer-Overlay werden Probleme zentraler Architekturen wie Engpässe und zentrale Fehlerpunkte vermieden.

Allerdings setzt das Konzept der Aktionen zwingend eine bidirektionale Kommunikation zwischen RP und Konsument respektive Produzent sowie eine eindeutige Adressierbarkeit desselben voraus. Beides ist in heterogenen, veränderlichen Umgebungen nicht gewährleistet.

Auch Peer-to-Peer-Overlays, insbesondere Chord, sind für den Einsatz in solchen Umgebungen nicht geeignet.

SodaPop

Im Rahmen des DyNAMITE Projektes (KIRSTE, 2004) wurden die Kommunikationsanforderungen einer Multi-Agenten-Infrastruktur zur spontanen Kooperation von Geräten zur Bildung intelligenter Umgebungen untersucht. Dabei entstand SodaPop (HELLENSCHMIDT und KIRSTE, 2004b; HELLENSCHMIDT und KIRSTE, 2004a) als Kommunikationsmiddleware.

SodaPop basiert auf Kanälen, die so genannte Umsetzer (Transducer) miteinander verbinden. Die grundsätzliche Architektur eines verteilten Multi-Agenten-Systems besteht aus mehreren Schichten, in denen sich Kanäle und Umsetzer jeweils abwechseln. Kanäle vermitteln Nachrichten zwischen semantisch kompatiblen Umsetzern, die wiederum Nachrichten verarbeiten und in die jeweils nächste Kanalschicht einspeisen. Auf diese Weise gelangt ein Ereignis zum Beispiel von den Sensoren über eine Kanalschicht zu den Parsern, über eine weitere Kanalschicht zur Assistenzplanung und über eine letzte Kanalschicht zu den Aktuatoren. In diesem Fall sind Sensoren, Parser, Assistenzplanung und Aktuatoren jeweils Umsetzer, die paarweise durch Kanäle miteinander verbunden werden.

Kanäle geben die Semantik der Kommunikation vor, die Umsetzer abonnieren Ereignisse mit Hilfe komplexer Auswahlkriterien. Diese beinhalten inhaltsbasierte Filter zur Vorgabe, welche Nachrichten verstanden und verarbeitet werden können, eine Bewertung darüber, wie gut sie zur Verarbeitung dieser Nachricht geeignet sind, Informationen darüber, inwiefern eine Verarbeitung parallel zu einem anderen Umsetzer möglich ist sowie Informationen darüber, inwiefern eine Verarbeitung gemeinsam mit einem anderen Umsetzer möglich ist. Die Zuordnung der Nachrichten zu Umsetzern, die Aufteilung der Nachrichten zur gemeinsamen Verarbeitung in Umsetzern sowie die Auflösung von Konflikten ist Aufgabe der Kanäle.

SodaPop geht damit weit über die Definition einer Kommunikationsmiddleware hinaus. Es benötigt insbesondere sehr viel Domainwissen über den Inhalt der Kommunikation und die Funktion der Umsetzer, so dass eine korrekte Zuordnung von Ereignissen zu Umsetzern und eine Auflösung von Konflikten gelingen können.

Zur eigentlichen Kommunikation wird auf Standard-Middleware zurückgegriffen. Genannt werden JXTA und UPnP (HELLENSCHMIDT, 2005). Zweiteres wurde für diesen Einsatzzweck um Service-Klassen, Service-Qualität und Nähe als zusätzliche Möglichkeiten zum Beschreiben und Finden von Geräten erweitert (KUTTER, NEUMANN und SCHMITZ, 2005). Grundsätzlich werden an eine benutzbare Kommunikationsmiddleware die Voraussetzungen der Bereitstellung von Peer-to-Peer-Kommunikation, gruppenbasiertem Multicast sowie Unicast zwischen Geräten gestellt.

Betrachtet man SodaPop und die darunterliegende Kommunikationsmiddleware als Einheit, wird durch die automatische Einbindung von Umsetzern eine inhaltlich entkoppelte Kommunikation zwischen Anwendungen bereitgestellt. Auch räumliche und zeitliche Entkopplung sind zu einem gewissen Grad möglich. Die Fire and Forget Entkopplung der Produzenten wird genau wie die kontinuierliche Anfrage bei Konsumenten bereitgestellt. Allerdings sowohl die sehr statische Definition von äußerst spezialisierten Kanälen als auch die Voraussetzung von Jeder-mit-Jedem-Kommunikation im gesamtem System begrenzt den Einsatz auf homogene, möglichst TCP/IP-basierende Systeme in der Netzwerkschicht und ein sehr eingeschränktes, auf die geplanten Kanäle abgestimmtes agentenbasiertes Anwendungsfeld in der Anwendungsschicht.

4.1.4. Zusammenfassung

Entkoppelte Kommunikation entsprechend der eingangs definierten Anforderungen, insbesondere die inhaltliche Entkopplung, in Verbindung mit der Implementierbarkeit in heterogenen, veränderlichen Topologien wird von keiner der vorgestellten Forschungsprojekte zufriedenstellend ermöglicht.

Während Gryphon als Publish/Subscribe-System den benötigten Grad an Entkopplung in allen Dimensionen gewährleisten kann, ist die starre, von Anfang an globales Wissen voraussetzende Struktur des Information-Flow-Graphen für eine Nutzung in heterogenen, veränderlichen Umgebungen nicht geeignet.

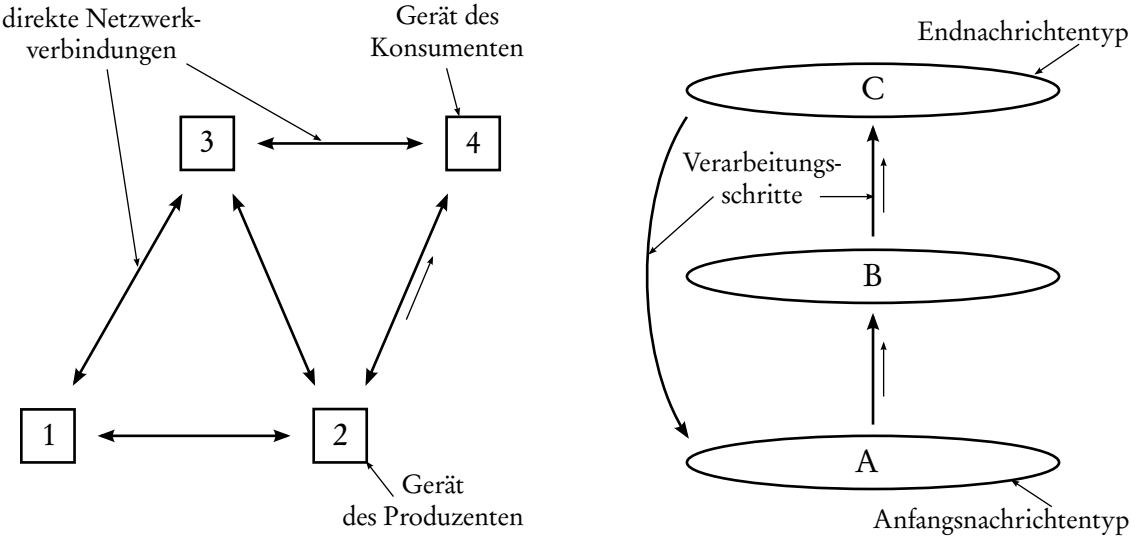
Das speziell für den Einsatz in intelligenten Umgebungen und damit heterogenen, veränderlichen Topologien entwickelte MundoCore hingegen ermöglicht nur eine Entkopplung in einigen Dimensionen. Insbesondere eine inhaltliche Entkopplung ist nicht vorgesehen.

Im folgenden Abschnitt werden die theoretischen Grundlagen für eine inhaltlich entkoppelte Kommunikation in heterogenen, veränderlichen Topologien gelegt.

4.2. Definition der Vermittlungstopologie

Die Kommunikation in heterogenen, veränderlichen Netzwerken und die inhaltliche Entkopplung entsprechen jeweils dem Finden eines Pfades in einer Topologie. Für die Kommunikation ist dies die **Netzwerktopologie**. Hier muss ein Pfad zwischen zwei Geräten gefunden werden, der benutzt werden kann, um Nachrichten von einer Anwendung auf dem ersten Gerät zu einer Anwendung auf dem zweiten Gerät zu übertragen. Das entstehende Routingproblem kann sehr effizient durch inhaltsbasiertes Routing (siehe Abschnitt 2.1.2 auf Seite 15) gelöst werden. Ein einfaches Beispiel für einen Pfad durch eine Netzwerktopologie mir vier Geräten ist in Abbildung 4.2(a) dargestellt.

Die inhaltliche Entkopplung entspricht dem Finden eines Pfades durch eine **Verarbeitungstopologie**. Diese enthält alle Nachrichtentypen als Ecken und die möglichen Verarbeitungsschritte durch Verarbeiter als gerichtete Kanten. Sie ähnelt dem Information Flow



(a) Netzwerktopologie (b) Verarbeitungstopologie

Abbildung 4.2.:
Eine Netzwerktopologie und eine Verarbeitungstopologie repräsentiert jeweils durch einen gerichteten Graphen mit einem Pfad vom Gerät des Produzenten zum Gerät des Konsumenten respektive vom Ausgangstyp zum Endtyp einer Nachricht.

Graph von Gryphon (siehe Abschnitt 4.1.1 auf Seite 56). Das Ziel ist es hier, einen Pfad vom Ausgangstyp einer Nachricht, wie sie von einem Produzenten veröffentlicht wurde, zu einem Endtyp dieser Nachricht, wie sie von einem Konsumenten verstanden wird, zu finden. Ein einfaches Beispiel für einen Pfad durch eine Verarbeitungstopologie mit drei Nachrichtentypen ist in Abbildung 4.2(b) dargestellt.

Zwischen diesen beiden Topologien gibt es eine Reihe von Gemeinsamkeiten und Unterschieden. Gemeinsamkeiten sind,

- dass beide Topologien eingangs allen Teilnehmern unbekannt sind,
- sie sich über die Zeit verändern und damit ein Erlernen und ständiges Aktualisieren mühsam und bedingt sinnvoll ist,
- die Repräsentation von Nachbarschaftsbeziehungen durch Kanten, die der Middleware an beiden Enden dieser Kanten bekannt sind,
- dass jede Kante gerichtet und mit einem Maß für den Aufwand der Nachrichtenübertragung entlang dieser Kante gewichtet ist, sowie
- dass ein gefundener Pfad idealerweise derjenige mit dem geringsten Pfadgewicht, also den entlang des Pfades summierten Kantengewichten, ist.

Es gibt drei wesentliche Unterschiede zwischen den beiden Topologien:

- In der Netzwerktopologie sind die Kanten zwar gerichtet, aber zwischen zwei benachbarten Ecken befinden sich immer zwei entgegengesetzt gerichtete Kanten, so dass eine

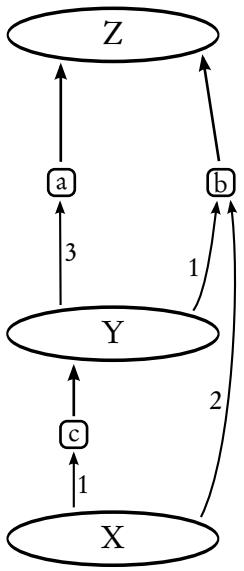
Übertragung von Nachrichten und Steuerungsbefehlen in beide Richtungen problemlos möglich ist. In der Verarbeitungstopologie kommen die Kanten jedoch im Normalfall nicht paarweise vor. Eine Verarbeitung ist nur von einem Nachrichtentyp zu einem anderen möglich. Die Übertragung von Steuerungsbefehlen kann von der Middleware allerdings in beide Richtungen simuliert werden, ohne den Verarbeiter zu involvieren. Voraussetzung hierfür ist, dass der Middleware das Vorhandensein der Kante vorher bekannt ist, also der Verarbeiter der Middleware mitgeteilt hat, dass er Nachrichten von einem bestimmten Nachrichtentyp in einen anderen überführen kann.

- In der Netzwerktopologie werden Nachrichten von einem Gerät zu einem benachbarten übertragen und kommen dort in identischer Form an. Es muss eventuell eine Fragmentierung und Defragmentierung durchgeführt sowie der Verlust von Nachrichten vermieden werden. Diese Probleme sind vielfach gelöst(COMROE und COSTELLO, 1984; CLARK, 1982). In der Verarbeitungstopologie hingegen verändert sich eine Nachricht bei der Übertragung. Insbesondere die Größe der Nachricht kann sich signifikant verändern.
- In der Netzwerktopologie verbinden alle Kanten genau zwei benachbarte Geräte. Eine Nachricht kann jederzeit von einem der Geräte zu einem anderen übertragen werden und wird dort als genau eine Nachricht empfangen. In der Verarbeitungstopologie ist diese zustandslose Verarbeitung einer Nachricht von einem Ausgangstyp in einen Endtyp ebenso möglich. Allerdings existiert noch eine zweite Form der Verarbeitung, die zustandsbehaftete Verarbeitung. Diese unterscheidet sich in zwei Punkten von der zuvor beschriebenen zustandslosen Verarbeitung. Zum Einen müssen dem Verarbeiter alle veröffentlichten Nachrichten des Ausgangstyps zugestellt werden, da er nur dann die für die Verarbeitung nötigen Nachrichten auswählen kann. Zum Anderen können dabei mehrere Nachrichten in ihren jeweiligen Ausgangstypen zu genau einer Nachricht im Endtyp verarbeitet werden. Abbildung 4.3 illustriert die zustandsbehaftete Verarbeitung in der Verarbeitungstopologie. Abschnitt 4.2.2 beschreibt noch einmal detailliert die Unterschiede zur zustandslosen Datenverarbeitung.

Bei der inhaltlich entkoppelten Kommunikation in einem verteilten System können die beiden Aufgaben einzeln nicht mehr gelöst werden. In der Netzwerktopologie ist ein Pfad vom Produzenten zum Konsumenten nutzlos, da dieser die Nachricht nicht verstehen würde. In der Verarbeitungstopologie kann ein solcher Pfad nicht gefunden werden, da die Verarbeiter in der Netzwerktopologie verteilt sind. Die Middleware kennt zwar beide Enden der Kante zwischen zwei Nachrichtentypen aber nur auf dem Gerät, auf dem der Verarbeiter auch installiert ist. Eine Kombination der beiden Topologien ist notwendig. Dazu werden Die Netzwerktopologie und die Verarbeitungstopologie in eine **Vermittlungstopologie** überführt.

Die Netzwerktopologie sei durch einen attributierten, symmetrischen, stark zusammenhängenden, schleifenfreien, gerichteten Graphen beschrieben, wie er beispielhaft in Ab-

Abbildung 4.3:
Im Unterschied zur in Abbildung 4.2(b) dargestellten zustandslosen Nachrichtenverarbeitung, werden zustandsbehaftete Verarbeiter durch eine eigene Ecke dargestellt. Damit sie korrekt arbeiten können, müssen ihnen alle veröffentlichten Nachrichten der Ausgangstypen zugestellt werden. Die eingehenden Kanten werden mit der Anzahl der vom jeweiligen Nachrichtentyp typischerweise ausgewählten Nachrichten gewichtet, um eine Nachricht vom Endtyp zu erzeugen. Die ausgehende Verbindung wird analog zur, die zustandslose Nachrichtenverarbeitung repräsentierenden Kante mit dem Verarbeitungsaufwand zur Erzeugung einer Nachricht gewichtet. In der Abbildung erzeugt Verarbeiter a aus drei Nachrichten vom Typ Y eine Nachricht vom Typ Z, Verarbeiter b erzeugt aus einer Nachricht vom Typ Y und zwei vom Typ X eine Nachricht vom Typ Z. Verarbeiter c erzeugt aus einer Nachricht vom Typ X eine Nachricht vom Typ Y.



bildung 4.2(a) dargestellt ist. Die Verarbeitungstopologie sei durch einen attributierten, schwach zusammenhängenden, schleifenfreien, gerichteten Graphen beschrieben wie er in Abbildung 4.2(b) oder mit zustandsbehafteter Verarbeitung in Abbildung 4.3 dargestellt ist.

Die Vermittlungstopologie ergäbe sich nun bildlich als das Kreuzprodukt der beiden Graphen wenn jeder Verarbeiter auch auf jedem Gerät installiert wäre. Die Netzwerktopologie wird dabei horizontal für jeden Nachrichtentyp reproduziert. Die Verbindungen der Verarbeitungstopologie gegebenenfalls mit den die zustandsbehafteten Verarbeiter repräsentierenden Ecken bilden die vertikalen Verbindungen zwischen den Nachrichtentypen auf jedem Gerät falls auf jedem Gerät alle Verarbeiter installiert sind. Da dies in den meisten Vermittlungstopologien nicht der Fall ist, müssen anschließend alle Verarbeiter repräsentierenden Ecken und vertikalen Verbindungen, für die auf dem entsprechenden Gerät kein Verarbeiter installiert ist, aus der Vermittlungstopologie gestrichen werden. Das Entstehen einer Vermittlungstopologie aus der Netzwerktopologie aus Abbildung 4.2(a) und der Verarbeitungstopologie ohne zustandsbehaftete Verarbeiter aus Abbildung 4.2(b) skizziert Abbildung 4.4 sehr anschaulich.

Übertragen in die Vermittlungstopologie bedeutet das Problem der inhaltlich entkoppelten Kommunikation in einer heterogenen, veränderlichen Umgebung demzufolge das Zusammenführen von Produzenten und Konsumenten über einen effizienten Pfad, auf dem anschließend Nachrichten übermittelt werden können. Nachfolgend werden die Komplexität dieses Problems sowie der Umgang mit Spezialfällen, die durch die Aggregation von Nachrichten durch zustandsbehaftete Verarbeitung auftreten, diskutiert.

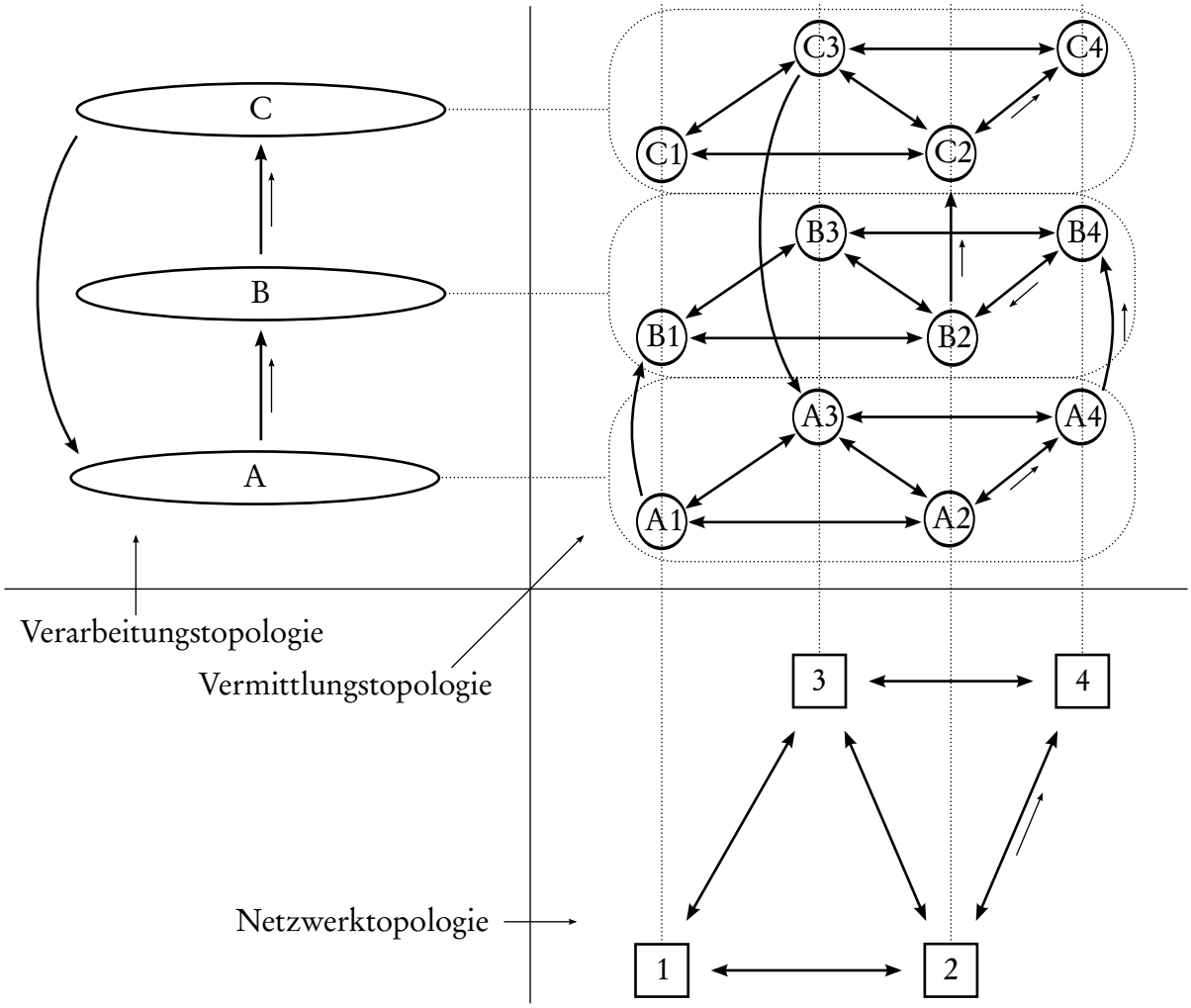


Abbildung 4.4.:
Aus der Netzwerktopologie (siehe auch Abbildung 4.2(a)) und der Verarbeitungstopologie (siehe auch Abbildung 4.2(b)) ergibt sich die Vermittlungstopologie mit einem nun nutzbaren Pfad vom Produzenten zum Konsumenten.

4.2.1. Komplexität

Zwei Faktoren haben einen wesentlichen Einfluss auf die Komplexität der Vermittlung zwischen einem Produzenten und einem Konsumenten. Aus Sicht eines Produzenten muss ein Konsument einerseits gefunden werden. Andererseits muss ein effizienter Pfad zwischen dem Produzenten und dem Konsumenten gefunden werden.

Für das Finden des Konsumenten ist es von Bedeutung, wie viele potentiellen Konsumenten sich in die Vermittlungstopologie befinden. Die Anzahl $|E|$ der Ecken, die als potentielle Konsumenten in Frage kommen ergibt sich als das Produkt der Anzahl $|T|$ in der Nachrichtentypen der Verarbeitungstopologie und der Anzahl $|G|$ der Geräte in der Netzwerktopologie: $|E| = |T| \cdot |G|$

Das Finden eines effizienten Pfades hängt stark von der Wahl eines geeigneten Routingalgorithmus ab. Die wesentliche Grundlage für die Komplexität eines solchen Algorithmus ist die Anzahl der für die Pfadsuche signifikanten Kanten in der Vermittlungstopologie. Die Anzahl $|K|$ dieser Kanten in Abhängigkeit von der Anzahl $|B|$ der Kanten in der Netzwerktopologie und der Anzahl $|V|$ der unterschiedlichen Verarbeitungsschritte in der Verarbeitungstopologie liegt zwischen $|K| = |B| \cdot |T| + |V|$, wenn jeder Verarbeiter nur auf genau einem Gerät installiert ist und $|K| = |B| \cdot |T| + |V| \cdot |G|$, wenn jeder Verarbeiter auch auf jedem Gerät installiert ist.

Nachfolgend werden die Auswirkungen einzelner Veränderungen in den Ausgangstopologien auf die Komplexität der Vermittlung zwischen einem Produzenten und einem Konsumenten in der Vermittlungstopologie untersucht. Dabei repräsentiert der Ausdruck $C(x+=1)$ den Anstieg der Komplexität, wenn der Parameter x um 1 erhöht wird.

Zusätzlicher Verarbeiter in der Verarbeitungstopologie ($C(|V|+=1)$)

Wird ein neuer Verarbeiter auf einem Gerät installiert, ohne dadurch einen neuen Nachrichtentypen einzuführen, oder ein existierender Verarbeiter auf einem zusätzlichen Gerät installiert, so bleibt $|E|$ unverändert und $|K|$ erhöht sich um 1. Ein zusätzlicher Verarbeiter, der keinen neuen Nachrichtentypen einführt, hat somit einen minimalen Einfluss auf die Komplexität der Pfadsuche in der Vermittlungstopologie.

Zusätzliche Kommunikationsverbindung in der Netzwerktopologie ($C(|B|+=2)$)

Eine zusätzliche Kommunikationsverbindung zwischen zwei Geräten erhöht die Anzahl der Kanten in der Netzwerktopologie um 2, da eine solche Verbindung immer aus zwei entgegengesetzt gerichteten Kanten besteht. $|E|$ bleibt wiederum unverändert, $|K|$ erhöht sich um $2|T|$.

Zusätzliches Gerät in der Netzwerktopologie ($C(|G|+=1)$)

Ein zusätzliches Gerät in der Netzwerktopologie muss ebenfalls wenigstens eine zusätzliche Kommunikationsverbindung, also zwei Kanten in der Netzwerktopologie mit sich bringen, damit selbige ein symmetrischer, zusammenhängender Graph bleibt. Es geht also mit der Erhöhung von $|G|$ um 1 eine Erhöhung von $|B|$ um $2n|1 \leq n \leq G$ einher. $|E|$ erhöht sich somit um $|T|$ und $|K|$ um $2n|T|$.

Zusätzlicher Nachrichtentyp in der Verarbeitungstopologie ($C(|T|+=1)$)

Damit die Verarbeitungstopologie ein schwach zusammenhängender Graph bleibt, muss zur Einführung eines zusätzlichen Nachrichtentypen wenigstens ein zusätzlicher Verarbeiter existieren, der Nachrichten in diesen Nachrichtentyp umwandeln kann. Es geht also mit

Parameterveränderung	$\|E\|$ erhöht sich um	$\|K\|$ erhöht sich um
$\|V\|+=1$	0	1
$\|B\|+=2$	0	$2\|T\|$
$\|G\|+=1$	$\|T\|$	$2n\|T\|$
$\|T\|+=1$	$\|G\|$	$\|B\|+m$

Tabelle 4.1.:
Erhöhung der Komplexität für die Erhöhung verschiedener Parameter. Für die Anzahl n der Geräte, mit denen ein neues Gerät verbunden ist, und die Anzahl m der Geräte, auf denen ein Verarbeiter zur Erzeugung eines neuen Nachrichtentyps installiert ist, gilt: $1 \leq n \leq |G|$ und $1 \leq m \leq |G|$.

der Erhöhung von $|T|$ um 1 eine Erhöhung von $|V|$ um $n|1 \leq m \leq |G|$ einher. $|E|$ erhöht sich demzufolge um $|G|$ und $|K|$ erhöht sich um $|B|+m$.

Die Auswirkungen der Erhöhung der vier untersuchten Parameter werden in Tabelle 4.1 dargestellt. Aus Ihnen lassen sich folgende Aussagen ableiten:

- Das Hinzufügen eines Verarbeiters, der keinen zusätzlichen Nachrichtentyp erzeugt, erhöht die Komplexität der Vermittlung minimal: $C(|V|+=1) < C(|B|+=2)$
- Das Hinzufügen eines Gerätes zur Netzwerktopologie erhöht die Komplexität der Vermittlung stärker als das Hinzufügen einer Kommunikationsverbindung zwischen zwei existierenden Geräten: $C(|B|+=2) < C(|G|+=1)$
- Sobald in der Vermittlungstopologie mehr Geräte als Nachrichtentypen existieren, was sehr wahrscheinlich ist, und damit auch mehr als doppelt so viele Kommunikationsverbindungen wie Nachrichtentypen, hat das Hinzufügen eines weiteren Nachrichtentyps den größten Einfluss auf die Komplexität der Vermittlung: $C(|G|+=1) \lessapprox C(|T|+=1)$

Diese Ergebnisse sind für die Diskussion der Skalierungsfähigkeit von Routingalgorithmen in der Vermittlungstopologie von Bedeutung. Diese findet sich in Abschnitt 5.4.2 auf Seite 115.

4.2.2. Arten der Datenverarbeitung

Wie in Abbildung 4.3 bereits angedeutet, wird genau wie im IFG von Gryphon zwischen zustandsloser und zustandsbehafteter Datenverarbeitung unterschieden. Zustandslose Datenverarbeitung orientiert sich dabei am mathematischen Vorbild einer Verarbeitungsvorschrift in Form einer Funktion. Aus einer beliebigen Nachricht des richtigen Nachrichtentyps generiert ein Verarbeiter durch die Anwendung seiner Verarbeitungsvorschrift eine neue Nachricht. Dies kann zum Beispiel eine Einheitenumwandlung eines Messwertes oder eine Formatumwandlung eines Dokumentes sein.

Bei der zustandsbehafteten Datenverarbeitung versetzen die bisher empfangenen Nachrichten den Verarbeiter in einen Zustand. Wenn sich dieser Zustand durch den Empfang einer weiteren Nachricht verändert, so wird eine neue Nachricht generiert. Ein einfaches Beispiel ist ein Verarbeiter, der aus einem Strom von Messwerten den laufenden Durchschnitt über die letzten zehn Messwerte bildet. Sein Zustand wird durch den Durchschnittswert der letzten zehn Messwerte beschrieben. Ändert sich dieser Zustand, also vermutlich bereits beim Empfang des nächsten Messwertes, generiert er eine neue Nachricht mit dem neuen Durchschnittswert. Ein komplexeres Beispiel ist ein Verarbeiter, der in Bildern einer Überwachungskamera Gesichter detektiert. Sein Zustand wird durch die bisher detektierten Gesichter beschrieben. Sobald auf einem Bild ein ihm noch unbekanntes Gesicht erkannt wird, das muss nicht mit dem nächsten empfangenen Bild der Fall sein, so generiert er eine Nachricht, die diese Zustandsveränderung repräsentiert.

Zwei für den Entwurf einer Middleware wichtige Aussagen lassen sich ableiten:

1. Ein zustandsloser Verarbeiter kann aus einer beliebigen Nachricht eine neue Nachricht generieren. Zwei Instanzen desselben zustandslosen Verarbeiter können sich also beliebig mit der Verarbeitung von Nachrichten abwechseln.
   Ein zustandsbehafteter Verarbeiter benötigt alle Nachrichten seiner Ausgangstypen um jede Veränderung seines Zustands korrekt erkennen und die entsprechenden Nachrichten generieren zu können. Sind also mehrere Instanzen eines zustandsbehafteten Verarbeiters in der Vermittlungstopologie vorhanden, so müssen alle Instanzen, die Nachrichten verarbeiten sollen, mit allen Ausgangsnachrichten versorgt werden. Alle anderen Instanzen sind nicht aktiv. Eine Verteilung des Bearbeitungsaufwands ist hier nicht möglich.
2. Ein zustandsbehafteter Verarbeiter kann Nachrichten mehrere Ausgangstypen benötigen, um seinen Zustand korrekt abzuleiten und neue Nachrichten zu generieren.
   Für zustandslose Verarbeiter ist die Annahme, dass sie immer Nachrichten von genau einem Ausgangstyp in Nachrichten eines Endtyps umwandeln ausreichend. Angenommen, es existiere ein zustandsloser Verarbeiter, der Nachrichten des Typs A und des Typs B in Nachrichten des Typs C umwandelte. Dann ist es theoretisch möglich, zwei Instanzen dieses Verarbeiters parallel arbeiten zu lassen. Per Definition können diese beiden Instanzen ohne Einschränkung ihrer Funktion jeweils mit beliebigen Ausgangsnachrichten versorgt werden. Eine gültige Versorgungsstrategie wäre es, dass die erste Instanz ausschließlich mit Nachrichten des Typs A und die zweite Instanz ausschließlich mit Nachrichten des Typs B zu versorgen. Die beiden Instanzen können somit als unterschiedliche Verarbeiter mit jeweils genau einem Ausgangs- und einem Endtyp betrachtet werden.

Diese beiden Aussagen begründen die gewählte und in den Abbildungen 4.2(b) und 4.3 skizzierte unterschiedliche Darstellungsweise von Verarbeitern.

Die in der ersten Aussage festgestellten Versorgungsanforderungen zustandsbehafteter Verarbeiter muss durch die Middleware befriedigt werden und schränkt die Optimierungsmöglichkeiten bei der Nachrichtenvermittlung stark ein. Eine einfache Lösung dieses Problems ist es, zustandsbehaftete Verarbeiter als Kombination aus Konsument und Produzent abzubilden und die Tatsache der zustandsbehafteten Datenverarbeitung vollständig vor der Middleware zu verbergen. Der Vorteil ist hier, dass jedem Verarbeiter alle Ausgangsnachrichten zugestellt werden und somit sichergestellt ist, dass er seinen aktuellen Zustand immer korrekt bewerten kann. Die Nachteile dieser Lösung, wie sie zuvor auf Seite 50 dargestellt wurden, gelten hier entsprechend.

Um diesen Nachteilen zu entgehen muss die Middleware sicherstellen, dass

- wenn Konsumenten für von einem zustandsbehafteten Verarbeiter erzeugte Nachrichten existieren, dieser Verarbeiter alle Nachrichten der benötigten Ausgangstypen erhält sowie
- wenn außerdem mehrere zustandsbehaftete Verarbeiter existieren, genau die für eine effiziente Kommunikation benötigten Verarbeiter alle entsprechenden Nachrichten erhalten und die anderen Verarbeiter inaktiv bleiben.

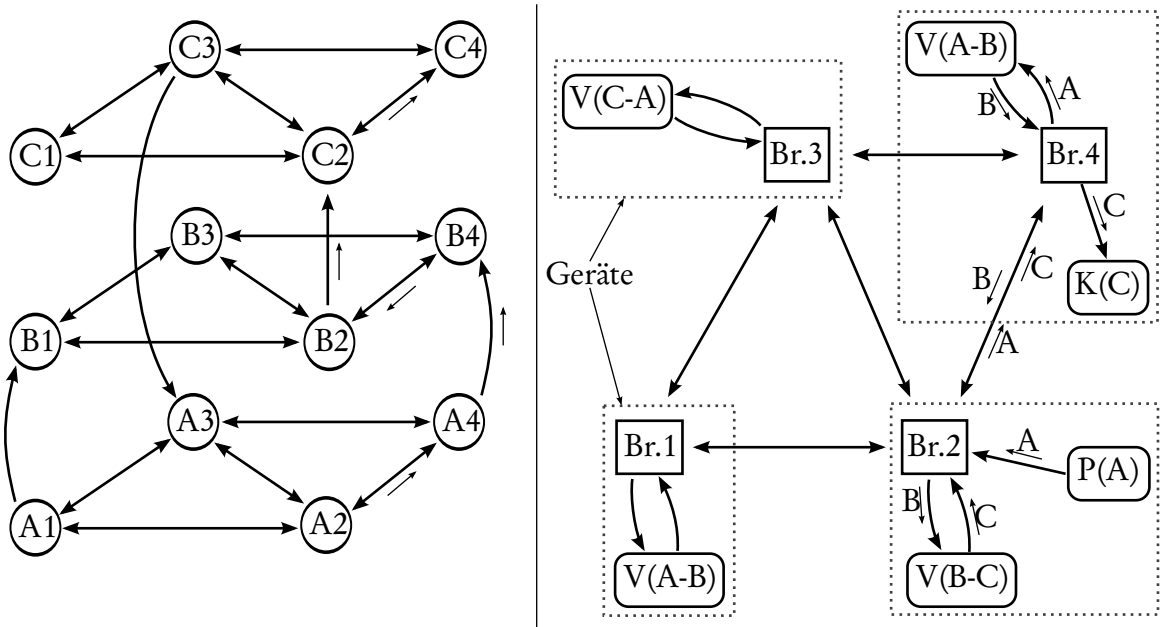
Die einfachste Möglichkeit, dies sicherzustellen ist, immer nur einen Pfad pro Senke zu benutzen. In den in Kapitel 5 vorgestellten Algorithmen wird dies durch die Freischaltung nur eines Pfades oder die Auswahl des einen optimalen Pfades unten allen freigeschalteten Pfaden sichergestellt.

Um die, in diesem Abschnitt vorgestellte Abbildung der Problemlösung auf eine Pfadsuche in einer zweidimensionalen Vermittlungstopologie gewährleisten zu können, wird im nächsten Abschnitt eine Systemarchitektur für eine entsprechende Middleware vorgestellt.

4.3. Vorstellung einer geeigneten Systemarchitektur

Aufgabe dieser Systemarchitektur ist das Verbinden von Produzenten und Konsumenten über geeignete Pfade durch die zuvor vorgestellte Vermittlungstopologie in einer heterogenen, veränderlichen Geräteumgebung. Dazu wird ein Broker pro Gerät eingesetzt, der die lokal vorhandenen Anwendungen in die Gesamttopologie einbindet. Netzwerkverbindungen zwischen Geräten stellen bidirektionale Nachbarschaftsbeziehungen zwischen Brokern dar und Nachrichtenverarbeitung in Verarbeitern werden als gerichtete Nachbarschaftsbeziehungen zwischen zwei Nachrichtentypen auf demselben Broker abgebildet.

Eine solche verteilte Systemarchitektur erlaubt die vollständige Abbildung des Routingproblems auf inhaltsbasierte Kommunikation in der Vermittlungstopologie. Eine adressbasierte Kommunikation zwischen Brokern und entfernten Anwendungen ist nicht mehr nötig. Dadurch ist eine flexible Reaktion auf Veränderungen in der Vermittlungstopologie möglich.



Vermittlungstopologie | Brokernetzwerk mit Anwendungen

Abbildung 4.5.:
Die linker Hand dargestellte Vermittlungstopologie entspricht dem auf der rechten Seite skizzierten Brokernetzwerk. Auf jedem Gerät läuft ein Broker (Br.x), der mit den lokalen Anwendungen kommuniziert. Dabei bedeutet P(A) - Produzent von Nachrichten des Typs A, K(C) - Konsument von Nachrichten des Typs C sowie V(x-y) - Verarbeiter von Nachrichten des Typs x in den Typ y.

4.3.1. Zusammenspiel zwischen den Brokern

Abbildung 4.5 zeigt ein Brokernetzwerk, dass die dargestellte Vermittlungstopologie reflektiert. Der in der Vermittlungstopologie gekennzeichnete Pfad findet sich im Brokernetzwerk wieder. Beim Produzenten P(A) beginnend wird die Nachricht an dessen Broker, Broker 2, übermittelt. Dieser übernimmt die Vermittlung der Nachricht an Stelle des Produzenten und fungiert somit als Nachrichtenquelle oder kurz **Quelle**. Für den Produzenten ist die Nachrichtenveröffentlichung damit abgeschlossen.

Die Nachricht muss nun entsprechend der Vermittlungstopologie an alle interessierten Konsumenten zugestellt werden. Der einzige interessierte Konsument im Beispiel, K(C), ist bei seinem Broker, Broker 4, angemeldet und wird durch diesen in der Vermittlungstopologie vertreten. Broker 4 fungiert demzufolge als Nachrichtensenke oder kurz **Senke** für die Nachrichten, die K(C) empfangen möchte.

Die Zustellung der Nachricht erfolgt zuerst über die Netzwerkverbindung zu Broker 4, dann wird sie von V(A-B) in den Nachrichtentyp B umgewandelt, zurück über die gleiche Netzwerkverbindung zu Broker 2 übertragen, dort von V(B-C) in den Nachrichtentyp C

umgewandelt, dann wiederum zu Broker 4 übertragen. Dieser benachrichtigt K(C) durch die Übermittlung der Nachricht.

Die Aufgaben des einzelnen Brokers sind in diesem Brokernetzwerk

- das Vertreten von Produzenten in der Rolle der Quelle,
- das Vertreten von Konsumenten in der Rolle der Senke,
- die Abstraktion von direkten Netzwerkverbindungen zu anderen Brokern als bidirektionale Nachbarschaftsbeziehungen in der Vermittlungstopologie sowie
- die Abstraktion von Verarbeitern als unidirektionale Nachbarschaftsbeziehungen in der Vermittlungstopologie.

Entsprechend dieser Anforderungen wird der Aufbau eines solchen Brokers im Folgenden beschrieben.

4.3.2. Aufbau des Brokers

Aufgrund der Anforderungen an einen Broker, bietet sich eine Architektur aus drei Schichten an. Die obere Schicht übernimmt die Abstraktion von den Anwendungen, die untere Schicht ist für die Abstraktion von den Netzwerkverbindungen zuständig. Die mittlere Schicht übernimmt das Routing von Nachrichten in der Vermittlungstopologie. Abbildung 4.6 zeigt einen solchen Broker. Die Aufgabe der einzelnen Schichten wird im Folgenden konkretisiert.

Anwendungsabstraktionsschicht

Diese Schicht hat drei Aufgaben, das Vertreten der Produzenten in der Rolle der Nachrichtenquelle, das Vertreten der Konsumenten in der Rolle der Nachrichtensenke und die Abbildung der Verarbeiter auf unidirektionale Nachbarschaftsbeziehungen.

Nachrichtenquelle Ankündigungen[3] müssen vom Produzenten übernommen und an die Vermittlungsschicht übergeben werden. Da im System ein inhaltsbasiertes Routing zum Einsatz kommt und somit der Inhalt einer Ankündigung deren Identität darstellt, müssen in der Ankündigung nicht-eindeutige Anteile separiert werden. Außerdem muss eine Ankündigung später in Abonnements und Veröffentlichungen leicht referenzierbar sein.

Beide Anforderungen lassen sich sehr effizient durch die Generierung eines Hashwertes über die eindeutigen Anteile einer Ankündigung realisieren, der dann später zur platzsparenden, eindeutigen Referenzierung eingesetzt werden kann. Die Generierung eines solchen Hashwertes, der **Ankündigungs-ID (AID)**, ist Aufgabe dieser Schicht.

Spätere Veröffentlichungen des Produzenten müssen seiner Ankündigung zugeordnet und entsprechend mit der AID versehen und an die Vermittlungsschicht übergeben werden.

[3] Eine Diskussion zum Einsatz von Ankündigungen findet sich im folgenden Abschnitt 4.4

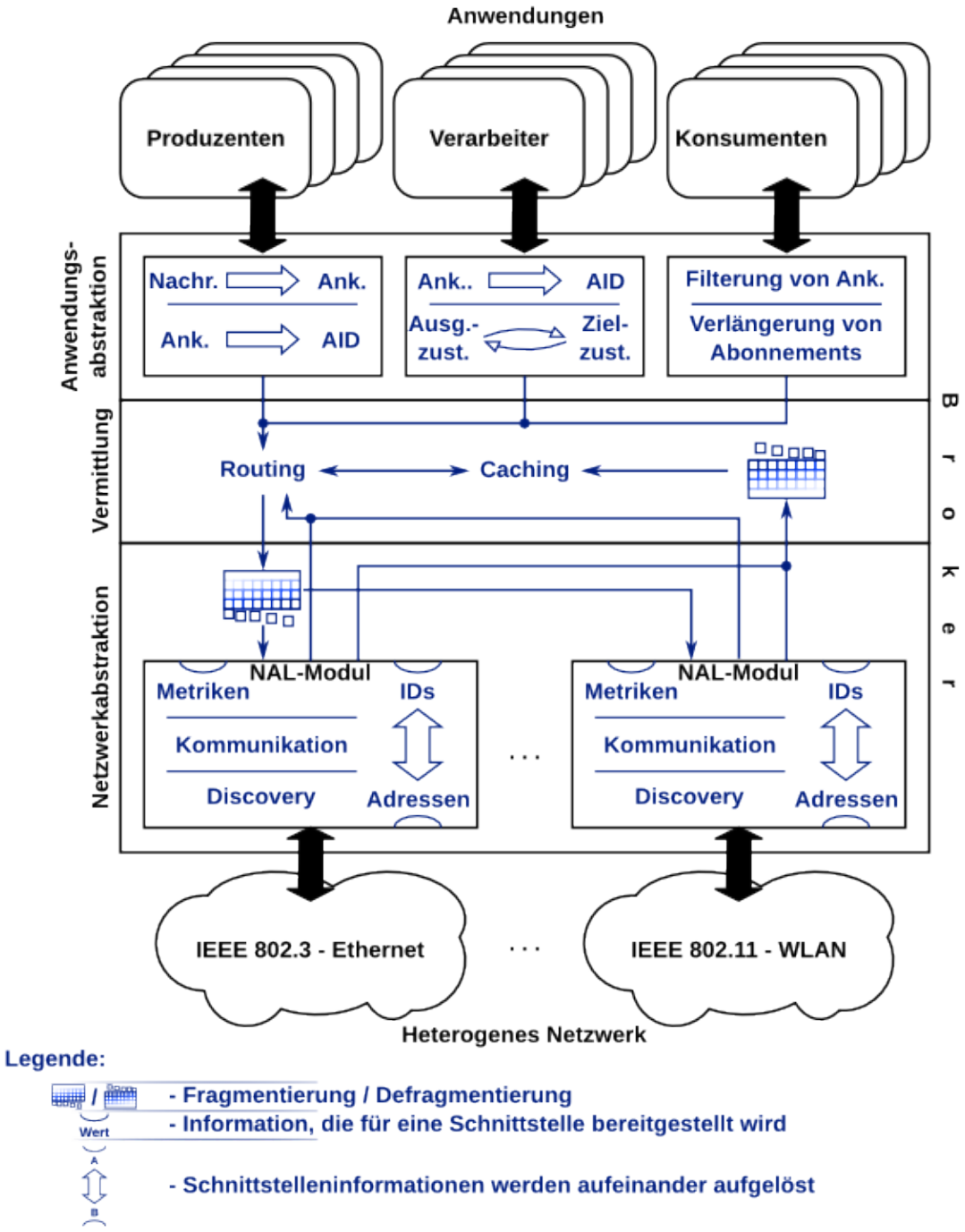


Abbildung 4.6.: Konzeptioneller Aufbau eines Brokers.

Nachrichtensenke Konsumenten registrieren sich am Broker mit ihrem Interesse und gegebenenfalls der Option, basierend auf passenden Ankündigungen zu entscheiden, ihr Interesse zu verfeinern. Die Anwendungsabstraktionsschicht muss demzufolge verfügbare Ankündigungen mit dem Interesse des Konsumenten abgleichen und bei Erfolg, gegebenenfalls nach Rückfrage beim Konsumenten, Abonnements basierend auf Ankündigungen generieren und an die Vermittlungsschicht übergeben.

Um einen ununterbrochenen Fluss von Nachrichten zum Konsumenten zu gewährleisten, müssen bei eventuellen Verlängerungen der Gültigkeit von Ankündigungen so lange automatisch Abonnements generiert und abgesetzt werden bis der Konsument sich abmeldet oder sein Interesse ändert.

Abstraktion vom Verarbeiter Verarbeiter registrieren sich am Broker ebenfalls mit ihrem Interesse. Dieses spiegelt die Nachrichtentypen wieder, die sie zu verarbeiten in der Lage sind. Entsprechend muss die Anwendungsabstraktionsschicht Ankündigungen, die auf dieses Interesse passen, an den Verarbeiter weiterreichen. Entsprechend der erhaltenen Ankündigungen generiert der Verarbeiter nun neue Ankündigungen, die einem anderen Nachrichtentyp entsprechen. Damit signalisiert er dem Broker, dass er in der Lage ist, Nachrichten von den Ausgangstypen in diesen Nachrichtentyp umzuwandeln.

Vom Verarbeiter generierte Ankündigungen müssen genau wie in der Rolle der Nachrichtenquelle mit einer AID versehen und an die Vermittlungsschicht übergeben werden. Die AIDs der Ausgangsnachrichten, aus denen die entsprechende Ankündigung generiert wurde, müssen vorrätig gehalten werden, um später einen Rückkanal für Kontrollnachrichten über die ansonsten unidirektionale Verarbeitungsverbindung realisieren zu können.

Abonnements für die vom Verarbeiter generierten Ankündigungen müssen über diesen Rückkanal später in Abonnements für die Ausgangsnachrichten übersetzt und an die Vermittlungsschicht übergeben werden. Entsprechend müssen, auf diese Abonnements hin empfangene Nachrichten an den Verarbeiter zur Verarbeitung übergeben werden und die resultierenden Nachrichten mit der neuen AID versehen und an den Vermittler zur Weiterleitung übergeben werden.

Da aus Sicht der Vermittlungsschicht die unidirektionale Verbindung zwischen den Ausgangstypen und dem erzeugten Nachrichtentyp genau wie Nachbarschaftsbeziehungen gewichtet sein muss, muss die Anwendungsabstraktionsschicht entsprechend der Angaben des Verarbeiters für diesen Verarbeitungsschritt einen Kostenwert zur Verfügung stellen. Wie dieser Kostenwert aussieht und welche Metriken hier zum Einsatz kommen können, wird im Abschnitt 4.5 diskutiert.

Netzwerkabstraktionsschicht

Aufgabe der Netzwerkabstraktionsschicht ist die Abstraktion von den Netzwerkverbindungen zu benachbarten Brokern. Damit sind folgende Anforderungen verbunden:

Neighbor Discovery Neue Nachbarn müssen gefunden werden und eine lokal gültige Adresse muss für jeden Nachbarn generiert werden. Die Verfügbarkeit eines neuen Nachbarn muss an die Vermittlungsschicht übermittelt werden.

Kommunikation Ankündigungen, Nachrichten und Kontrollnachrichten müssen sicher zum Nachbarn übertragen werden. Ist dies nicht mehr möglich, muss das Verschwinden des Nachbarn an die Vermittlungsschicht weitergegeben werden.

Verbindungsbewertung Die Verbindung zum Nachbarn muss für die Vermittlungsschicht mit einem Kostenwert versehen werden. Dieses muss die Kosten der Datenübertragung an den Nachbarn reflektieren und entsprechend von der Netzwerkabstraktionsschicht gemessen und aktuell gehalten werden.

Ankündigungen, Abonnements und Nachrichten von der Vermittlungsschicht werden an den durch die lokale Adresse definierten Nachbarbroker übermittelt. Sollten Nachrichten für das genutzte Kommunikationsprotokoll zu groß sein, muss die Netzwerkabstraktionsschicht für die Fragmentierung der Nachrichten sorgen. Ankündigungen, Abonnements, Nachrichten und Nachrichtenfragmente von einem benachbarten Broker werden mit dessen lokaler Adresse an die Vermittlungsschicht übergeben.

Vermittlungsschicht

Durch die Anwendungs- und die Netzwerkabstraktionsschicht werden die gewichteten bidirektionalen Verbindungen zu den benachbarten Brokern und die gewichteten unidirektionalen Verbindungen zu den anderen Nachrichtentypen bereitgestellt. Damit sind in der Vermittlungsschicht alle Nachbarn in der Vermittlungstopologie bekannt. Außerdem werden von der Anwendungsabstraktionsschicht die Quellen und Senken bereitgestellt. Auf Basis dieser lokalen Informationen müssen mit Hilfe eines geeigneten Routingalgorithmus effiziente Pfade von den Quellen zu den Senken gefunden werden. Die Art von Routingalgorithmen, die hierfür grundsätzlich in Frage kommen, werden in Abschnitt 4.4 diskutiert. Zwei geeignete Routingalgorithmen werden in dieser Arbeit in Kapitel 5 vorgestellt und in Kapitel 6 untersucht.

Eine weitere Aufgabe der Vermittlungsschicht ist das Defragmentieren von Nachrichtenfragmenten von der Netzwerkabstraktionsschicht. Dies ist nötig, um Nachrichten an die Anwendungsabstraktionsschicht übergeben zu können. Allerdings sollte für die Weiterleitung an die Netzwerkabstraktionsschicht aus Gründen der Effizienz direkt mit den Nachrichtenfragmenten gearbeitete werden.

Weiterhin ist die Zwischenspeicherung von Ankündigungen und Nachrichten in der Vermittlungsschicht angesiedelt. Dies ist notwendig um Doppeltübertragungen zu vermeiden sowie eine gegenwärtige, sowie eingeschränkt zukünftige zeitliche Entkopplung von Konsumenten und Produzenten gewährleisten zu können.

4.4. Routing in einer Vermittlungstopologie

In diesem Abschnitt werden die in Abschnitt 2.1.2 vorgestellten inhaltsbasierten Routingansätze aus Publish/Subscribe-Systemen hinsichtlich ihrer Eignung für die Pfadsuche in einer Vermittlungstopologie diskutiert. Die zu untersuchenden Ansätze sind das Flooding, bei dem jede Nachricht an jeden Broker übermittelt wird, das inhaltsbasierte Routing ohne Ankündigungen, bei dem das Interesse in der gesamten Topologie verteilt wird sowie das inhaltsbasierte Routing mit Ankündigungen, bei dem Ankündigungen der Produzenten im gesamten System verteilt werden. Die einzelnen Ansätze werden nachfolgend in dieser Reihenfolge diskutiert.

4.4.1. Auf Flooding basierende Ansätze

Der einfachste Ansatz ist das Verteilen aller Nachrichten in der gesamten Vermittlungstopologie. Das bedeutet, dass jeder Broker alle veröffentlichten Nachrichten in allen möglichen Nachrichtentypen erhält und lokal entscheiden kann, ob diese Nachricht für einen Konsumenten benötigt oder verworfen wird. Durch die Zustellung von Nachrichten an Broker, die nicht auf einem Pfad zwischen Quelle und Senke liegen, ist dieses Verfahren höchst ineffizient und kann sehr schnell zur Überlastung von Netzwerkverbindungen und Verarbeitern führen.

Dieser Ansatz ist sinnvoll, wenn die zu übertragenden Informationen aus einer einzelnen oder sehr wenigen sehr kleinen Nachrichten bestehen so dass der zu erwartende Overhead an Kontrollnachrichten zur Ermittlung geeigneter Pfade größer wäre als die eigentlich zu übertragende Datenmenge. Außerdem kann Flooding sinnvoll sein, wenn das Interesse an Nachrichten so groß ist, dass kaum Broker existieren, die nicht auf einem Pfad zwischen der Quelle und einer Senke liegen. Hier wäre allerdings ein optimierter Flooding-Algorithmus ohne zu großen Overhead durch Implosion[4] wünschenswert.

In Szenarien mit wenigen Interessenten pro Nachrichtenquelle und vielen zu veröffentlichenden Nachrichten ist die Nutzung von Flooding nicht sinnvoll. Aufgrund seiner Flexibilität und Robustheit ist es in solchen Szenarien aber sinnvoll, Flooding als Referenzverfahren zu benutzen. Durch Flooding kann garantiert werden, dass eine Nachricht an alle Interessenten, die von einer Quelle aus zu erreichen sind, zugestellt wird. Dabei muss allerdings sichergestellt werden, dass das Netzwerk durch das Flooding nicht überlastet wird. Es eignet sich also nur als Vergleichsgrundlage für Szenarien mit geringer Nachrichtengröße und -frequenz.

[4] siehe Grundlagen zu Flooding in Abschnitt 2.2.1

4.4.2. Routing ohne Ankündigungen

Voraussetzung für ein inhaltsbasiertes Routingverfahren ohne Ankündigungen ist die Verteilung des Interesses an alle Ecken der Vermittlungstopologie. Daraus entstehen dann für jede Ecke Routingtabellen, die im Anschluss für das gezielte Weiterleiten von Nachrichten in Richtung der Senken genutzt werden.

Bei diesem Ansatz enthält eine Routingtabelle für eine Ecke der Vermittlungstopologie für jedes Interesse einer von dieser Ecke aus erreichbaren Senke einen Distanzvektor zu dieser und allen anderen erreichbaren Senken, die dieses Interesse geäußert haben. Dadurch können diese Routingtabellen sehr schnell sehr groß werden. Außerdem repräsentiert jeder Broker $|T|$ Ecken der Vermittlungstopologie[5], und muss somit $|T|$ dieser Routingtabellen verwalten.

Ein weiteres Problem dieses Ansatzes liegt in der Natur zustandsbehafteter Nachrichtenverarbeitung. Die Ausgangsannahme für ein effizientes Routing ist es, dass Nachrichten nur dann versandt werden, wenn auch Konsumenten für diese Nachrichten vorhanden sind. Angenommen ein Verarbeiter benötigt Nachrichten vom Typ A und vom Typ B um daraus Nachrichten des Typs C zu erzeugen und dieser Sachverhalt ist dem Broker, an dem dieser Verarbeiter registriert ist, auch bekannt. Für Nachrichten des Typs C ist ein Konsument vorhanden. Nun wird das Interesse des Konsumenten in der Vermittlungstopologie verbreitet. Der Broker des Verarbeiters muss nun also das Interesse des Typs C in Interesse an den Typen A und B umwandeln und dieses weiterverbreiten. Gelangt dieses zu einer Quelle für den Typ A kann diese nicht entscheiden, ob ein Absenden der Nachrichten sinnvoll ist, weil dies davon abhängt, ob auch eine Quelle für Nachrichten des Typs B vorhanden ist. Die Abstimmung zwischen diesen beiden Quellen hätte zusätzlichen Overhead zur Folge, der zu Lasten der Effizienz, der räumlichen Trennung zwischen Anwendungen und vermutlich der Robustheit des Routings ginge.

4.4.3. Routing mit Ankündigungen

Beim inhaltsbasierten Routing mit Ankündigungen werden von den Quellen aus Ankündigungen in der Vermittlungstopologie verbreitet. Sind bei einem Verarbeiter Ankündigungen für alle notwendigen Ausgangsnachrichtentypen eingegangen, wird eine Ankündigung für den resultierenden Nachrichtentyp generiert und in der Vermittlungstopologie verbreitet. Die daraus entstehenden Routingtabellen werden ähnlich groß wie beim vorhergehenden Ansatz. Allerdings werden sie nur zum Routing von Abonnements benutzt.

Aus den Abonnements entstehen sehr kleine Routingtabellen für die einzelnen Ecken der Vermittlungstopologie, die nur die Abonnements enthalten, für die auch tatsächlich Nachrichten im System über die jeweiligen Ecke verbreitet werden müssen. Dieser Vorgang ist immer dann effizienter in Bezug auf die Suche nach geeigneten Einträgen in der entspre-

[5] $|T|$ ist die Anzahl der in der Vermittlungstopologie erreichbaren Nachrichtentypen. Vgl. Abschnitt 4.2.1

Routingansatz	Vorteile	Nachteile
Flooding	+ hohe Flexibilität + hohe Robustheit	– geringe Effizienz
Ohne Ankdg.		– Probleme mit zustandsbehafteter Nachrichtenverarbeitung
Mit Ankdg.	+ Registrierung mit Rückfrage + transparente Ankündigungen	

Tabelle 4.2.:
Zusammenfassung der Vor- und Nachteile der diskutierten Routingansätze zur Verbreitung von Nachrichten in der Vermittlungstopologie.

chenden Routingtabelle als der Ansatz ohne Ankündigungen wenn mehr Nachrichten oder Nachrichtenfragmente als Abonnements zu verteilen sind.

Der Fall der zustandsbehafteten Datenverarbeitung stellt für diesen Ansatz kein Problem dar, wenn nur ein Pfad pro Abonnement freigeschaltet wird. In diesem Fall befindet sich ein Verarbeiter entweder auf dem Pfad und erhält somit alle notwendigen Ausgangsnachrichten oder er befindet sich nicht auf dem Pfad und ist somit auch nicht aktiv. Benutzt ein Algorithmus Mehrwegeausbreitung oder werden Teilpfade im Nachhinein verändert, muss die Möglichkeit zustandsbehafteter Datenverarbeitung berücksichtigt werden. Eine mögliche Lösung wird im Abschnitt 5.3.2 auf Seite 107 diskutiert.

Bei Nutzung dieses Ansatzes kann ein Verarbeiter aus empfangenen Ankündigungen verschiedener Nachrichtentypen spontan entscheiden, welchen Nachrichtentyp er daraus zu generieren in der Lage ist, indem er eine neue Ankündigung generiert und mit einem Verweis auf die Ausgangstypen an die Middleware überträgt. Dieses Vorgehen birgt einen wesentlichen Vorteil in sich. Der Inhalt der Ankündigungen kann für die Middleware vollständig transparent bleiben und allein durch die Anwendungen bestimmt werden.

Weiterhin wird erst durch die Verbreitung von Ankündigungen im gesamten System die Registrierung mit Rückfrage auf Seite des Konsumenten ermöglicht. Diese erlaubt dem Konsumenten, aus verfügbaren Informationen auszuwählen und erst anschließend entsprechende Nachrichten zu abonnieren. Dies kann insbesondere dann sinnvoll sein, wenn der Konsument eine Anwendung mit direkter Schnittstelle zum Benutzer ist.

Tabelle 4.2 fasst die Ergebnisse dieses Abschnittes zusammen.

4.5. Bewertung gefundener Pfade

Für ein inhaltsbasiertes Routing zwischen Nachrichtenquellen und Nachrichtensenken in einer Vermittlungstopologie müssen Pfade in der Topologie gefunden, bewertet und mitein-

ander verglichen werden. Dies erfolgt auf Basis der Kantengewichte die von einer geeigneten Routingmetrik zur Verfügung gestellt werden müssen. An eine solche Metrik werden durch die Beschaffenheit der Vermittlungstopologie besondere Anforderungen gestellt, die über die Anforderungen an Routingmetriken für Netzwerkprotokolle in spontanen Umgebungen (CAMPISTA et al., 2008) hinausgehen:

- Sowohl Kanten, die Nachrichtenversand repräsentieren, als auch Kanten, die Nachrichtenverarbeitung repräsentieren, müssen in vergleichbarer Weise gewichtet werden.
- Die Kantengewichte sollten den Aufwand für die Übertragung einer einzelnen Nachricht repräsentieren. Nachrichten können dabei größer sein als ein einzelnes Netzwerkpaket, was eine Übertragung in Fragmenten nötig macht. Außerdem können Nachrichten ihre Größe auf dem Weg von der Quelle zur Senke durch die Nachrichtenverarbeitung verändern.
- Bei der Planung der Pfade in der Phase der Verbreitung der Ankündigungen und der Abonnements ist die Größe einer Nachricht noch nicht bekannt. Hier ist ein Mechanismus zur Abschätzung der Nachrichtengröße nötig.

Die Grundlage für eine geeignete Metrik bildet im Folgenden die Zeit. Im Fall der Nachrichtenübermittlung ist dies die zu erwartende Übertragungszeit und im Fall der Nachrichtenverarbeitung die zu erwartende Verarbeitungszeit. In den Abstraktionsschichten wird diese für eine gegebene Nachrichtengröße ermittelt. Im Folgenden wird für die beiden Schichten eine Möglichkeit zum Finden eines korrekten Kantengewichtes in Abhängigkeit von der Nachrichtengröße erklärt. Den Abschluss bildet die Aggregation dieser Kantengewichte zu Pfadgewichten in der Vermittlungsschicht.

4.5.1. Netzwerkabstraktionsschicht

Aufgabe der Netzwerkabstraktionsschicht ist die Bewertung der Nachrichtenübertragung auf den Netzwerkverbindungen zu benachbarten Brokern durch ein Kantengewicht. Diese Gewichtung ist unabhängig vom übertragenen Nachrichtentyp, also in der Vermittlungstopologie für jede vertikal verteilte Version der gleichen horizontalen Kante identisch. Um somit für eine Netzwerkverbindung, also eine Kante $b \in B$ und eine gegebene Nachrichtengröße s eine Gewichtung der Kante errechnen zu können, wird die prognostizierte Übertragungszeit errechnet. Diese setzt sich aus der konstanten Ausbreitungsverzögerung l_b und der paketgrößenabhängigen Übertragungsverzögerung $e_b(s)$ zusammen (PETERSON und DAVIE, 2007, S.42). Die Aufreihungsverzögerung spielt keine Rolle, da eine direkte Verbindung zwischen zwei benachbarten Geräten betrachtet wird. Da bei vielen Kommunikationsverfahren eine erfolglose Übertragung wiederholt wird, ist die Einbeziehung der zu erwartenden Anzahl notwendiger Übertragungen a_b sinnvoll.

Da die Ermittlung dieser Werte stark von der verwendeten Kommunikationstechnologie abhängig ist, werden hier nur grundsätzliche Möglichkeiten aufgezeigt. Zur Messung von a_b

wird die Anzahl der nötigen Übertragungsversuche je übermitteltem Netzwerkpaket überwacht und ein gleitender Mittelwert gebildet. Dieser Ansatz basiert grob auf der Berechnung der ETX-Metrik in DE COUTO et al. (2003).

Um l_b und $e_b(s)$ zu ermitteln ist es sinnvoll, regelmäßig kleine Testpakete zu versenden und deren Paketumlaufzeit zu messen. Als zweiter Messwert ist die Paketumlaufzeit großer Datenpakete nötig. Dies können, wenn die Kommunikationsverbindung in Benutzung ist, tatsächliche Datenpakete, ansonsten große Testpakete sein. Unter Kenntnis der Größe der Testpakete und der jeweiligen Antwortpakete und der charakteristischen Eigenschaften der jeweiligen Übertragungstechnik ist es möglich, l_b und $e_b(s)$ durch die Lösung eines Gleichungssystems aus zwei Gleichungen mit zwei Unbekannten zu errechnen. Um die Wirkung von Messfehlern zu verringern, ist es sinnvoll für die Berechnung der zu erwartenden Übertragungszeit, auch hier gleitende Durchschnitte der ermittelten Werte zu verwenden. Dieser Ansatz basiert grob auf der Berechnung der ETT-Metrik in DRAVES, PADHYE und ZILL (2004).

Die Nutzung der zu erwartenden Übertragungszeit hat den Vorteil der Vergleichbarkeit über verschiedene Kommunikationstechnologien hinweg. Um andere Unterschiede zwischen diesen Technologien, wie zum Beispiel die Anfälligkeit für Störungen, den Energieverbrauch oder entstehende finanzielle Kosten auszugleichen, ist eine Ergänzung um einen Gewichtungsfaktor w_b einer jeden Verbindung sinnvoll. Mit diesem kann die Netzwerkabstraktionsschicht verschiedene Kommunikationstechnologien unterschiedlich gewichten um deren Nutzung dadurch attraktiver oder weniger attraktiv zu machen.

In der Berechnung der gewichteten größenabhängigen zu erwartenden Übertragungszeit einer Nachricht über eine Netzwerkverbindung $c_b(s)$ ist anschließend noch die Fragmentierung zu beachten, wobei die MTU der verwendeten Übertragungstechnologie durch m_b ausgedrückt wird. Daraus ergibt sich die folgende Formel:

$$c_b(s) = w_b a_b (l_b + \lceil \frac{s}{m_b} \rceil e_b(s)) \tag{4.1}$$

4.5.2. Anwendungsabstraktionsschicht

Aufgabe der Anwendungsabstraktionsschicht ist die größenabhängige Bewertung des Aufwands der Nachrichtenverarbeitung. Diese unterscheidet sich nicht nur zwischen verschiedenen Verarbeitern, sondern auch zwischen den verschiedenen Instanzen desselben Verarbeiters auf unterschiedlichen Geräten. Sie wird also für die konkrete Instanz ι einer Verarbeiters in der Vermittlungstopologie berechnet. Auch hier dient die zu erwartende Bearbeitungszeit als Basis. Dazu wird von den Verarbeitern eine Grundlatenz l_ι und ein größenabhängiger Anteil $e_\iota(s)$ geliefert. Die Anwendungsabstraktionsschicht ergänzt eine Gewichtung w_ι um die zu erwartende Belastung von Verarbeitern oder die parallele Anbindung mehrerer Ver-

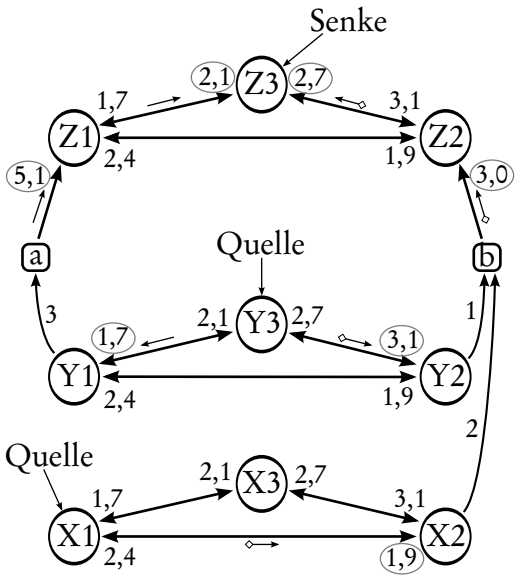


Abbildung 4.7:
Die Abbildung zeigt eine beispielhafte Vermittlungstopologie mit annotierten Kantengewichten für eine bestimmte zu übertragende Nachricht an den Pfeilspitzen, die die entsprechende gerichtete Kante repräsentieren. Markiert sind zwei Pfade zur Senke Z3. Der erste (→) beginnt bei Quelle Y3 und der zweite (◇→) bei den Quellen X1 und Y3. Die Pfadgewichte setzen sich wie folgt aus den markierten Kantengewichten zusammen: Für den ersten Pfad ergibt sich $1,7 \cdot 3 + 5,1 + 2,1 = 12,3$ sowie für den zweiten Pfad $3,1 + 1,9 \cdot 2 + 3,0 + 2,7 = 12,6$.

arbeiter ausgleichen zu können. Damit errechnet sich die gewichtete größenabhängige zu erwartende Verarbeitungszeit $c_\iota(s)$ einer Nachricht durch einen Verarbeiter als

$$c_\iota(s) = w_\iota(e_\iota(s) + l_\iota) \tag{4.2}$$

4.5.3. Vermittlungsschicht

In der Vermittlungsschicht müssen die von den Abstraktionsschichten gelieferten Kantengewichte zu Pfadgewichten aggregiert werden. Außerdem müssen die Probleme der möglichen Größenveränderung der Nachrichten durch Verarbeitung, der zum Zeitpunkt der Pfadsuche unbekannten Nachrichtengröße sowie der möglichen Aggregation mehrerer Nachrichten zu einer einzigen bei der zustandsbehafteten Datenverarbeitung gelöst werden.

Kombination der Kantengewichte

Da die Kantengewichte den Aufwand für die Nutzung der Kanten in Form von Zeit repräsentieren, werden zur Bildung von Pfadgewichten die Gewichte der beteiligten Kanten addiert. Dabei werden für einen zustandsbehafteten Verarbeiter, der mit mehreren Ausgangsnachrichtentypen verbunden ist, die Gewichte aller eingehenden Pfade, jeweils mit der an der entsprechenden Kante attributierten Anzahl multipliziert, addiert und anschließend das Gewicht der ausgehenden Kante addiert. Abbildung 4.7 veranschaulicht diesen Vorgang.

Die Verknüpfung der Pfadgewichte bei der Nachrichtenaggregation über deren Summe hat die positive Eigenschaft, dass eine frühere Aggregation mehrerer Nachrichten unter gleichen Rahmenbedingungen auf dem Weg von den Quellen zur Senke zu niedrigeren Pfad-

gewichten des gesamten Pfades führt und somit von einem Routingalgorithmus bevorzugt wird.

Umgang mit der Nachrichtengröße

Die Nutzung einer Metrik, die bei der Berechnung der Kantengewichte die Größe der zu übertragenden Nachricht mit einbezieht, hat einen wesentlichen Vorteil. Bei der Verfügbarkeit mehrerer geeigneter Verarbeiter werden Pfade, auf denen Nachrichten frühzeitig durch Verarbeitung verkleinert respektive spät durch Nachrichtenverarbeitung vergrößert werden, anderen Pfaden gegenüber bevorzugt.

Dies ist aber nur möglich, wenn die zu übertragende Nachrichtengröße bei der Auswahl eines geeigneten Pfades bereits bekannt ist oder sinnvoll geschätzt werden kann. Dazu wird folgendes Vorgehen vorgeschlagen: Ein Nachrichtenproduzent liefert bei der Anmeldung neben der Ankündigung für die von ihm zu veröffentlichenden Nachrichten die zu erwartende Größe s_0 der Nachrichten. Basierend auf dieser Größe werden die Kantengewichte für die Verbreitung von Nachrichten dieses Produzenten berechnet solange sie nicht verarbeitet wurden.

Läuft eine Ankündigung durch einen Verarbeiter, so liefert dieser neben der Ankündigung für den neuen Nachrichtentyp auch einen Größenveränderungsfaktor f_ι. Die Größe der Nachricht wird nach der Verarbeitung entsprechend angepasst: $s_\iota = f_\iota s$. Bei der Verarbeitung mehrerer Nachrichten bezieht sich f_ι auf die Summe der Größen der Ausgangsnachrichten. Die Gewichte der folgenden Kanten zur Nachrichtenverbreitung werden nun bis zum nächsten Verarbeiter mittels s_ι berechnet.

4.6. Anforderungen an die Anwendungen

Ziel dieser Arbeit ist die möglichst vollständige Entkopplung der Anwendungen voneinander, von der Middleware sowie von der Netzwerk- und Verarbeitungstopologie. Trotzdem sind bestimmte grundsätzliche Eigenschaften der Anwendungen für das Routing in der Middleware von Bedeutung. Dazu werden im ersten Abschnitt Anwendungsklassen definiert und die Zuordnung von Anwendungen zu diesen Klassen sowie deren Auswirkung auf das Routing diskutiert. Die wesentlichen Anforderungen an Anwendungen aus Sicht der Schnittstellen zwischen Anwendung und Middleware sowie der kommunizierten Inhalte ist Thema des darauf folgenden Abschnittes.

4.6.1. Anwendungsklassen

Im Folgenden wird eine Einteilung von Anwendungen anhand von zwei wesentlichen Eigenschaften der von ihnen veröffentlichten Nachrichten vorgeschlagen und diskutiert. Die

betreffenden Eigenschaften dabei sind die Größe sowie die Anzahl der zu veröffentlichenden Nachrichten.

Anwendungseigenschaften

Für die Anzahl der Nachrichten wird zwischen der Veröffentlichung einer einzelnen Nachricht und der Veröffentlichung mehreren Nachrichten, also eines Nachrichtenstromes, unterschieden.

In Bezug auf die Nachrichtengröße wird zwischen kleinen Nachrichten und großen Nachrichten unterschieden. Nachrichten sind klein, wenn sie in die Nutzlast einer Ankündigung passen und somit in der Verarbeitungstopologie problemlos per Flooding an alle Ecken verteilt werden können.

Die Maximale Größe einer Ankündigung wurde bisher noch nicht festgelegt. Das Kriterium der einfachen Verteilbarkeit ist hier von wesentlicher Bedeutung. Es ist somit sinnvoll, sie so zu wählen, dass eine Ankündigung möglichst selten durch Fragmentierung in mehrere zu verteilende Pakete zerlegt werden muss. Da auf Anwendungsebene aber weder die MTU lokaler, noch aller in der Vermittlungstopologie vorhandenen Technologien bekannt ist, kann hier ein beliebiger sinnvoller Wert gewählt werden. Wenn man die verbreitetesten Kommunikationstechnologien auswählt, und deren MTUs vergleicht, so ergibt sich eine sinnvolle maximale Größe für Ankündigungen von ca. 1,4kiB. Sie liegt unterhalb der MTU von Ethernet und WLAN. Andere Technologien wie zum Beispiel ZigBee verwenden eine MTU, die weit niedriger und damit als obere Grenze von Ankündigungen mit hoher Wahrscheinlichkeit nicht geeignet ist.

Aus der Kombination der beiden Anwendungseigenschaften ergibt sich eine Matrix mit vier Anwendungsklassen. Diese ist in Abbildung 4.8 dargestellt und für jede Klasse mit Beispielen versehen.

Aus Sicht der Anwendungen sind diese Eigenschaften bekannt und können bereits bei der Anmeldung an der Middleware bekannt gegeben werden. Für das Routing der Nachricht bzw. Nachrichten durch die Middleware haben diese Eigenschaft einen wesentlichen Einfluss wie weiter unten ausführlich diskutiert wird.

Einfluss auf die Dienstgüte

Aus der Einteilung in die vier zuvor beschriebenen Anwendungsklassen ergeben sich verschiedene Implikationen auf verschiedene Dienstgütemerkmale. In vielen Fällen kann es für eine Anwendung daher sinnvoll sein, durch die entsprechende Aufbereitung ihrer Daten gezielt eine geeignete Klasse auszuwählen.

Eine große Nachricht muss für die Übertragung zwischen Quellen und Senken mit hoher Wahrscheinlichkeit fragmentiert werden. Dies hat zwei wesentliche Implikationen:

Für jeden auf dem Pfad von Quelle zur Senke notwendigen Verarbeitungsschritt muss die gesamte Nachricht vorliegen. Dazu muss sie defragmentiert werden. Es entsteht also zur

	Einzelnachricht	Nachrichtenstrom	
klein	(1) Beispiele: • Steuerungsbefehl • Kontaktinformation	(2) Beispiele: • Einfache Sensordaten	klein
groß	(3) Beispiele: • Bild • Dokument • Visitenkarte	(4) Beispiele: • komplexe Sensordaten • Multimediadaten	groß
	Einzelnachricht	Nachrichtenstrom	

Abbildung 4.8.:
Untergliederung der Anwendungen nach der Größe und Anzahl der zu veröffentlichenden Nachrichten

durch die Übertragung und Verarbeitung bedingten Verzögerung noch eine Aufreihungsverzögerung entsprechend der Anzahl der Verarbeitungsschritte.

Aus Sicht der Middleware bildet eine Nachricht eine Einheit, die zu einem Zeitpunkte veröffentlicht, dann fragmentiert, übertragen, defragmentiert und zu einem Zeitpunkt zugestellt wird. Eine Defragmentierung kann nur erfolgreich sein, wenn alle dazu notwendigen Fragmente erfolgreich übertragen wurden. Die Middleware muss dies durch Zwischenspeicherung, Fehlerbehandlung und bedarfsgerechte erneute Übertragung sicherstellen.

Bei der Übertragung von Nachrichtenströmen wird aus Sicht der Middleware davon ausgegangen, dass jede Nachricht in sich abgeschlossen und die Anwendung am Empfang möglichst vieler, möglichst aktueller Nachrichten interessiert ist. Dabei wird der Verlust einzelner Nachrichten durchaus toleriert wenn ein Teil der Verbindung oder ein Konsument zeitweise nicht verfügbar ist.

Ein drittes Merkmal von Nachrichten, dass vor allem Einfluss auf die inhaltliche Entkopplung hat, ist die logische Einheit. Eine Nachricht sollte in sich verständlich und vollständig sein. Auf diese Weise ist es für eventuell notwendige Verarbeitungsschritte eher möglich, zustandslose Verarbeiter einzusetzen. Diese sind flexibler einsetzbar und jederzeit ohne Nebenwirkungen austauschbar.

Einfluss der Anwendungsklassen auf das Routing

Die Anwendungsklassen haben verschiedene Implikationen auf das Routing von Nachrichten zwischen Anwendungen:

Anwendungsklasse 1 Hier ist ein Routing an sich nicht sinnvoll. Die Etablierung eines Pfades zwischen Quelle und Senke benötigt vermeidbaren Overhead, da dieser Pfad zu keinem späteren Zeitpunkt mehr benutzt wird. Ein einfaches Flooding der Nachricht in der Vermittlungstopologie ist einem inhaltlichen Routing mit Ankündigung vorzuziehen. Lediglich zur Absicherung der zeitlichen Entkopplung zwischen Quelle und Senke ist die Zwischenspeicherung der Nachricht für eine kurze Zeit sinnvoll.

Anwendungsklasse 2 Abhängig von der Frequenz der Nachrichtenveröffentlichung, der tatsächlichen Nachrichtengröße, der Anzahl und Fluktuation der Senken sowie der Veränderlichkeit der Netzwerktopologie kann ein Fluten der Nachrichten sinnvoll sein. In den meisten Fällen wird vermutlich ein inhaltsbasiertes Routing mit der Etablierung und Nutzung von Pfaden überlegen sein.

In dieser Anwendungsklasse bietet sich der Vergleich mit Flooding zur Evaluierung der Performance von inhaltsbasierten Routingalgorithmen an, da bei kleinen Nachrichtengrößen und einer nicht zu hohen Veröffentlichungsfrequenz nicht mit einer Überlastung des Netzwerkes durch das Flooding zu rechnen ist. Eine mit diesen Ansätzen vergleichbare Robustheit und Effektivität bei besserer Effizienz des inhaltsbasierten Routings ist hier wünschenswert.

Anwendungsklassen 3 und 4 Aufgrund der großen Nachrichtengröße und der Notwendigkeit des Empfangs aller Fragmente je Nachricht ist ein inhaltsbasierter Routingansatz in jedem Fall überlegen. Ein Fluten entsprechend großer Nachrichten führt hier sehr schnell zur Überlastung der Netzwerkinfrastruktur.

4.6.2. Besonderheiten der Schnittstellen

Dieser Abschnitt gibt einen Überblick über die besonderen Anforderungen der Anwendungsschnittstellen an die Anwendungen. Dabei werden die Schnittstellen zu den drei Anwendungsarten Produzent, Konsument und Verarbeiter betrachtet wobei die inhaltlichen Aspekte ausgespart werden. Diese werden im Anschluss separat diskutiert.

Produzenten

Die Aufgabe der Produzenten ist das Veröffentlichen von Nachrichten. Dieser Prozess findet zweistufig statt. Das heißt, zuerst muss der Produzent die von ihm zu veröffentlichenden Nachrichten beschreiben. Erst im Anschluss darf er Nachrichten veröffentlichen.

Die Beschreibung der zu veröffentlichenden Nachrichten besteht aus der Angabe der Anwendungsklasse und der Ankündigung. Der Inhalt von Ankündigungen wird im Anschluss separat diskutiert.

Anschließend werden Nachrichten nach dem Prinzip Fire and Forget veröffentlicht. Die Nachrichten werden einzeln an die Middleware übergeben und dort gespeichert. Der Kontrollfluss geht direkt zurück an die Anwendung. Die Nachricht wird von der Middleware entsprechend des vorhandenen Interesses und der Dienstgüteimplikationen der Anwendungsklasse verteilt.

Konsumenten

Konsumenten registrieren sich bei der Middleware mit oder ohne Filter. Filter dienen dazu, die Zahl der an den Konsumenten übertragenen Ankündigungen zu beschränken. Inhaltliche Anforderungen an Filter werden im Anschluss separat diskutiert.

Entsprechend der Registrierung erhält ein Konsument danach Ankündigungen von der Middleware, die er akzeptieren oder ablehnen kann. Wird eine Ankündigung akzeptiert, so werden anschließend die dazugehörigen Nachrichten übermittelt.

Eine Realisierung der Kopplungseigenschaft kontinuierliche Anfrage lässt sich über eine Registrierung mit Filter und das Akzeptieren aller Ankündigungen erreichen. Registrierung mit Rückfrage ist mit und ohne Filter zur Einschränkung der Anzahl der Rückfragen möglich.

Verarbeiter

Ein Verarbeiter ist anwendungsseitig eine Kombination aus Konsument und Produzent. Er meldet sich genau wie ein Konsument mit oder ohne Filter an der Middleware an. Anschließend erhält er Ankündigungen.

Wenn er in der Lage ist, empfangene Ankündigungen zu verarbeiten, dann veröffentlicht er wie ein Produzent eine neue Ankündigung mit dem Unterschied, dass dabei zusätzlich Referenzen auf die Ankündigungen, aus deren Nachrichten er zur generierten Ankündigung passende Nachrichten produzieren kann, an die Middleware übertragen werden müssen. Nur so ist für die Middleware nachvollziehbar, welche Art von Verarbeitung der Verarbeiter realisiert. Außerdem muss die Middleware vom Verarbeiter wissen, ob er die Nachrichten zustandslos oder zustandsbehaftet verarbeiten kann.

Wenn der Verarbeiter anschließend Nachrichten entsprechend der angegebenen Ausgangsankündigungen erhält, so muss er diese in Nachrichten zu den von ihm veröffentlichten Ankündigungen verarbeiten und diese veröffentlichen.

4.6.3. Inhaltliche Anforderungen

Aus der Beschreibung der Schnittstellen sind drei Nachrichtentypen hervorgegangen. Diese sind Nachrichten, Ankündigungen und Filter. Außerdem werden zur Etablierung eines Kommunikationspfades noch Abonnements benötigt. In diesem Abschnitt werden die inhaltlichen Anforderungen dieser vier Nachrichtentypen diskutiert.

Nachrichten

Nachrichten werden auf den durch Ankündigungen und Abonnements etablierten Pfaden von Produzenten zu Konsumenten übertragen. Dabei findet eine Manipulation des Inhaltes ausschließlich durch Verarbeiter statt. Die Middleware selbst hat nur die Aufgaben, notwendige Verarbeiter korrekt einzubinden, so dass Produzenten und Konsumenten sich verstehen, sowie Nachrichten auf den entsprechenden Pfaden weiterzuleiten.

Eine Verarbeitung von Nachrichten, eine Auswahl oder eine Aggregation findet in der Middleware nicht statt. Zu große Nachrichten werden bei Bedarf in Fragmente zerlegt und zu einem späteren Zeitpunkt wieder zusammengefügt. Weiterleitung und Fragmentierung stellen keinerlei Anforderungen an den Inhalt von Nachrichten. Dieser ist somit einzig den Anwendungen überlassen.

Ankündigungen und Abonnements

Die Verbreitung von Ankündigungen in der gesamten Verarbeitungstopologie sowie die Etablierung von einzelnen Pfaden durch Abonnements sind die Voraussetzung für die erfolgreiche Vermittlung von Nachrichten.

Die flexibelste Variante aus Sicht der Anwendungen ist ein vollständig durch die Anwendungen definierter Inhalt der Ankündigungen sowie eine einfache Referenz auf diese in Abonnements. Die Referenz wäre in Form einer Prüfsumme über einen von der Anwendung als signifikant markierten Teil des Inhaltes der Ankündigung realisierbar. Das inhaltsbasierte Routing findet im Wesentlichen über die Prüfsumme statt.

Der große Nachteil dieser Strategie ist, dass bei der Verarbeitung von Ankündigungen häufig stark parametrisierbare Ergebnisse entstehen können. Die Wahl von Parametern durch den Empfänger dieser Ankündigung ist hier nicht möglich.

Ein Ausweg ist die Aufteilung der Ankündigung in einen durch die Anwendung definierten Inhalt und einen durch die Anwendung definierten Parameterblock. Zweiterer muss dann auch Inhalt der Abonnements sein, da auf diesem Weg der Empfänger der Ankündigung eine Auswahl der Parameter definieren kann.

Ankündigungen werden in Abonnements nun über die Prüfsumme des Inhaltsblockes referenziert und mit einem ausgefüllten Parameterblock versehen. Eine Prüfsumme über den Parameterblock dient in Verbindung mit der zuvor genannten Prüfsumme der Identifikation von Pfaden.

Ob unterschiedliche Parameterblöcke tatsächlich zu unterschiedlichen Nachrichten führen oder eine Überdeckung[6] ausgenutzt werden kann, muss bei dieser Strategie der jeweilige Verarbeiter entscheiden. Die Zustellung in der Middleware erfolgt dann ausschließlich durch einen Abgleich der Prüfsummen.

[6] siehe Überdeckungsbasiertes Routing in Abschnitt 2.1.2

Filter

Filter dienen der lokalen Entscheidung, ob ein Broker eine Ankündigung an eine Anwendung überträgt, oder nicht. Sie werden nicht in der Vermittlungstopologie verteilt. Daher ist in Verbindung mit durch die Anwendung definierten Inhalten der Ankündigungen eine große Flexibilität bei der Realisierung dieser Filter möglich. Einfache Inhaltsvergleiche, Platzhalter und reguläre Ausdrücke sind genauso denkbar wie ausführbarer Programmcode.

Auch der vollständige Verzicht auf Filter ist im Prinzip kein Problem. Die Prüfung, ob eine Ankündigung von einer Anwendung akzeptiert wird oder nicht, zum Beispiel durch eine Rückfragefunktion, beeinflusst den Kontrollfluss der Anwendung selbst nicht.

Alternativ sind selbstverständlich durch die Middleware vorgegebene Inhaltsstrukturen möglich, wie sie bereits in vielen Publish/Subscribe-Systemen eingesetzt werden und in Abschnitt 2.1.1 beschrieben wurden. Diese führen aber zu einem Verlust der Flexibilität des Systems aus Anwendungssicht.

4.7. Zusammenfassung

In diesem Kapitel wurde ein Konzept entwickelt, dass es erlaubt, die Anforderungen an die Kommunikation, die sich durch die Problemstellung ergeben, auf ein inhaltsbasiertes Routingproblem abzubilden. Dabei wird die inhaltliche Entkopplung auf eine Pfadsuche durch die Verarbeiter in einer Vermittlungstopologie abgebildet. Eine beliebige Heterogenität und Veränderlichkeit der Netzwerktopologie mit allen damit verbundenen Problemen wird dadurch ermöglicht, dass jedes Gerät in der Systemarchitektur einen eigenen Broker unterhält und die Netzwerktopologie ausschließlich als ein verbundener Graph aus lokal eindeutigen Nachbarschaftsbeziehungen betrachtet wird. Darin werden Nachrichten lediglich von Nachbar zu Nachbar kommuniziert. Somit muss kein entferntes Gerät eindeutig oder dauerhaft adressiert werden.

Um dies zu erreichen, wurden eingangs verschiedene Ansätze aus dem Stand der Forschung vorgestellt und deren Schwächen in Bezug auf die Kombination von inhaltlicher Entkopplung zwischen Produzenten und Konsumenten in Verbindung mit der Funktion in heterogenen, veränderlichen Umgebungen dargestellt. Eine adäquate Problemlösung gibt es im Stand der Forschung bisher nicht.

Anschließend wurde die Vermittlungstopologie als Kombination von Netzwerktopologie und Verarbeitungstopologie vorgestellt und hinsichtlich ihrer Komplexitätseigenschaften und der Implikationen zustandsbehafteter im Unterschied zu zustandsloser Nachrichtenverarbeitung diskutiert. Die Vermittlungstopologie bildet die Grundlage für die sich anschließenden Überlegungen.

Um die Vermittlungstopologie ausreichend flexibel auf ein heterogenes, veränderliches Netzwerk abbilden zu können, wurde eine vollständig verteilte Systemarchitektur vorgestellt, in der jeder Broker in je einer Instanz pro Nachrichtentyp alle dem Gerät zugeordneten

Ecken der Vermittlungstopologie repräsentiert. Dabei wird die Verwaltung von Konsumenten und Produzenten sowie die Abstraktion von Nachrichtenverarbeitungsverbindungen in der Vermittlungstopologie durch eine Anwendungsabstraktionsschicht des Brokers und die Kommunikation mit den benachbarten Brokern durch eine Netzwerkabstraktionsschicht des Brokers übernommen. Zwischen diesen beiden Schichten liegt die Vermittlungsschicht des Brokers, in der das inhaltsbasierte Routing angesiedelt ist.

Für das Routing wurde die grundsätzliche Eignung der drei bekannten Ansätze zum inhaltsbasierten Routing diskutiert. Ergebnis dieser Diskussion ist, dass das Flooding höchstens als Referenz für die Flexibilität und Robustheit für bestimmte Arten von Anwendungen geeignet ist. Das Verbreiten allen Interesses ist in der Vermittlungstopologie prinzipbedingt sehr problematisch. Damit verbleibt das inhaltsbasierte Routing mit Ankündigungen als Basis für geeignete Kommunikationsalgorithmen.

Zur Bewertung und zum Vergleich gefundener Pfade ist eine Routingmetrik nötig, die sich in wesentlichen Anforderungen von den existierenden Routingmetriken unterscheidet. Diese Anforderungen wurden diskutiert und eine einfache auf der Zeit basierende Metrik, die diesen Anforderungen gerecht wird, vorgestellt.

Abschließend wurden die Anforderungen an Produzenten, Konsumenten und Verarbeiter diskutiert. Kernpunkt bildeten dabei die Schnittstellen und die Inhalte der einzelnen Nachrichtentypen. Außerdem wurde eine Möglichkeit zur Klassifizierung von Anwendungen basierend auf der Größe und Anzahl der von ihnen veröffentlichten Nachrichten vorgestellt und die Implikationen der Wahl einer bestimmten Klasse auf die Dienstgüteeigenschaften der Kommunikation sowie auf die Voraussetzungen des Routing in der Middleware diskutiert.

Aufbauend auf diesen konzeptuellen Überlegungen können konkrete Routingalgorithmen entworfen werden, um in einer Vermittlungstopologie Produzenten und Konsumenten effizient miteinander zu verbinden. Der Entwurf zweier solcher Algorithmen ist Inhalt des nächsten Kapitels.

Kapitel 5.

Algorithmen zum Routing in einer Vermittlungstopologie

Inhalt

5.1 Grundsätzliche Entscheidungen **92**
5.1.1 Überblick 93
5.1.2 Notation 93
5.1.3 Verbreitung der Ankündigungen 94
5.1.4 Gültigkeitsdauer von Ankündigungen 97
5.1.5 Identifikation der Senken 99
5.1.6 Zwischenspeichern von Veröffentlichungen 99
5.2 Pfadauswahl in Phase 2 . . . . . **101**
5.2.1 Umgang mit Ankündigungen 101
5.2.2 Abonnements, Nachrichten und Sonderfälle 102
5.2.3 Diskussion 103
5.3 Pfadauswahl in Phase 3 . . . . . **103**
5.3.1 Umgang mit Ankündigungen 104
5.3.2 Markieren möglicher Pfade 105
5.3.3 Vorgehen bei Pfadverlust 108
5.3.4 Diskussion 110
5.4 Diskussion **111**
5.4.1 Komplexitätsabschätzung der entwickelten Algorithmen 111
5.4.2 Skalierungsfähigkeit der Algorithmen 115
5.4.3 Parameter und deren Einfluss 118
5.4.4 Zuordnung zu Anwendungsklassen 120
5.5 Zusammenfassung **121**

In diesem Kapitel werden zwei Algorithmen vorgestellt, die es erlauben, inhaltsbasiertes Routing in der zuvor vorgestellte Vermittlungstopologie zu benutzen, um effiziente Pfade von Konsumenten zu Produzenten zu finden. Dabei wird wie folgt vorgegangen.

Nach einem kurzen Überblick über die Funktionsweise der beiden Algorithmen werden detailliert grundsätzliche Entscheidungen vorgestellt, die bei beiden Algorithmen von Bedeutung sind. Anschließend werden die speziellen Annahmen für einen Ein-Wege-Ansatz vorgestellt, mit dem auf sehr einfache Art ein Pfad pro Quellen-Senken-Kombination gefunden und aufrecht erhalten werden kann. Die Vor- und Nachteile dieses Ansatzes werden diskutiert bevor ein komplexerer Mehr-Wege-Ansatz vorgestellt wird, der es ermöglicht, ständig den effizientesten bekannten Pfad pro Quellen-Senken-Kombination zu benutzen und bei Problemen ohne aktive Reparatur des Pfades auf andere Pfade auszuweichen. Auch dieser Ansatz wird diskutiert.

Abschließend werden wesentliche Parameter beider Ansätze identifiziert und deren zu erwartender Einfluss auf die Pfadsuche und die Zustellung von Nachrichten sowie die Komplexität der Algorithmen und deren Skalierungsfähigkeit abgeschätzt.

5.1. Grundsätzliche Entscheidungen

Bei beiden Algorithmen handelt es sich um inhaltsbasierte Routingalgorithmen mit Ankündigungen entsprechend der Diskussion in 4.4.3. Sie arbeiten demzufolge in drei Phasen:

Ankündigung Phase 1 ist die Verbreitung von Ankündigungen in der Vermittlungstopologie. Sie dient der Verbreitung der Verfügbarkeit von Nachrichten eines bestimmten Nachrichtentyps.

Abonnement Als Reaktion auf die Verfügbarkeit wird von der Nachrichtensenke als zweite Phase ein Abonnement versandt. Damit registriert sich die Nachrichtensenke als Empfänger von Nachrichten, die mit der Ankündigung assoziiert sind.

Nachrichten In der dritten Phase werden die Nachrichten selbst verbreitet. Diese müssen jede der in Phase 2 registrierten Nachrichtensenken erreichen (gegenwärtige zeitliche Entkopplung). Wobei durch gezielte Zwischenspeicherung zusätzlich erreicht werden soll, dass eine neue Nachrichtensenke sehr schnell mit den aktuellsten verfügbaren Nachrichten versorgt wird, ohne auf die Veröffentlichung neuer Nachrichten warten zu müssen (teilweise zukünftige zeitliche Entkopplung).

Die Phasen dienen dabei der logischen Gliederung des Ablaufs des Algorithmus. Sie beschreiben nicht den Zustand des Gesamtsystems. Es ist durchaus möglich, dass sich eine Nachrichtensenke für eine Ankündigung registriert und damit in Phase 2 befindet während eine andere Nachrichtensenke schon Nachrichten erhält und sich somit in Phase 3 befindet.

Während die drei Phasen lediglich den Aufbau eines Vermittlungspfades beschreiben, so gehören auch die Wartung, also das Erkennen eines Defektes und dessen Reparatur zum Ablauf des Algorithmus. Außerdem kann es gewünscht sein, einen Pfad abzubauen, also

ein Abonnement zu kündigen. Diese Anforderungen müssen sowohl in diesem Abschnitt als auch deren Implikation auf die einzelnen Algorithmen in den folgenden Abschnitten berücksichtigt werden.

Nach einem Überblick über die Funktionsweise und die wesentlichen Unterschiede der beiden Ansätze, werden in diesem Abschnitt wichtige Aspekte der Notation eingeführt. Im Anschluss werden die gemeinsamen Strategien zur Verbreitung von Ankündigungen, zur Identifikation von Senken und zum Zwischenspeichern von Veröffentlichungen diskutiert. Den Abschluss bildet eine Diskussion der Notwendigkeit expliziter Pfadreparatur und deren Realisierung im Allgemeinen.

5.1.1. Überblick

Bei beiden Algorithmen wird in der ersten Phase die Verfügbarkeit von Nachrichten eines Typs durch eine Ankündigung im System verbreitet. Kopien dieser Ankündigung werden von der Nachrichtenquelle aus entsprechend dem Flooding-Algorithmus verteilt. Der wesentliche Unterschied besteht in beziehungsweise vor der Phase 2. Hier wird beim Ein-Wege-Ansatz für jede empfangene Ankündigung der Absender mit den besten Pfadkosten gespeichert. Von der Senke aus wird nach Empfang der Ankündigung ein Abonnement versandt, dass jeweils an den gespeicherten aktuell kostengünstigsten Absender weitergeleitet wird, bis es bei der Nachrichtenquelle ankommt. Dadurch wird der momentan kostengünstigste bekannte Pfad zwischen Quelle und Senke etabliert und im Anschluss so lange benutzt, wie er verfügbar ist. Wird der Pfad auf einer Verbindung unterbrochen, so muss dies zur Quelle hin signalisiert und von dort beginnend ein neuer Pfad etabliert werden.

Beim Mehr-Wege-Ansatz wird mehr als ein Absender pro empfangener Ankündigung gespeichert, an die anschließend das Abonnement weitergeleitet wird. So entsteht eine größere Anzahl von Pfaden, die in Phase 3 zum Nachrichtenversand benutzt werden können. Die Auswahl des kostengünstigsten Pfades geschieht hier bei der Nachrichtenübermittlung. Hier kann bei der Unterbrechung eines Pfades mit sehr geringem Signalisierungsaufwand auf einen weniger günstigen Pfad ausgewichen werden.

5.1.2. Notation

Zentrales Element bei der Beschreibung und Umsetzung der Algorithmen ist die Ecke der Vermittlungstopologie. Dabei wird eine Ecke von einer Instanz eines Brokers abgebildet. Jeder Broker besteht aus je einer Instanz pro Nachrichtentyp, verwaltet also so viele Ecken wie Nachrichtentypen. Bei der Beschreibung der Algorithmen wird dieser Sachverhalt ignoriert und der Algorithmus für genau eine Ecke beschrieben.

Die Nachbarn einer Ecke sind die angeschlossenen Produzenten und Konsumenten sowie die über Verarbeiter erreichbaren Ecken auf demselben Broker und die über das Netzwerk erreichbaren Ecken auf benachbarten Brokern. Produzenten werden von der Anwendungs-

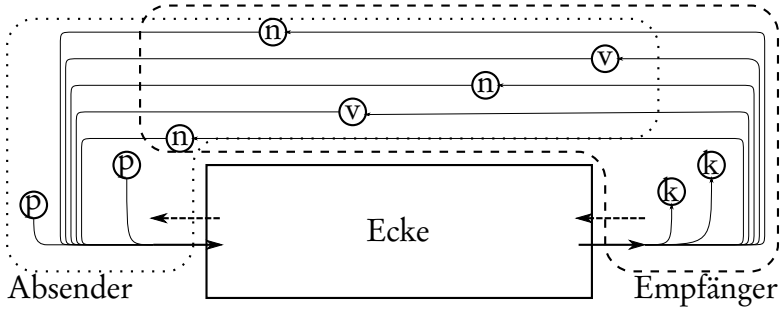


Abbildung 5.1.:
Graphische Repräsentation einer Ecke der Vermittlungstopologie mit ihren Nachbarn. Dazu gehören die über Netzwerkverbindungen angebundenen Ecken (n) sowie die registrierten Produzenten (p), Konsumenten (k) und Verarbeiter (v). Dargestellt sind die möglichen Pfade für Ankündigungen und Nachrichten (durchgezogene Pfeile und dünne Linien) sowie die Richtung, in der Abonnements die Ecke durchlaufen (gestrichelte Pfeile).

abstraktionsschicht als reine Absender und Konsumenten als reine Empfänger von Ankündigungen und Nachrichten abgebildet. Die anderen Nachbarn können sowohl als Absender, als auch Empfänger agieren.

Die Erklärung der Algorithmen wird von einer graphischen Notation begleitet. Zentrales Element dieser Notation ist die Ecke. Die Absender sind virtuell links dieser Ecke angeordnet und die Empfänger rechts. Diese Anordnung dient lediglich der bildlichen Darstellung, da die Mengen der Absender und Empfänger nicht disjunkt sind. Abbildung 5.1 veranschaulicht dies. Später werden die einzelnen Nachbarn der Übersichtlichkeit halber nicht mehr dargestellt.

5.1.3. Verbreitung der Ankündigungen

Eine Ankündigung repräsentiert genau einen Nachrichtentyp. Dieser wird in der Nutzlast der Ankündigung beschrieben und durch die AID vertreten[1]. Da jede Ecke auch genau einen Nachrichtentyp repräsentiert, wird diese AID bei der Beschreibung des Algorithmus nicht benötigt. Eine Ecke kann nur Ankündigungen mit einer AID erhalten.

Neben der AID enthält eine Ankündigung eine Gültigkeit und eine Sequenznummer. Kurz vor Ablauf der Gültigkeit wird eine neue Ankündigung mit gleicher AID und erhöhter Sequenznummer verbreitet, um die Gültigkeit rechtzeitig zu verlängern. Zwei Ankündigungen mit gleicher AID und Sequenznummer gelten als identisch, da sie denselben Ankündigungsvorgang repräsentieren. Für die Bewertung und den Vergleich von Pfaden enthält eine Ankündigung außerdem eine Schätzung der Größe assoziierter Nachrichten sowie die

[1] siehe dazu in Abschnitt 4.3.2

Kürzel	Bedeutung
aid	repräsentierter Nachrichtentyp
val	verbleibender Gültigkeitszeitraum
seq	Sequenznummer
size	Schätzung der Größe assoziierter Nachrichten
cost	Pfadkosten des bisher zurückgelegten Pfades

Tabelle 5.1.:
Bestandteile einer Ankündigung und die dafür in Zukunft verwendeten Kürzel.

Pfadkosten des bisher zurückgelegten Pfades. Tabelle 5.1 fasst den Inhalt einer Ankündigung zusammen.

Ankündigungen werden mit Hilfe des Flooding-Algorithmus[2] verbreitet. Dazu prüft die Ecke, die eine Ankündigung erhält, ob sie eine identische Ankündigung bereits erhalten hat. Wenn nicht, dann speichert sie die erhaltene Ankündigung und leitet sie im Anschluss an alle Nachbarn außer deren Absender weiter. Wenn doch, dann wird die Ankündigung nicht weitergeleitet. Ob Informationen über diese bereits empfangene Ankündigung gespeichert werden oder nicht, hängt vom konkreten Algorithmus ab.

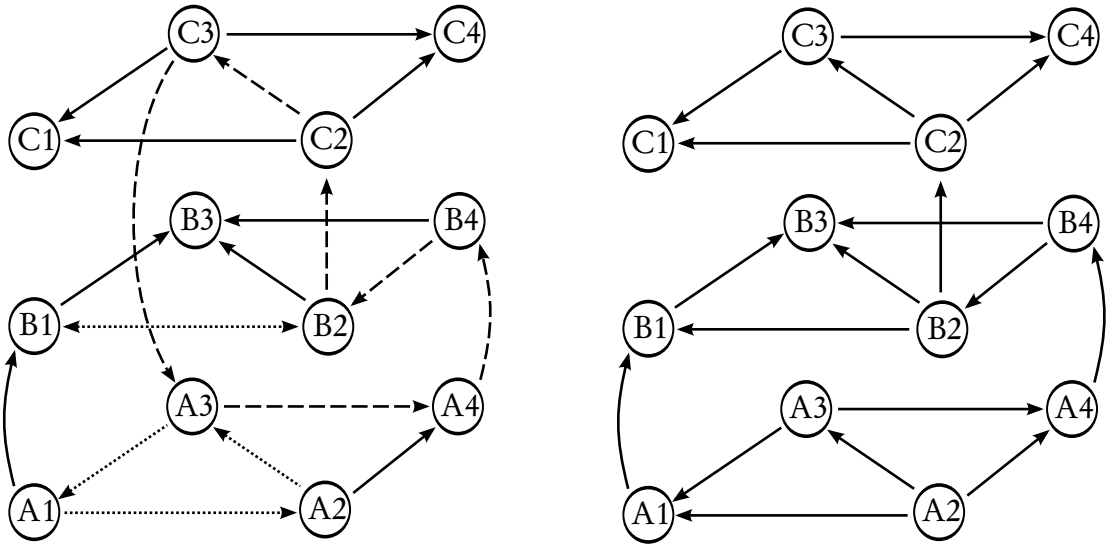
Sobald durch Veränderung der Vermittlungstopologie ein neuer Empfänger bekannt wird, wird die gespeicherte Ankündigung mit entsprechend angepasstem *val* auch an diesen weitergeleitet. Dadurch ist eine zeitlich entkoppelte Verteilung aller Ankündigungen im Gesamtsystem gewährleistet.

Versand von Ankündigungen

Beim Empfang einer Ankündigung vom Produzenten bzw. der Anwendungsabstraktionsschicht als Quelle, sind *aid*, *val* und *seq* bereits initialisiert und werden bis auf *val*, im im letzten Abschnitt erwähnten Fall, nicht mehr verändert. *size* ist ebenfalls initialisiert und wird lediglich bei der Weiterleitung durch Nachbarschaftsverbindungen, die Verarbeiter repräsentieren, verändert, wenn diese die Größe assoziierter Nachrichten durch deren Verarbeitung verändern.

cost wird von der Quelle auf `0` initialisiert und wird mit jedem Weiterleitungsvorgang um die größenabhängigen Verbindungskosten zwischen den beteiligten Ecken erhöht[3]. Diese Erhöhung findet in der Netzwerk- bzw. Anwendungsabstraktionsschicht statt, die Höhe der Veränderung wird der versendenden Ecke mitgeteilt und sie ist immer größer als `0`. Bei der Weiterleitung an einen Konsumenten wird gegebenenfalls *val* angepasst. Alle anderen Werte bleiben unverändert.

[2] siehe Abschnitt 2.2.1
[3] Details zur Erhöhung der Pfadkosten und der Veränderung der Größe wurden im Abschnitt 4.5.3 erklärt



(a) Zyklischer Verteilungsgraph (b) Azyklischer Verteilungsgraph

Abbildung 5.2.:
Zwei Beispiele für Verteilungsgraphen der Ankündigungen mit A2 als Quelle und C4 als Senke: Drei Arten von Zyklen sind skizziert. Zwischen A2, A3 und A1 entsteht ein Zyklus, weil A2 eine selbst an A3 gesandte Ankündigung wieder akzeptiert. Zwischen B1 und B2 entsteht ein Zyklus, weil beide ihre von dritten erhaltene Ankündigung absenden, während die Ankündigung des jeweils anderen noch nicht empfangen wurde, dann aber auch beide die empfangene Ankündigung akzeptieren. Der lange Zyklus entsteht, weil eine Ankündigung nach mehreren Verarbeitungsschritten wieder im Ausgangstyp ankommt und dort von A3 akzeptiert wurde. Jeder dieser Zyklen lässt sich auflösen, indem eine der Ankündigung nicht und gegebenenfalls stattdessen eine auf einer entgegengesetzt gerichteten Verbindung übertragene Ankündigung akzeptiert wird.

Verteilungsgraph der Ankündigung

Der Verteilungsgraph einer Ankündigung entsteht aus den gerichteten Verbindungen zwischen jeweils der Ecke, die eine Ankündigung versendet, und der Ecke, die diese Ankündigung empfängt und akzeptiert, also Informationen über diese Ankündigung zum Zweck der späteren Etablierung von Pfaden speichert. Die Entscheidung, ob eine Ankündigung akzeptiert wird oder nicht, hängt vom konkreten Algorithmus und dem Wert von *cost* der jeweiligen Ankündigung ab.

Der auf diese Weise entstehende zusammenhängende, gerichtete Graph beinhaltet alle Pfade, auf denen spätere Nachrichten versandt werden können. Daher muss jeder Algorithmus Sorge dafür tragen, dass dieser Graph entweder bei seiner Entstehung bereits zyklenfrei ist oder in der Phase der Abonnements entsprechende Zyklen entfernt werden.

Abbildung 5.2(a) zeigt einen Verteilungsgraphen der Ankündigung, der mehrere Zyklen enthält, die im späteren Verlauf der Kommunikation zu zyklischen Pfaden führen können. In der gleichen Vermittlungstopologie zeigt Abbildung 5.2(b) einen zyklenfreien Verteilungsgraphen der Ankündigungen, der sich lediglich dadurch unterscheidet, dass in jedem Zyklus eine Ankündigung entweder nicht akzeptiert und somit der Zyklus unterbrochen oder anstelle dieser Ankündigung eine in entgegengesetzter Richtung übertragene Ankündigung akzeptiert wurde.

5.1.4. Gültigkeitsdauer von Ankündigungen

Ankündigungen werden von der Quelle mit der Gültigkeitsdauer *val* versehen. Nach dem Ablauf dieser Zeit werden die Ankündigungen ungültig und aus den Zwischenspeichern aller Broker gelöscht. Die Verlängerung der Gültigkeit erfolgt regelmäßig durch die Nachrichtenquelle, also den Broker des Produzenten. Durch dieses Verfahren wird gewährleistet, dass Ankündigungen, für die keine Nachrichten mehr veröffentlicht werden, weil zum Beispiel der Produzent nicht mehr an seinem Broker angemeldet ist, nicht mehr verlängert und dadurch automatisch aus dem System entfernt werden.

Zur Verlängerung einer Ankündigung wird deren Sequenznummer *seq* von der Quelle erhöht. Eine Ankündigung mit höherer Sequenznummer ersetzt alle vorherigen Ankündigungen gleichen Nachrichtentyps. Nach der Verarbeitung durch einen Verarbeiter darf die generierte Ankündigung von der Middleware nicht einfach mit einer neuen Gültigkeit und Sequenznummer versehen werden. Dies würde beim Vorhandensein mehrerer gleichartiger Verarbeiter zwangsläufig zu einer Race Condition führen, bei der sich der Verarbeiter durchsetzt, für den die höhere Sequenznummer erzeugt wird (SEUCHTER, 2009, S.72f).

Um diese Race Condition zu verhindern, müssen die Sequenznummern der neuen Ankündigungen aus den Sequenznummern der Ausgangsankündigungen errechnet werden. Hier bietet sich die Summe der entsprechenden Sequenznummern an, da diese sich immer erhöht, wenn sich eine der Ausgangssequenznummern erhöht.

Für die Ermittlung der Gültigkeit einer verarbeiteten Ankündigung gibt es zwei Strategien. Es kann die verbleibende Gültigkeit der kürzesten (a) oder längsten (b) gültigen Ausgangsankündigung verwendet werden. Nachteil der Strategie a ist, dass sehr viele neue Ankündigungen erzeugt werden. Nachteil der Strategie b ist, dass nicht sichergestellt wird, dass bis zum Ablauf der Gültigkeit der neuen Ankündigung auch Daten der Ausgangstypen verfügbar sind. Abbildung 5.3 veranschaulicht diesen Sachverhalt.

Mehrere identische Quellen

Ein Spezialfall für die Gültigkeit und Verlängerung von Ankündigungen stellt der Fall dar, in dem zwei Quellen identische Nachrichten veröffentlichen möchten. Angenommen die Beschreibung der Nachrichten in der Ankündigung wäre ebenfalls identisch und würde so-

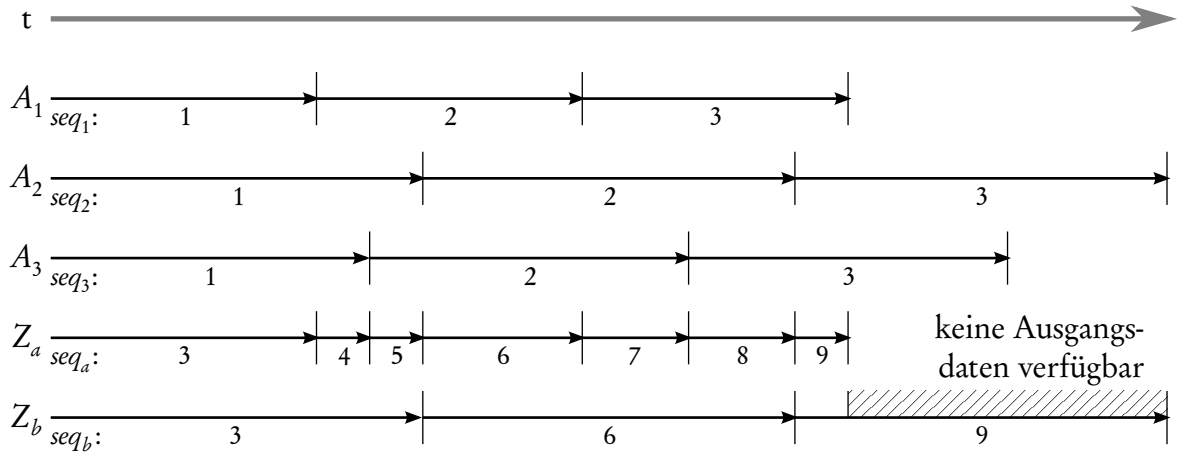


Abbildung 5.3.:
Ein Verarbeiter erzeugt aus den Ankündigungen A_1, A_2 und A_3 eine neue Ankündigung $Z_{(a/b)}$. Z_a entspricht Strategie a, wobei jeweils die minimale verbleibende Gültigkeitsdauer der Ausgangsankündigungen für die neue Ankündigung übernommen wird. Z_b entspricht Strategie b mit der maximalen verbleibenden Gültigkeitsdauer. Die Sequenznummern seq_a und seq_b berechnen sich jeweils als $seq_1 + seq_2 + seq_3$. Man erkennt deutlich, dass für Strategie a sehr viele Ankündigungen entstehen, während bei Strategie b die letzte Ankündigung weit länger gültig ist, als überhaupt Nachrichten der Ausgangstypen produziert werden können.

mit die gleiche inhaltsbasierte AID ergeben. Es existieren also zwei Quellen für denselben Nachrichtentyp. In diesem Fall würde die in SEUCHTER (2009, S.72f) beschriebene Race Condition ebenfalls auftreten, ließe sich aber nicht mit den zuvor beschriebenen Mitteln beseitigen.

Tritt dieses Problem auf, würde die Quelle mit der höheren Sequenznummer die Ankündigungen der anderen Quelle sofort ungültig machen. Dies bedeutet, dass langfristig die Quelle mit der geringeren Ankündigungsgültigkeit erfolgreich Nachrichten veröffentlichen kann während die andere Quelle gar keine Nachrichten mehr veröffentlichen kann.

Im Prinzip ist das kein Problem, da beide Quellen die gleichen Nachrichten veröffentlichen und die Senken diese Nachrichten erfolgreich empfangen. Nimmt man aber an, dass zwei Quellen die gleichen Nachrichten veröffentlichen, so wird dies nicht zufällig zustande kommen, sondern könnte gerade zum Zweck der Lastverteilung geschehen. Dieser Zweck würde durch die faktische Abschaltung einer Quelle durch den Verteilungsalgorithmus ad absurdum geführt.

Um beiden Quellen das erfolgreiche Veröffentlichen von Nachrichten zu erlauben, ist es notwendig, dass sie die Gültigkeit ihrer Ankündigungen und Sequenznummern aufeinander abstimmen. Dies ist ohne Probleme möglich. Sobald eine Quelle q_1 eine Ankündigung mit

der gleichen AID, wie sie veröffentlicht, und einer größeren Sequenznummer erhält ist klar, dass eine zweite Quelle q_2 existieren muss.

In diesem Fall leitet sie die fremde Ankündigung nicht entsprechend dem Algorithmus weiter, sondern erzeugt eine eigene Ankündigung mit der gleichen Sequenznummer und Gültigkeit und veröffentlicht diese. Dadurch werden, von q_1 ausgehend, die Ankündigungen der anderen Quelle so lange überschrieben, bis die Pfadkosten größer werden als die der Ankündigungen von q_2. Dies führt dazu, dass jede Senke automatisch die Nachrichten von der ihr nächstgelegenen Quelle abonniert.

5.1.5. Identifikation der Senken

Da Produzent und Konsument räumlich entkoppelt miteinander kommunizieren, ist eine Identifikation der Senke grundsätzlich nicht notwendig. Um jedoch ein Abonnement rückgängig machen oder das Wegbrechen eines Pfades zu einer spezifischen Senke erkennen zu können, ist eine Identifikation des Pfades zu einer jeden Senke notwendig. Diese Identifikation wird von der Senke für jedes Abonnement zufällig generiert und als SID bezeichnet.

Um einen Pfad eindeutig kennzeichnen zu können, muss die SID in Kombination mit der AID eindeutig sein. Dies kann prinzipbedingt nicht gewährleistet werden. Je nach Wertebereich der SID kann die Wahrscheinlichkeit einer Kollision verringert werden. Tritt trotzdem eine Kollision auf, wird ein Broker, spätestens die Quelle, die beiden Pfade als zu einer Senke führend ansehen und Nachrichten nur an einen der beiden Pfade versenden. Es wird also nur die Senke mit dem kostengünstigeren Pfad Nachrichten erhalten. Da die SID mit jeder Ankündigung neu generiert wird, hält dieser Zustand nur so lange an, wie die aktuelle Ankündigung gültig ist. Außerdem werden Teile der nicht übertragenen Daten nach der Neuankündigung und dem Abonnement mit einer eindeutigen SID aus den Zwischenspeichern der Broker übertragen. Die Zwischenspeicherung von Nachrichten wird im nächsten Abschnitt diskutiert.

5.1.6. Zwischenspeichern von Veröffentlichungen

Um zeitliche Entkopplung gewährleisten zu können, müssen Nachrichten in der Middleware, also auf den Brokern, zwischengespeichert werden. Eine Anforderung durch die Problemstellung ist gegenwärtige zeitliche Entkopplung. Das heißt, jede der Middleware zum Zeitpunkt der Veröffentlichung bekannten Senken sollen eine Nachricht erhalten. Übersetzt in die Semantik des Algorithmus bedeutet dies, eine Nachricht muss zwischengespeichert werden bis alle defekten Pfade repariert und die entsprechenden Senken die Nachrichten erhalten hat.

Die zweite Anforderung der eingeschränkten zukünftigen zeitlichen Entkopplung bedeutet, dass Nachrichten für eventuell zukünftig bekannt werdende Senken noch eine Weile

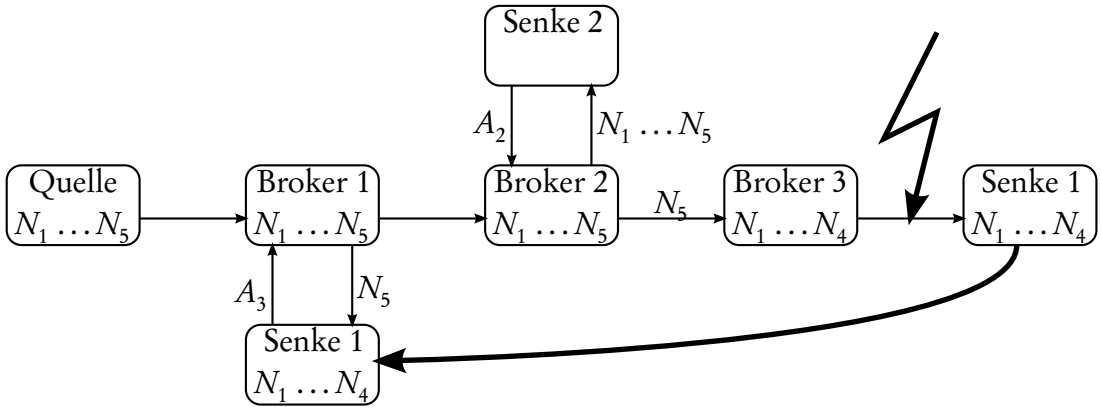


Abbildung 5.4.:
Der Produzent an der Quelle hat fünf Nachrichten N_1 bis N_5 veröffentlicht. Diese wurden erfolgreich bis zu Broker 2 übertragen und jeweils zwischengespeichert. Senke 2 kommt neu hinzu. Durch die Zwischenspeicherung kann Broker 1 bereits alle fünf Nachrichten übertragen bevor das Abonnement A_2 überhaupt weitergeleitet wird. Eine wiederholte Übertragung zwischen der Quelle und Broker 2 wurde eingespart. Bis zu Senke 1 sind erst die Nachrichten N_1 bis N_4 übertragen worden. Dann verändert sie ihre Position, so dass die Verbindung von Broker 3 abbricht. Eine neue Verbindung zu Broker 1 wird aufgebaut und ein neues Abonnement A_3 versandt. An dieser Stelle muss die unnötige Übertragung der Nachrichten N_1 bis N_4 verhindert werden, die durch die Zwischenspeicherung ausgelöst würde.

vorrätig gehalten werden. Wie lange dies sinnvoll ist, hängt sehr stark davon ab, wann Nachrichten bedeutungslos werden, ist also eine Frage der Anwendungssemantik.

Für Anwendungen der Klassen[4] 1 und 3 lässt sich grundsätzlich sagen, dass die Aufbewahrung der einzelnen Nachricht bis die entsprechende Ankündigung ungültig wird, sinnvoll ist. Bei Nachrichtenströmen ist es eine Frage der Anwendung, wie viele Nachrichten oder wie lange Nachrichten sinnvoll zwischengespeichert werden sollten.

Neben der zeitlichen Entkopplung hat die Zwischenspeicherung von Nachrichten Vorteile und Nachteile in Bezug auf die benötigte Bandbreite. Durch Zwischenspeicherung können Mehrfachübertragungen von Nachrichten zwischen denselben Brokern vermieden werden, wenn neue Senken daran interessiert sind. Gleichzeitig kann sie zur Mehrfachübertragung von Nachrichten führen, wenn sich der Pfad zu einer Senke verändert. Abbildung 5.4 verdeutlicht diesen Sachverhalt.

Um die Mehrfachübertragung von Nachrichten durch Zwischenspeicherung zu verhindern ist es sinnvoll, Nachrichten mit eindeutigen Identifikatoren zu versehen und anschließend in jedem Broker zu speichern, auf welchem Pfad eine Nachricht bereits übertragen

[4] siehe Einteilung der Anwendungen in Klassen in Abschnitt 4.6.1

wurde. Bei Pfadverlust wird der alte Pfad dann im neuen Abonnement referenziert. Dieses Vorgehen führt zu einer höheren Belastung der Broker mit Statusinformationen.

Eine zweite Möglichkeit besteht darin, in den Abonnements die bereits übertragenen Nachrichten zu referenzieren. Dies ist möglich, da Abonnements eher kleine Netzwerkpakete darstellen. Dies führt zu einer erhöhten Netzwerklast durch größere Abonnements. Aufgrund der Entlastung der Broker von zusätzlichen Statusinformationen und der eher geringen Mehrbelastung des Netzwerkes durch größere Abonnements wird in der Realisierung die zweite Methode umgesetzt.

In den folgenden Abschnitten werden die Details zweier verschiedener Routingalgorithmen beschrieben. Dazu gehören

- die Definition eines Kriteriums, anhand dessen eine Ecke entscheidet, ob eine Ankündigung akzeptiert wird, oder nicht,
- der Nachweis, dass der daraus entstehende Verteilungsgraph der Ankündigungen zyklenfrei ist,
- die Definition geeigneter Maßnahmen, um in Phase 2 Pfade zu markieren und in Phase 3 Nachrichten auf diesen Pfaden zu versenden sowie
- die Reaktion auf Veränderungen der Topologie und die Zerstörung von etablierten Pfaden.

5.2. Pfadauswahl in Phase 2

Beim Ein-Wege-Ansatz wird ein Pfad pro Senke etabliert und für die Übertragung aller Nachrichten benutzt. Dies geschieht in Phase 2. Bricht dieser Pfad zusammen weil ein Broker, ein Verarbeiter oder eine Netzwerkverbindung auf dem Pfad nicht mehr verfügbar ist, muss ein neuer Pfad gefunden werden.

Nachfolgend werden die Spezifika dieses Ansatzes erläutert und anschließend dessen Vor- und Nachteile diskutiert.

5.2.1. Umgang mit Ankündigungen

Wenn eine Ecke eine Ankündigung Ank_i empfängt, wird zwischen drei Fällen unterschieden. Im ersten Fall ($i = 0$) wurde die Ankündigung zuvor noch nicht empfangen. Sie wird entsprechend der Beschreibung in Abschnitt 5.1.3 gespeichert und weitergeleitet. Außerdem werden der Absender in $absd'$ und $Ank_0.cost$ in $cost'$ gespeichert.

Im zweiten Fall enthält die Ankündigung einen geringeren als den gespeicherten Pfadkostenwert ($Ank_i.cost < cost'$). Dann wird der in $absd'$ gespeicherte Absender mit dem vom Ank_i überschrieben. Außerdem werden die in $cost'$ gespeicherten Kosten mit $Ank_i.cost$ überschrieben. Im dritten Fall ($Ank_i.cost \geq cost'$) wird die Ankündigung nicht akzeptiert.

Aus den jeweils zuletzt akzeptierten Ankündigungen entsteht ein sehr verbindungsarmer Verteilungsgraph der Ankündigungen. Für jede akzeptierte Ankündigung in diesem Verteilungsgraphen gilt, dass die in ihr gespeicherten Pfadkosten kleiner als die Pfadkosten aller bisher empfangenen Ankündigungen sind. Außerdem werden die Pfadkosten einer Ankündigung bei ihrer Weiterleitung immer erhöht. Nimmt man nun die Existenz eines Zyklus in diesem Verteilungsgraphen an, so muss für jede Ankündigung, die zur Etablierung einer Kante in diesem Zyklus beigetragen hat, gelten, dass die in ihr gespeicherten Pfadkosten geringer sind als die in der zur Etablierung der nächsten Kante übermittelten Ankündigung gespeicherten Pfadkosten. Dies kann nicht für alle Kanten in einem Zyklus gelten, also kann ein solcher Zyklus nicht existieren. Der Verteilungsgraph der Ankündigungen für diesen Algorithmus ist demzufolge zyklenfrei.

5.2.2. Abonnements, Nachrichten und Sonderfälle

Ist ein Konsument an Nachrichten eines durch eine Ankündigung beschriebenen Nachrichtentyps interessiert, so generiert deren Broker in der Rolle der Senke ein Abonnement. Dieses enthält die AID als Referenz auf den Nachrichtentyp bzw. die entsprechende Ankündigung sowie die zufällig generierte SID, die den Pfad identifiziert und wird an die Ecke, an die der Konsument angeschlossen ist, übermittelt.

Jedes von einer Ecke empfangene Abonnement wird an die momentan in $absd'$ gespeicherte Ecke weitergeleitet. Für den dadurch freigeschalteten Pfad, der durch die in der Ankündigung befindlichen SID *sid* gekennzeichnet ist, wird der Absender des Abonnements als die nächste auf diesem Pfad liegende Ecke in $next_{sid}$ gespeichert. Außerdem wird der Empfänger des weitergeleiteten Abonnements in $last_{sid}$ gespeichert.

Auf diese Weise etabliert sich der zum Zeitpunkt des jeweiligen Empfangs jedes Abonnements kostengünstigste bekannte Pfad zwischen Quellen und Senke als aktiver Pfad, indem alle Nachrichten im Anschluss von jeder Ecke an $next_{sid}$ weitergeleitet werden.

Wird eine Kündigung für den durch *sid* gekennzeichneten Pfad erhalten, so wird diese an $last_{sid}$ weitergeleitet und $next_{sid}$ sowie $last_{sid}$ nicht länger gespeichert.

Ein Zusammenbruch des etablierten Pfades von der Quelle zur Senke wird spätestens dann bemerkt, wenn eine Nachricht nicht an einen Nachfolger auf dem Pfad übertragen werden kann weil dieser nicht mehr verfügbar ist. In diesem Fall wird von der betroffenen Ecke eine Störungsmeldung an die lokal in $last_{sid}$ gespeicherte Ecke übertragen. Die auf einem Pfad am dichtesten an jeder Quelle generierte Störungsmeldung kommt bei dieser Quelle an, da zwischen der dichtest gelegenen Störung und der Quelle alle Verbindungen intakt sind, und löst an der jeweiligen Quelle eine vorzeitige Neuankündigung mit höherer Sequenznummer aus. Mit deren Hilfe werden neue Pfade zur nicht mehr erreichbaren Senke und allen anderen Senken etabliert.

5.2.3. Diskussion

Vorteile dieses Ansatzes sind seine Einfachheit und der geringe Overhead bei der Etablierung von Pfaden sowohl die Anzahl und Größe von Abonnements, als auch die Menge der pro Pfad in den Brokern gespeicherten Informationen betreffend. Neben den Standardaufgaben, also der Zwischenspeicherung der jeweils ersten Ankündigung sowie der aktuellsten Nachrichten, müssen Broker pro Nachrichtentyp nur den Absender der jeweils kostengünstigsten Ankündigung sowie pro Pfad zu einer Senke den Absender und Empfänger des Abonnements speichern.

Im Gegensatz dazu hat dieser Ansatz zwei grundsätzliche Schwächen. Eine Instabilität des gewählten Pfades führt zu einem enormen Kommunikationsoverhead. Sobald ein Pfad unterbrochen wird, wird eine neue Ankündigungsphase ausgelöst, um einen neuen Pfad zur entsprechenden Senke zu finden. Je nach ursprünglicher Gültigkeitsdauer der Ankündigungen kann dieses Vorgehen den Overhead des Algorithmus vervielfachen.

Die zweite Schwäche ist die Wahl nicht-optimaler Pfade. Empfängt eine Senke eine Ankündigung, so ist zu keinem Zeitpunkt klar, ob diese Ankündigung bereits den optimalen durch den Flooding-Algorithmus auffindbaren Pfad repräsentiert oder zu einem späteren Zeitpunkt eine Ankündigung mit geringeren Pfadkosten empfangen wird. In veränderlichen Topologien kann durch die Entstehung zusätzlicher Verbindungen über die Zeit auch zu einem deutlich späteren Zeitpunkt noch immer ein besserer Pfad gefunden werden. Dieser würde bei diesem Ansatz nicht benutzt werden.

Ein Ausweg aus der zweiten Schwäche scheint auf den ersten Blick eine Erweiterung des Algorithmus dahingehen zu sein, aktive Pfade beim Empfang von Ankündigungen mit geringeren Pfadkosten zu aktualisieren. Dies würde durch den Versand eines Abonnements auf dem besseren Pfad gefolgt von einer Kündigung auf dem alten Pfad geschehen. Allerdings wird durch die potentiell vielen zusätzlichen Abonnements und Kündigungen der Vorteil des geringen Kommunikationsoverheads aufgehoben. Auch die Komplexität des Algorithmus steigt dadurch signifikant. Es müssen, um einen Verlust von Nachrichten bei der Pfadverbesserung zu vermeiden, zusätzliche Informationen gespeichert und Entscheidungslogiken etabliert werden, um Laufzeitunterschiede zwischen neuem Abonnement und Kündigung des alten Pfades erkennen zu können.

5.3. Pfadauswahl in Phase 3

Der Mehr-Wege-Ansatz nutzt die Phase 2 zum Markieren möglicher Pfade und sortiert diese nach ihren Pfadkosten. Dabei kennt jede Ecke, die sich auf einem möglichen Pfad befindet, die Nachbarn, über die die Senke erreichbar ist und die bestmöglichen Pfadkosten bis zur Senke beim Versand über jeden dieser Nachbarn. Entsprechend kann keine Ecke je wissen, ob sie selbst Teil des besten Pfades ist, wohl aber über welchen Nachbarn der von ihr aus beste

Pfad weitergeht. Anhand dieser Informationen werden in der dritten Phase die Nachrichten auf dem aktuell besten bekannten Pfad zur Senke übertragen.

Bricht ein Pfad zusammen, kann jederzeit über einen schlechteren Pfad ausgewichen werden. Erst wenn kein Pfad mehr verfügbar ist, muss unter Umständen reagiert werden. Nachfolgend werden der Umgang mit Ankündigungen zur Generierung zyklenfreier Pfade, das Markieren möglicher Pfade durch Abonnements, so dass die besten Kosten zur Senke stets korrekt sind, und der Umgang mit dem Zusammenbruch von Pfaden erklärt. Im Anschluss daran werden Vor- und Nachteile dieses Ansatzes diskutiert.

5.3.1. Umgang mit Ankündigungen

Die Verteilung von Ankündigungen erfolgt ebenfalls, wie in Abschnitt 5.1.3 erklärt, durch Speicherung und Weiterleitung der ersten empfangenen Ankündigung Ank_0. Da das Ziel des Mehr-Wege-Ansatzes ist, möglichst viele Pfade zur Verfügung zu haben, um daraus später den optimalen Pfad auszuwählen, werden von allen empfangenen Ankündigungen Ank_i deren Absender in $absd_i$ und Pfadkostenwert $Ank_i.cost$ in $cost_i$ gespeichert, so lange $Ank_i.cost$ kleiner ist als die Schwelle *mincost*. *mincost* ergibt sich als der kleinste Wert von $Ank_0.cost$ nach dessen Weiterleitung.

Wird also Ank_0 erstmalig empfangen und entsprechend weitergeleitet, so wird *mincost* auf das Minimum aller $Ank_0.cost$ nach deren Weiterleitung festgelegt. Im Anschluss werden nur noch Ankündigungen akzeptiert, deren Pfadkostenwert kleiner ist als *mincost*. Wird Ank_0 zu einem späteren Zeitpunkt noch einmal weitergeleitet, weil neue benachbarte Ecken bekannt werden, so wird *mincost* wenn nötig angepasst und alle akzeptierten Ankündigungen, deren gespeicherte Pfadkostenwerte größer gleich dem neuen Wert von *mincost* sind, gelöscht. Auf diese Weise wird sichergestellt, dass für den gesamten Verteilungsgraphen der Ankündigungen zu jedem Zeitpunkt gilt, dass die Pfadkosten aller akzeptierten Ankündigungen einer Ecke kleiner sind als die geringsten Pfadkosten der von dieser Ecke weitergeleiteten Ankündigung Ank_0 nach deren Weiterleitung.

Nimm man nun wiederum an, dass in dem Verteilungsgraphen nach dem beschriebenen Algorithmus ein Zyklus enthalten ist, so muss der Pfadkostenwert c einer der Ankündigungen, die zur Etablierung des Zyklus versandt und akzeptiert wurden größer oder gleich der Pfadkostenwerte aller anderen dieser Ankündigungen sein. Der Empfänger e dieser Ankündigung muss auch die zur Etablierung der nächsten im Zyklus liegenden Verbindung nötige Ankündigung versandt haben. Deren Pfadkostenwert sei k und für ihn muss dementsprechend $k \leq c$ gelten. Da k ein Pfadkostenwert einer von e versandten Ankündigung ist, muss für den Wert von *mincost* für e gelten: $mincost \leq k$. Demzufolge muss auch gelten: $c \geq mincost$. Die Ecke e hätte also laut Algorithmus die Ankündigung mit dem Kostenwert c nie akzeptieren dürfen und somit wäre der angenommene Zyklus nie entstanden. Der Verteilungsgraph der Ankündigungen für den Mehr-Wege-Ansatz ist demzufolge zyklenfrei.

Abbildung 5.5 stellt beispielhaft eine Ecke in Phase eins dar.

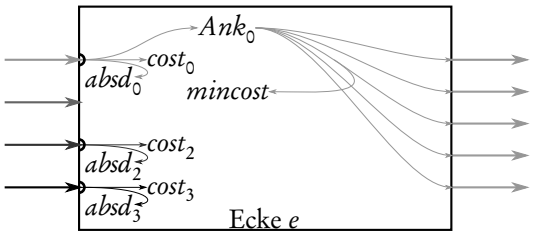


Abbildung 5.5.:
Graphische Repräsentation einer Ecke nach dem Empfang von vier Ankündigungen. Die in der Ecke notierten Werte werden dort gespeichert. Die Transparenz der Pfeile entspricht dem Zeitpunkt der jeweiligen Aktion, je transparenter desto früher. Es wurden vier Ankündigungen von verschiedenen Absendern empfangen. Die erste wurde gespeichert und an alle fünf möglichen Empfänger weitergeleitet. Dabei wurden die minimalen Kosten ermittelt und gespeichert. Von den im Anschluss empfangenen drei Ankündigungen wurden die Absender und Kosten der beiden Ankündigungen gespeichert, deren Kosten geringer als *mincost* waren.

5.3.2. Markieren möglicher Pfade

Der Versand der Abonnements erfolgt bei der Senke beginnend im Verteilungsgraph der Ankündigungen in entgegengesetzte Richtung. Das heißt für jede Ecke, die ein Abonnement empfängt, dass sie es an alle gespeicherten Absender $absd_i$ weiterleiten muss. Im Abonnement wird für den Mehr-Wege-Ansatz neben der SID (*sid*) noch ein Wert für die Pfadkosten (*kost*) gespeichert. Aus diesem und dem lokal gespeicherten Pfadkostenwert $cost_0$ kann jede Ecke die Kosten des kostengünstigsten Pfades über die Ecke, von der das entsprechende Abonnement empfangen wurde, bis zur Senke berechnen. Dazu wird wie folgt vorgegangen:

Die Senke initialisiert den Wert für *kost* im Abonnement für jede gespeicherte Ecke $absd_i$, an die sie das Abonnement weiterleitet, jeweils mit dem in Phase 1 gespeicherten Wert $cost_i$. Die Pfadkosten von der Senke zum Konsumenten betragen 0, da der Konsument direkt an der Senke angemeldet ist.

Für eine beliebige Ecke e unter denen, die eines dieser Abonnements empfangen, gilt, dass sie ursprünglich ihre Ankündigung Ank_0 an die Senke s weitergeleitet hat. Demzufolge muss die Differenz zwischen dem Wert *kost* aus dem von s empfangenen Abonnement und dem gespeicherten $cost_0$ genau die Pfadkosten $pathcost_s$ zwischen der Ecke und der Senke ergeben: $pathcost_s = Abo_s.kost - cost_0$. Die Ecke e leitet nun wiederum das Abonnement an jeden ihrer lokal gespeicherten $absd_i$ jeweils mit den Pfadkosten $pathcosts_s + cost_i$ weiter.

Eine beliebige Ecke e' unter denen, die eines der von e versandten Abonnements empfangen, hat ursprünglich ihre Ankündigung Ank_0 an e versandt. Da die Differenz zwischen dem auf e gespeicherten $cost_i$ aus dieser Ankündigung, wie sie von e empfangen wurde, und $cost_0$ auf e', mit denen sie dort abgeschickt wurde, genau den Verbindungskosten x zwischen e' und e entspricht, gilt: $pathcost_e = x + pathcost_s = Abo_e.kost - cost_0$. Dies gilt entsprechend für

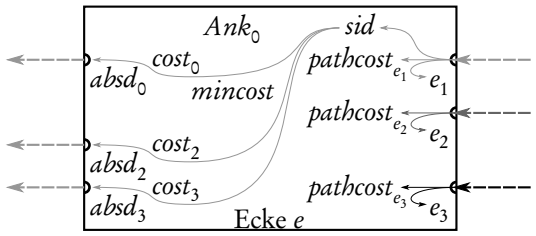


Abbildung 5.6.:
Graphische Repräsentation einer Ecke nach dem Empfang von drei Abonnements für die gleiche SID. Auf den Empfang des ersten hin, wurde entsprechend dem Algorithmus an alle drei gespeicherte Absender ein neues Abonnement versandt. Von allen Abonnements wurden deren Absender und die Pfadkosten bis zur Senke gespeichert.

jeden weiteren Schritt der Weiterleitung von Abonnements bis zur Quelle. Eine jede Ecke f, die von einer Ecke e ein Abonnement Abo_e erhält, kann die Pfadkosten des Pfades über e bis zur Senke berechnen als $pathcost_e = Abo_e.kost - cost_0$.

Gespeichert werden für jede SID, die Ecken e_j, von denen ein Abonnement mit $sid = \text{SID}$ empfangen wurde und zu jeder dieser Ecken die berechneten Pfadkosten $pathcost_{e_j}$. Jedes dieser Paare aus Ecke und Pfadkosten repräsentiert einen möglichen Pfad zur Senke. Abbildung 5.6 veranschaulicht diesen Prozess.

Erhalt des optimalen Pfades beim Empfang mehrerer Abonnements

Wird, wie zuvor beschrieben, immer nur auf das erste empfangene Abonnement hin ein Abonnement an die bekannten Absender von Ankündigungen versandt, so würde auf einem Pfad zwischen den Ecken $e \ldots f \ldots s$ eine Pfadverbesserung zwischen f und s für e unbemerkt bleiben. Um dies zu vermeiden muss auf eine Verbesserung der Pfadkosten zur Senke aufgrund eines empfangenen Abonnements mit einem weiteren Abonnement an alle Absender von Ankündigungen reagiert werden. So werden bessere Pfade in Richtung Quellen propagiert und können entsprechend benutzt werden.

Eine Ausnahme bildet hier die Aggregation mehrerer unterschiedlicher Ankündigungen zu einer einzigen wie sie in Abbildung 5.7 skizziert wird. In diesem Fall werden vom Verarbeiter v die beiden Abonnements von den Ecken e (Abo_e) und f (Abo_f) empfangen, in ein neues Abonnement (Abo_v) umgewandelt und dieses an die Ecke g weitergeleitet. Die Pfadkosten verhalten sich in diesem Fall entsprechend der in Abschnitt 4.5.3 beschriebenen Metrik wie folgt: $Abo_v.cost = Abo_e.cost + Abo_f.cost + w$. Dazu kommt, dass w größenabhängig und somit eigentlich eine Funktion von $Abo_e.size$ und $Abo_f.size$ ist.

Es wird deutlich, dass die Verbindungen $e \rightarrow g$ sowie $f \rightarrow g$ nicht unabhängig voneinander betrachtet werden können. Sobald sich $Abo_f.cost$, also der Pfad zwischen der Quelle und f verändert, hat dies Einfluss auf die Verbindung zwischen e und g, also auf die Pfadkosten zwischen e und der Senke. Im Fall aggregierender Nachrichtenverarbeitung muss somit eine

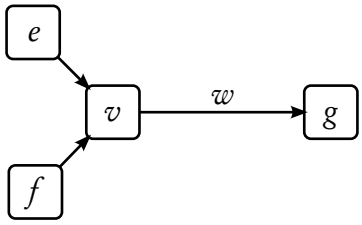


Abbildung 5.7:
Schematische Darstellung einer Situation mit Nachrichtenaggregation. Die von den Ecken e und f versandten Ankündigungen werden vom Verarbeiter v aggregiert und an die Ecke g übertragen.

Pfadverbesserung, die zwischen der Quelle und f entstanden ist, auch in Richtung g propagiert werden, da durch die Verarbeitung diese Pfadkosten zu einer Verbesserung des Schrittes $e \rightarrow g$ und somit des Pfades zwischen e und der Senke führen.

Um diesem Spezialfall gerecht zu werden, muss die Existenz eines aggregierenden Verarbeiters in den Abonnements vermerkt werden. Ist eine lokale Pfadverbesserung auf einer Ecke entstanden, die ein solches Abonnement erhalten hat oder erhält, muss sie diese Pfadverbesserung zusätzlich vorwärts in Richtung des Verarbeiters propagieren.

Empfang weiterer Ankündigungen

Wird nach dem Empfang und der Weiterleitung des ersten Abonnements eine weitere Ankündigungen Ank_i empfangen, akzeptiert und dadurch ein neuer Absender $absd_i$ bekannt, so wird für jede durch Abonnements bekannte SID ein Abonnement generiert, dessen Kosten sich aus der Differenz zwischen den geringsten für diese SID gespeicherten Pfadkosten und $cost_i$ errechnen, und an $absd_i$ versandt. Dies ist häufig der Fall, wenn sich die Vermittlungstopologie durch die Mobilität einzelner Geräte verändert.

Werden durch das Bekanntwerden neuer Verbindungen weitere Ankündigungen versandt und *mincost* verringert sich, so dass in der Folge Absender ungültig werden, an die bereits ein Abonnement versandt wurde, so müssen diese Abonnements widerrufen werden, um die Zyklenfreiheit des Verteilungsgraphen zu erhalten. Dies kann problemlos durch Störungsmeldungen geschehen, wie sie in Abschnitt 5.3.3 eingeführt werden.

Ein Nebeneffekt dieser Reaktion auf Topologieveränderungen ist es, dass in stark veränderlichen Topologien nicht nur besonders viele Pfade zerstört werden, sondern unter günstigen Umständen auch neue Pfade entstehen. Dies kann dazu führen, dass eine Senke, für die kein Pfad mehr verfügbar ist, allein durch die Veränderlichkeit der Topologie wieder an die Quelle angebunden wird. Dieser Effekt wird im Abschnitt 5.3.3 noch einmal aufgegriffen, wenn es um die verzögerte Neuankündigung geht. Die erwähnten günstigen Umstände beziehen sich auf die Prüfung der empfangenen Ankündigungen gegen *mincost* um Zyklen im Verteilungsgraphen zu vermeiden. Es entstehen somit nur solche Pfade neu, die an keinem Punkt wesentlich schlechter sind als zuvor bereits etablierte Pfade.

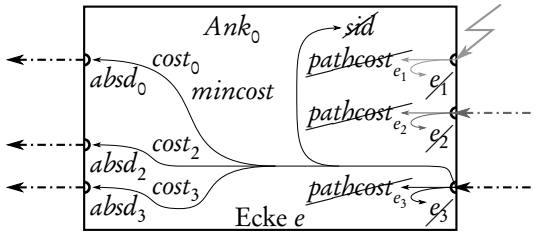


Abbildung 5.8.:
Beispielhafte Darstellung einer Ecke bei Pfadverlust. Die Verbindung zur Ecke e_1 ist verloren gegangen. Daraufhin ist von e_2 eine Störungsmeldung eingegangen und im Anschluss auch von e_3. Bei jeder diese Aktionen wurde der entsprechende Pfad gelöscht. Bei Empfang der letzten Störungsmeldung wird diese außerdem an alle Absender weitergeleitet und jede Referenz auf dieses Abonnement gelöscht.

Kündigung eines Abonnements

Empfängt eine Ecke eine Kündigung für ein Abonnement, so muss sie diese an alle gespeicherten Absender $absd_i$ weiterleiten und damit alle markierten Pfade ungültig machen. Eine Kündigung hat somit den Versand relativ vieler, sehr kleiner Netzwerkpakete zur Folge. Ungeachtet dessen ist es wahrscheinlich effizienter, die Kündigung von Abonnements zu tolerieren, als mit dem Ziel, Kündigungen einzusparen, die Gültigkeit der Ankündigungen signifikant zu senken.

5.3.3. Vorgehen bei Pfadverlust

Die Reaktion einer Ecke auf den Verlust der Verbindung zu einer auf dem Pfad folgenden Ecke hängt von der Verfügbarkeit anderer Pfade ab. In jedem Fall werden die Ecke e, zu der die Verbindung abgebrochen ist, und die mit ihr verbundenen Pfadkosten $pathcost_e$ zur Senke aus dem Speicher der Ecke gelöscht. Wurde ein Abonnement mit der gleichen SID von mehr als dieser einen Ecke empfangen und wenigstens eine der anderen Ecken ist noch erreichbar, so ist keine weitere Reaktion nötig weil noch wenigstens ein funktionierender Pfad verfügbar ist.

Erst wenn der letzte mögliche Pfad einer Ecke zur Senke für eine SID verloren geht, wird eine Störungsmeldung generiert und an alle dort gespeicherten Absender $absd_i$ verschickt. Alle Informationen zu dieser SID werden anschließend aus dem Speicher dieser Ecke gelöscht. Die Reaktion auf den Empfang einer Störungsmeldung durch eine Ecke entspricht der beschriebenen Reaktion auf den Verlust einer Verbindung. Abbildung 5.8 veranschaulicht diesen Vorgang.

Wenn bei der Quelle für den letzten dort vorhandenen Pfad für eine SID eine Störungsmeldung empfangen wird oder die Verbindung zur nächsten Ecke auf dem Pfad verloren geht, dann ist eine Reaktion nötig. Wie bereits in Abschnitt 5.3.1 erwähnt, besteht die Möglich-

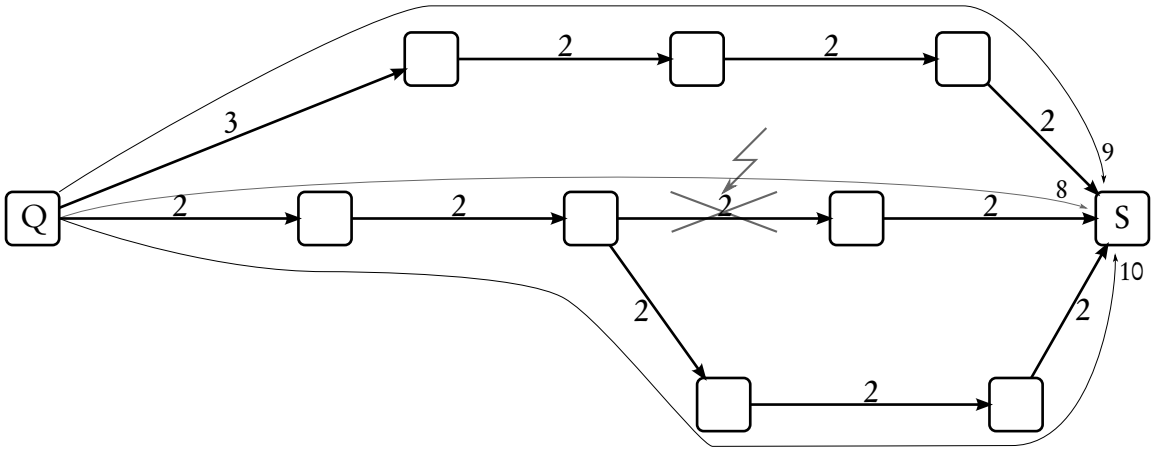


Abbildung 5.9.:
Zwischen der Quelle Q und der Senke S hat sich der durch die dicken Pfeile dargestellte Verteilungsgraph der Ankündigungen mit den angegebenen Verbindungskosten etabliert. Daraus resultiert der dargestellte Pfad mit den Kosten 8. Wird die markierte Verbindung nun zerstört, so wird ohne Information der Absender auf den Pfad mit den Kosten 10 ausgewichen. Insgesamt optimal wäre aber der Pfad mit den Kosten 9.

keit, dass auch wenn alle Pfade zwischen Quelle und Senke nicht mehr verfügbar sind, durch die Reaktion des Algorithmus auf Topologieveränderungen neue Pfade entstehen. Daher ist es nicht zweckmäßig, dass die Quelle sofort mit einer Neuankündigung reagiert.

Als Auslöser für eine Neuankündigung nach einem Pfadverlust eignen sich mehrere aufeinander folgende Pfadverluste für verschiedene SIDs, da dies auf eine starke Veränderung der Topologie schließen lässt, oder die Veröffentlichung so vieler Nachrichten, dass eine Sicherstellung der Zustellung durch die Zwischenspeicherung in den Ecken nicht mehr erfolgen kann.

Verlust optimaler Pfade

Das zuvor beschriebene Vorgehen stellt die effizienteste Reaktionsmöglichkeit auf den Verlust von Pfaden dar, hat allerdings einen Nachteil. Wenn eine Ecke den kostengünstigsten verfügbaren Pfad verliert und nicht darauf reagiert, weil es nicht der letzte verfügbare Pfad war, so wird die Nichtverfügbarkeit dieses Pfades auch nicht in Richtung Quelle propagiert.

Dieses Vorgehen führt dazu, dass bei einem Fehler auf dem insgesamt kostengünstigsten Pfad dieser weiterhin bis zur Fehlerstelle benutzt und erst dort auf den nächstschlechteren Pfad ausgewichen wird. Der dadurch entstehende Pfad muss nicht dem nun insgesamt kostengünstigsten Pfad entsprechen. Abbildung 5.9 veranschaulicht dieses Problem.

Um sicherzustellen, dass immer der insgesamt kostengünstigste Pfad benutzt wird, muss der Verlust eines lokal kostengünstigsten Pfades entsprechend der zuvor für die Verteilung

neuer kostengünstigster Abonnements beschriebenen Weiterleitungsmechanismen propagiert werden.

5.3.4. Diskussion

Der Mehr-Wege-Ansatz hat gegenüber dem zuvor beschriebenen Ein-Wege-Ansatz zwei Vorteile. Zwischen Quelle und Senke wird ebenfalls sofort ein Pfad etabliert, auf dem Nachrichten versandt werden können. Allerdings werden beim Mehr-Wege-Ansatz bessere Pfade, die durch die Verteilung der Ankündigungen gefunden werden, sofort markiert und anschließend auch für die Verteilung von Nachrichten genutzt. Der Mehr-Wege-Ansatz stellt also sicher, dass für den Versand von Nachrichten ständig der aktuell kostengünstigste bekannte Pfad benutzt wird. Je nach Wahl der Metrik kann dies bedeuten, dass die Bandbreite effizienter genutzt wird, die Nachrichten schneller an der Senke ankommen oder andere Qualitätsmerkmale der Verbindung optimiert werden.

Der zweite Vorteil ist, dass bei Verlust eines Pfades nicht sofort eine neue Ankündigung verteilt werden muss. So lange im Verteilungsgraphen der Ankündigungen Pfade verfügbar sind, werden diese auch benutzt. Selbst wenn alle Pfade nicht mehr verfügbar sind, besteht eine Chance, dass sich selbständig ein weiterer Pfad etabliert und eine Neuankündigung ebenfalls vermieden werden kann. Eine Neuankündigung bei Pfadverlust ist also im Mehr-Wege-Ansatz die Ausnahme, während sie im Ein-Wege-Ansatz die einzige Möglichkeit zur Fortsetzung der Nachrichtenvermittlung war. Das Einsparen von zusätzlichen Ankündigungsphasen hat eine deutliche Reduzierung des Kommunikationsoverheads zur Folge, da die Ankündigungsphase durch die Nutzung des Flooding-Algorithmus für einen wesentlichen Anteil am Gesamt-Kommunikationsoverhead des Algorithmus verantwortlich ist.

Den beiden Vorteilen stehen geringfügige Nachteile gegenüber. Durch den Versand von mehr Abonnements, um alle verfügbaren Pfade zu markieren, steigt der Kommunikationsoverhead leicht an. Außerdem erhöht sich die Netzwerklast bei der Kündigung von Abonnements, weil nun alle Pfade deaktiviert werden müssen.

Entsprechend der Gegebenheiten der Vermittlungstopologie kann demzufolge der Mehr-Wege-Ansatz deutliche Vorteile bezüglich der vom Algorithmus erzeugten Netzwerklast haben. Insbesondere in stark veränderlichen Topologien, in denen eher häufig mit der Zerstörung von Verbindungen zwischen Geräten zu rechnen ist, sollte der Mehr-Wege-Ansatz dem Ein-Wege-Ansatz überlegen sein.

Ausnutzung der Mehr-Wege-Eigenschaft

Ein weiterer Vorteil des Mehr-Wege-Ansatzes ist, dass in der Phase des Nachrichtenversandes alle verfügbaren Pfade bekannt sind. Somit ist nicht nur die Auswahl eines optimalen Pfades möglich, sondern auch die Verteilung der Nachrichten auf mehrere verfügbare Pfade.

Allerdings ist dieser Vorteil mit Einschränkungen verbunden. Wie bereits in Abschnitt 4.3 beschrieben, müssen Nachrichten unter Umständen auf ihrem Pfad von der Quelle zur Senke mehrfach defragmentiert werden, um verarbeitet werden zu können. Zur Defragmentierung ist es notwendig, dass alle Nachrichtenfragmente auch verfügbar sind. Eine Lastverteilung über mehrere Pfade, die Nachrichtenverarbeitung beinhalten, ist also nur dann möglich, wenn sichergestellt wird, dass an Verarbeitern auch alle für die Nachrichtenverarbeitung notwendigen Nachrichtenfragmente ankommen. Das gleiche gilt für zustandsbehaftete Verarbeiter, die alle Nachrichten der Ausgangstypen empfangen müssen.

5.4. Diskussion

Im folgenden Abschnitt werden zentrale Eigenschaften der entwickelten Algorithmen diskutiert. Dazu gehören die Komplexität und Skalierungsfähigkeit der Algorithmen, der zu erwartende Einfluss von Parametern sowie deren Zuordnung zu bestimmten Anwendungsklassen. Die dabei aufgestellten Thesen bilden die Grundlage für die anschließende simulative Evaluierung der Algorithmen.

5.4.1. Komplexitätsabschätzung der entwickelten Algorithmen

Zwei Komplexitätseigenschaften sind für die Kommunikationsalgorithmen von besonderer Bedeutung. Dies sind der entstehende Kommunikationsaufwand sowie der Speicherbedarf auf den Geräten in Abhängigkeit von der Anzahl und Größe der zu übertragenden Nachrichten sowie den zentralen Eigenschaften der Vermittlungstopologie und des Algorithmus.

Kommunikationsaufwand

Der Kommunikationsaufwand reflektiert die Menge der zwischen den Geräten übertragenen Daten. Zur Berechnung des Kommunikationsaufwandes wird zwischen den drei Phasen Ankündigung, Abonnement und Nachrichtenversand unterschieden. Der Aufwand jeder Phase wird in Tabelle 5.2 getrennt nach der Größe s der in der Phase übertragenen Nachricht, der Häufigkeit h der Initiierung einer Übertragung und der Anzahl a der für die Übertragung der Nachricht nötigen Kopien, die zwischen zwei benachbarten Geräten übertragen werden müssen, bewertet. Der Kommunikationsaufwand pro Phase ergibt sich dann als $s \cdot h \cdot a$ bzw. der gesamte Kommunikationsaufwand als $s_{Ank}h_{Ank}a_{Ank} + s_{Abo}h_{Abo}a_{Abo} + s_N h_N a_N$.

Für die Berechnung sind Größen von Bedeutung, die vom Szenario abhängen. Dazu gehören die Größen s_{Ank} und s_N von Ankündigungen und Nachrichten, die im Wesentlichen durch ihre jeweilige Nutzlast definiert werden. Außerdem gehören dazu die Anzahl $|E|$, $|K|$, q und v der Ecken, Kanten, Quellen und Verarbeiter sowie die Zeit t_N zwischen der Veröffentlichung zweier Nachrichten, die Menge *SID* der Senken, die Menge P_{SID} der möglichen Pfade zu jeder Senke, sowie der Faktor x, der angibt, wie stark sich diese Pfade überlappen.

Phase	Ankündigung	Abonnement	Nachricht
Nachrichtengröße	s_{Ank}	s_{Abo}	s_N
Häufigkeit	$h_{Ank} = 1/\min(t_g, t_p)$	$h_{Abo} = 1/\min(t_g, t_p)$	$h_N = 1/t_N$
Ein-Wege-Ansatz	$a_{Ank} = \lvert K\rvert - \lvert E\rvert + q + v$	$a_{Abo} = \sum_{j \in SID} \lvert p_j\rvert$	$a_N = x \sum_{j \in SID} \lvert p_j\rvert$
Mehr-Wege-Ansatz	$a_{Ank} = \lvert K\rvert - \lvert E\rvert + q + v$	$a_{Abo} \leq \sum_{j \in SID} \sum_{p \in P_j} \lvert p\rvert$	$a_N = x \sum_{j \in SID} \lvert p_j\rvert$

Tabelle 5.2.:
Übersicht über den Kommunikationsaufwand für die beiden Algorithmen unterteilt in die jeweiligen Phasen.

Im Wesentlichen vom Algorithmus abhängig sind die Größe s_{Abo} von Abonnements sowie der Gültigkeitszeitraum t_g der Ankündigungen, der als einflussreicher Parameter im Abschnitt 5.4.3 noch diskutiert wird. Die zwei von der Leistungsfähigkeit des Algorithmus in dem jeweiligen Szenario abhängigen Größen, die sich auch zwischen dem Ein-Wege- und dem Mehr-Wege-Ansatz deutlich unterscheiden, sind die Längen p_{SID} der besten zu jeder Senke gefundenen Pfade sowie die Zeit t_p bis einer Quelle alle Pfade zu einer Senke verloren gegangen sind.

Zur Vereinfachung der Berechnung wird davon ausgegangen, dass alle Zeiten (t_N, t_g und t_p) unabhängig von der Quelle und dem Zeitpunkt sind. Dies ist nicht realistisch, genügt aber, um einen Eindruck von deren Einfluss auf die Komplexität zu erhalten.

Aus der Übersicht lässt sich klar erkennen, dass der Mehr-Wege-Ansatz in der Phase des Versandes der Abonnements ein deutlich höheres Kommunikationsaufkommen verursacht. Anstelle der Summe der Pfadlängen der jeweils zu dem Zeitpunkt kürzesten Pfade zu jeder Senke wird ein Abonnement auf allen durch den Verteilungsgraphen der Ankündigungen überdeckten Pfaden zu allen Senken übertragen.

Da der Mehr-Wege-Ansatz zu jedem Zeitpunkt den optimalen im Verteilungsgraphen der Ankündigungen vorhandenen Pfad benutzt, während der Ein-Wege-Ansatz an den zum Zeitpunkt des Pfadaufbaus bekannten optimalen Pfad gebunden ist, ist davon auszugehen, dass für jede SID das $|p_j|$ für den Mehr-Wege-Ansatz kleiner oder maximal gleich dem $|p_j|$ für den Ein-Wege-Ansatz ist. Voraussetzung hierfür ist, dass die Metrik zur Pfadbewertung, die Pfadlänge mit einbezieht.

Weiterhin ist davon auszugehen, dass mit dem Mehr-Wege-Ansatz insbesondere in veränderlichen Topologien eine Quelle seltener einen Verlust aller Pfade zu einer Senke erleidet. Dies lässt sich darin ausdrücken, dass t_p für den Mehr-Wege-Ansatz deutlich größer ist als für den Ein-Wege-Ansatz.

Diese beiden Unterschiede wirken bezogen auf den Kommunikationsaufwand der Tatsache entgegen, dass der Mehr-Wege-Ansatz deutlich mehr Abonnements versenden muss.

Phase	Anzahl	Ein-Wege-Ansatz	Mehr-Wege-Ansatz
Ankündigung	Anzahl der Nachrichtentypen = Zahl lokaler Ecken	*aid* + Nutzlast + $cost'$ + $absd'$	*aid* + Nutzlast + *mincost* + $(cost_i + absd_i)$ · Zahl akzeptierter Ankündigungen
Abonnement	Zahl lokaler Ecken · Gesamtzahl der Senken	(falls Ecke $e \in p_{sid}$) *sid* + *next* + *last*	(falls Ecke $e \in P_{sid}$) *sid* + Nutzlast + (e + $pathcost_e$) · Anzahl der Pfade von e zur Senke
Nachricht	Zwischenspeicher für Nachrichten		

Tabelle 5.3.:
Übersicht über den Speicherbedarf für die beiden Algorithmen unterteilt in die jeweiligen Phasen.

Speicherbedarf

Der Speicherbedarf wird pro Gerät ermittelt. Auch hier wird zwischen den drei Phasen unterschieden. Tabelle 5.3 gibt einen Überblick über die Anzahl der gespeicherten Datensätze und die darin enthaltenen Daten. Es muss wiederum für jede Phase diese Anzahl mit der Größe der entsprechenden Daten multipliziert werden.

Für die Phase der Ankündigung wird bei beiden Ansätzen pro lokal verwalteter Ecke die AID und die Nutzlast der Ankündigung gespeichert. Beim Ein-Wege-Ansatz wird zusätzlich der Pfadkostenwert und der Absender der kostengünstigsten Ankündigung gespeichert, während beim Mehr-Wege-Ansatz die Kostengrenze sowie von jeder akzeptierten Ankündigung Pfadkostenwert und Absender gespeichert werden.

In der Phase der Abonnements wird vom Ein-Wege-Ansatz pro global vorhandener Senke und lokaler Ecke, die auf dem ausgewählten Pfad zu dieser Senke liegt, die SID sowie der Absender und Empfänger der Ankündigung gespeichert. Der Mehr-Wege-Ansatz speichert pro global vorhandener Senke und lokaler Ecke, die auf einem der Pfade zu dieser Senke liegt, die SID, die Nutzlast des Abonnements und für jeden Pfad, der von dieser Ecke zur Senke führt, den nächsten Schritt sowie die damit verbundenen Pfadkosten zur Senke.

In der letzten Phase, in der die Nachrichten auf den entsprechenden Pfaden versandt werden, sind bei beiden Ansätzen im Prinzip identische Mechanismen zur Zwischenspeicherung von Nachrichten nötig. Zusätzlich sind beim Mehr-Wege-Ansatz bei der Realisierung von Lastverteilung zusätzliche Statusinformationen nötig, um sicherzustellen, dass die Nachrichten auf den richtigen Pfaden versandt werden.

Der Speicherbedarf des Mehr-Wege-Ansatzes ist somit in jedem Fall größer als der des Ein-Wege-Ansatzes, da zur Realisierung des Routings wesentlich mehr Statusinformationen nötig sind.

Schlussfolgerungen

Der Speicherbedarf hängt jeweils ausschließlich von Eigenschaften der Topologie, wie der Anzahl der Nachrichtentypen, der Anzahl der Senken und der Anzahl möglicher Pfade, ab. Er ist für den Ein-Wege-Ansatz in den Phasen Ankündigung und Abonnement immer deutlich unterhalb des Speicherbedarfs für den Mehr-Wege-Ansatz.

Beim Kommunikationsaufwand kann eine derart eindeutige Aussage nicht getroffen werden. In der Phase der Abonnements führt der Mehr-Wege-Ansatz in Topologien mit mehr als einem möglichen Pfad vermutlich immer zu mehr Kommunikationsaufwand, weil alle Pfade durch Abonnements freigeschaltet und gegebenenfalls auch später durch Kündigungen wieder verworfen werden müssen. Abonnements sind aber die mit Abstand kleinsten Nachrichten, die die Algorithmen verbreiten.

Die Anzahl der Übermittlungen der etwas größeren und häufiger übertragenen Ankündigungen ist für beiden Ansätze identisch, deren Häufigkeit hängt aber von der Zeit zwischen einer Ankündigungsphase und dem Verlust eines Pfades bzw. so vieler Pfade, dass eine neue Ankündigungsphase nötig ist, ab. Eine starke Abhängigkeit vom gewählten Ansatz ist für diese Pfadnutzungsdauer t_p entsprechend der Ausführungen in Abschnitt 5.3.3 zu erwarten. t_p müsste in den meisten Szenarien für den Mehr-Wege-Ansatz wesentlich länger sein und somit den Kommunikationsaufwand stark verringern.

In vielen Fällen wesentlich größer sind die in der dritten Phase übertragenen Nachrichten. Die Anzahl der Übermittlungen hängt hier vor allem von der Pfadlänge $|p_j|$ des gewählten Pfades ab. Abhängig von der Korrelation zwischen der Pfadgüte durch die gewählte Metrik und der Länge eines Pfades, ist durch die ständige Nutzung optimaler verfügbarer Pfade durch den Mehr-Wege-Ansatz hier mit einer Reduktion des Kommunikationsaufwandes durch kürzere Pfade zu rechnen. Selbst wenn die Metrik auf anderen Dienstgütemerkmalen basiert und nicht oder negativ mit der Pfadlänge korreliert, so bietet der Mehr-Wege-Ansatz hier zwar nicht den Vorteil des geringeren Kommunikationsaufwandes, aber statt dessen den Vorteil der besseren Dienstgüte.

Um diese theoretische Betrachtung zu untermauern, ist die Evaluation der beiden folgenden Aussagen notwendig:

- Die Pfadnutzungsdauer in veränderlichen Topologien ist für den Mehr-Wege-Ansatz signifikant länger als für den Ein-Wege-Ansatz.
- Die Länge der gewählten Pfade ist bei Wahl einer Metrik, die mit der Pfadlänge positiv korreliert, für den Mehr-Wege-Ansatz erkennbar kleiner als für den Ein-Wege-Ansatz.

Eine solche Metrik ist die im Abschnitt 4.5 vorgestellte, da sie auf der Übertragungszeit zwischen benachbarten Ecken basiert.

5.4.2. Skalierungsfähigkeit der Algorithmen

Für die Skalierbarkeit der Algorithmen ist die Abhängigkeit von den wesentlichen Größen der Umgebung von Interesse. Diese lassen sich in vier Gruppen unterteilen:

- Aus der Netzwerktopologie entstammen die Anzahl $|G|$ der Geräte sowie die Anzahl $|B|$ der gerichteten Kommunikationsverbindungen.
- Die Verarbeitungstopologie bestimmt die Anzahl $|T|$ der zusammenhängenden Nachrichtentypen.
- Erst in der daraus generierten Vermittlungstopologie treten die Anzahl q und v der auf Geräten installierten Instanzen von Quellen und Verarbeitern zu Tage sowie die Anzahl $|P|$ der möglichen Pfade zwischen Quellen und Verarbeitern.
- Darüber hinaus können in derselben Geräteinfrastruktur durch weitere, nicht zusammenhängende Verarbeitungstopologien, parallel andere Vermittlungstopologien entstehen. Diese erzeugen ebenfalls Kommunikationsaufwand und Speicherbedarf auf den Geräten.

Der Speicherbedarf der beiden Algorithmen wird im Wesentlichen durch zwei der beschriebenen Umgebungseigenschaften beeinflusst. Es existiert eine lineare Abhängigkeit von der Anzahl der Nachrichtentypen in Bezug auf die Anzahl der pro Gerät zu verwaltenden Routingtabellen für Abonnements. Außerdem hängt die Anzahl der zu verwaltenden Routingtabellen für Nachrichten von der Anzahl und Länge der in der Vermittlungstopologie vorhandenen Pfade zwischen Quellen und Senken ab. Die Länge der jeweiligen Routingtabellen ist für den Ein-Wege-Ansatz auf 1 und für den Mehr-Wege-Ansatz durch die Anzahl lokal benachbarter Ecken begrenzt und damit sehr überschaubar. In Bezug auf den Speicherbedarf pro Gerät skalieren beide Algorithmen hervorragend, wobei der Ein-Wege-Ansatz weniger Speicherbedarf erzeugt.

Kommunikationsaufwand

Der Kommunikationsaufwand hängt an vielen Stellen von den veränderlichen Größen der Umgebung ab. Eine Untergliederung in die drei Phasen ist hier sinnvoll.

Ankündigungen Die Ankündigungsphase hat den größten Einfluss in Bezug auf die Skalierbarkeit, weil hier die meisten Nachrichten versandt werden. Die Anzahl pro Ankündigungsphase lässt sich für beide Algorithmen mit $a_{Ank} = |K| - |E| + q + v$ beziffern. Diese Werte lassen sich entsprechend den Ausführungen in Abschnitt 4.2.1 auf konkrete Größenveränderungen in der Umgebung zurückführen. Die entsprechenden Zusammenhänge sind in Tabelle 5.4 dargestellt.

Außerdem lässt sich die Anzahl der Ankündigungen direkt in Abhängigkeit der entsprechenden Größen ausdrücken. Es ergibt sich $a_{Ank} = |T|(|B| - |G|) + q + 2v$. Betrachtet man zusätzlich das Vorhandensein einer zweiten Vermittlungstopologie in derselben Netzwerkto-

Veränderung	Einfluss auf Größen	Einfluss auf Anzahl
neue Quelle	$q+=1$	$a_{Ank}+=1$
neuer Verarbeiter	$v+=1$	$a_{Ank}+=2$
neuer Nachrichtentyp mit m neuen Verarbeiterinstanzen	$\|E\|+=\|G\|$, $\|K\|+=\|B\|+m$, $v+=m$	$a_{Ank}+=\|B\|-\|G\|+2m$
neue Kommunikationsverbindung	$\|K\|+=2\|T\|$	$a_{Ank}+=2\|T\|$
neues Gerät mit n Kommunikationsverbindungen:	$\|E\|+=\|T\|$, $\|K\|+=2n\|T\|$	$a_{Ank}+=(2n-1)\|T\|$

Tabelle 5.4.:
Übersicht über die Veränderung des Kommunikationsaufwand in Abhängigkeit von den zentralen Größen der Umgebung.

pologie mit den Nachrichtentypen T', sowie der Anzahl q' und v' an Quellen und Verarbeiterinstanzen, so ergibt sich $a_{Ank} = (|T|+|T'|)(|B|-|G|)+q+q'+2(v+v')$. Die Anzahl der nicht zusammenhängenden Vermittlungstopologien ist somit irrelevant. Es ist nur die Anzahl insgesamt vorhandener Nachrichtenzustände, Quellen und Senken von Bedeutung.

Aus der formalen Darstellung der Anzahl der Ankündigungen in Abhängigkeit von den Größen der Umgebung und der entsprechenden Übersicht wird deutlich, dass eine geringe Abhängigkeit von der Anzahl der Quellen und Verarbeiterinstanzen existiert. Einen wesentlich größeren Einfluss haben die Anzahl der zusammenhängenden Nachrichtenzustände aus der Verarbeitungstopologie sowie die Anzahl der gerichteten Kommunikationsverbindungen abzüglich der Anzahl der Geräte aus der Netzwerktopologie. Ein weiterer interessanter Aspekt ist, dass die Anzahl der nicht zusammenhängenden Vermittlungstopologien, die aus nicht zusammenhängenden Verarbeitungstopologien entstehen, nicht von Bedeutung ist. Eine Entkopplung eines Verarbeiters durch eine Kombination aus Senke und Quelle, führt somit nicht zu einer Entlastung des Kommunikationsaufwandes in der Umgebung. Sie hat im Gegenteil durch die in Abschnitt 4 beschriebenen Nachteile einen höheren Kommunikationsaufwand zur Folge.

Abonnements Der Einfluss der Phase des Versandes der Abonnements auf die Skalierbarkeit des Algorithmus ist deutlich geringer, da Abonnements sehr kleine Nachrichten sind. Der Kommunikationsaufwand ist hier abhängig von der Anzahl verfügbarer Pfade und deren Länge. Die Anzahl der Pfade hängt von der Anzahl der Senken im Szenario ab. Mehr Senken bedeuten dabei also in jedem Fall mehr Abonnements. Mehr unterschiedliche Pfade zu diesen Senken bedeuten für den Mehr-Wege-Ansatz mehr Abonnements.

Nachrichten In der Phase des Nachrichtenversandes hängt der Kommunikationsaufwand im Wesentlichen von der Größe der Nachrichten und der Länge der gewählten Pfade ab.

Da in der Phase des Nachrichtenversandes kaum Overhead entsteht, spielt hier die Realisierbarkeit einer Übertragung in der vorhandenen Vermittlungstopologie eine größere Rolle als die Skalierbarkeit des Algorithmus. Solange der Algorithmus in den ersten beiden Phasen skaliert und mit Hilfe einer sinnvollen Metrik nutzbare Pfade findet, funktioniert auch die Übertragung der Nachrichten, außer natürlich die Übertragung der gewünschten Nachrichten ist in der gegebenen Umgebung gar nicht möglich.

Schlussfolgerungen

Die bestimmenden Faktoren für die Skalierbarkeit der Algorithmen sind die Anzahl der Nachrichtentypen und darauf folgend die Größe und Form der Netzwerktopologie ausgedrückt durch die Differenz aus der Anzahl der gerichteten Kommunikationsverbindungen und der Anzahl der Geräte. Faktoren mit geringerem Einfluss sind die Anzahl der Senken sowie der Quellen und Verarbeiter.

Das größte Optimierungspotential der Algorithmen steckt also in der Begrenzung der Anzahl der Nachrichtentypen und dem Umgang mit Netzwerktopologien mit sehr viel mehr Verbindungen als Geräten. Einige Möglichkeiten sollen nachfolgend aufgezeigt werden.

Begrenzung der Nachrichtentypen Die Anzahl der Nachrichtentypen lässt sich aus Sicht der Middleware schwer eingrenzen, da sie hauptsächlich durch die Anwendungen bestimmt wird. Eine Erweiterung der Middleware ist aber denkbar. Ansatzpunkt ist hier, dass ein Nachrichtentyp einem Gerät erst bekannt wird, wenn es entsprechende Ankündigungen erhält. Möglichkeiten, die Verbreitung von Ankündigungen sinnvoll einzuschränken sind Lokalität von Ankündigungen - Ankündigungen werden von der Quelle und Verarbeitern aus nicht sehr weit im Netzwerk verbreitet - oder Barrieren auf Geräten - Geräte an bestimmten Punkten im Netzwerk, zum Beispiel Firewalls, leiten bestimmte Ankündigungen nicht in angrenzende Netze weiter.

Umgang mit bestimmten Netzwerktopologien Die Algorithmen skalieren besonders schlecht in Netzwerktopologien mit deutlich mehr Kommunikationsverbindungen als Geräten. Hier bieten sich technologieabhängige Optimierungen an. Zum Beispiel in einem Ethernet LAN, in dem die Anzahl der Kommunikationsverbindungen bezogen auf die Gerätezahl maximal ist, ist davon auszugehen, dass die direkte Verbindung zwischen zwei Geräten in jedem Fall zu bevorzugen ist. Hier kann durch Ausnutzung der Möglichkeit eines sehr effizienten Ethernetbroadcasts anstelle der Weiterleitung an jeden einzelnen Nachbarn der Kommunikationsaufwand erheblich gesenkt werden.

In mobilen ad-hoc-Netzwerken mit einem hohen Grad an Vermaschung ist die Anzahl der Kommunikationsverbindungen ebenfalls deutlich größer als die der Geräte, was zu einem hohen Kommunikationsaufwand in der Ankündigungsphase führt. Hier bieten sich

wahrscheinlichkeitsgetriebene Ansätze wie das in Abschnitt 2.2.2 vorgestellte Gossiping an, um den Kommunikationsaufwand zu reduzieren.

Die beschriebenen und andere Optimierungen lassen sich im technologieabhängigen Teil der Netzwerkabstraktionsschicht des Brokers für den eigentlichen Algorithmus nahezu transparent realisieren.

5.4.3. Parameter und deren Einfluss

Die Algorithmen verwenden eine Reihe von Parametern, deren Werte Einfluss auf das Verhalten des jeweiligen Algorithmus haben. In diesem Abschnitt werden diese Parameter zusammengetragen und ihr Einfluss abgeschätzt. Dabei wird darauf eingegangen, welche Folgen eine Erhöhung bzw. Verringerung des Wertes des Parameters hat, welcher Wert vermutlich gut geeignet ist, welche Nebenwirkungen ein ungünstiger Wert hat und ob eine direkte oder indirekte Kontrolle des Parameters durch die Anwendungen sinnvoll sein kann.

Gültigkeitsdauer der Ankündigungen

Die Gültigkeitsdauer der Ankündigungen wird von der Quelle bestimmt und gibt die maximale Zeitspanne an, in der die Ankündigung gültig ist. Vor Ablauf dieser Zeitspanne muss die Ankündigung verlängert werden, falls die Quelle weiterhin aktiv bleiben möchte. Broker, die eine Ankündigung empfangen, gehen davon aus, dass für die Zeit der Gültigkeit auch Nachrichten von der Quelle veröffentlicht werden.

Die Gültigkeitsdauer hat einen wesentlichen Einfluss auf den Kommunikationsaufwand. Je länger Ankündigungen gültig sind, desto seltener müssen sie durch einen Flooding-Prozess verlängert werden. Längere Ankündigungsgültigkeit führt also zu geringerem Kommunikationsaufwand. Allerdings lässt sich die Häufigkeit der Ankündigungsphasen in einer veränderlichen Topologie nicht weiter senken, als die Pfadnutzungsdauer vorgibt. Eine längere Ankündigungsgültigkeit hat keinen Effekt.

Weiterhin macht eine längere Gültigkeit der Ankündigungen keinen Sinn wenn vom Produzenten gar keine entsprechenden Nachrichten mehr veröffentlicht werden. Zusätzlich gibt die Ankündigungsgültigkeit auch den Zeitraum vor, in dem eine Senke keine Nachrichten empfängt, wenn durch einen unvorhergesehenen Effekt ein Abonnement nicht funktioniert hat oder ein Pfadbruch von der Quelle nicht bemerkt wurde oder wenn zwei Senken die gleiche SID ausgewählt haben.

Die Gültigkeitsdauer der Ankündigungen sollte also so hoch wie möglich gewählt werden, aber ist durch bestimmte Anwendungskriterien in sinnvollen Schranken zu halten. Daher ist eine Kontrolle durch die Anwendung sinnvoll. Nur so können der Zeitraum der tatsächlichen Veröffentlichung von Nachrichten, sowie der maximale Zeitraum des Verzichtes auf Nachrichten im Worst-Case-Szenario abgeschätzt werden.

Anzahl zwischengespeicherter Nachrichten

Um dem Verlust von Nachrichten durch Topologieveränderungen entgegen zu wirken, werden Nachrichten auf den Geräten zwischengespeichert. Die Anzahl der dort gespeicherten Nachrichten wird durch diesen Parameter bestimmt. Die Speicherung zu weniger Nachrichten führt zum Verlust einzelner Nachrichten in veränderlichen Umgebungen. Die Speicherung zu vieler Nachrichten führt zur Verschwendung von Speicherplatz auf den Geräten.

Entsprechend dem vorhandenen Speicherplatz auf dem Gerät ist der für die Zwischenspeicherung von Nachrichten zu verwendende Speicherplatz sinnvoll zu begrenzen. Hierfür ist eine Kontrolle durch die Anwendung sinnvoll.

Gültigkeitsdauer der Nachrichten im Cache

Neben der Anzahl der von Brokern gespeicherten Nachrichten ist auch der Zeitraum der Speicherung ein sinnvoller Parameter. Genau wie bei der Anzahl gilt für die Dauer der Speicherung, je länger, desto weniger Nachrichtenverlust. Dabei ist es sinnvoll, die Dauer der Speicherung mit der Anzahl der gespeicherten Nachrichten abzustimmen.

Außerdem kann eine Speicherung von Nachrichten über einen längeren Zeitraum von der Anwendung unerwünscht sein. Zum Beispiel bei der Übertragung von Audio-Daten eines Telefongespräches werden Nachrichten, die länger als 150ms unterwegs sind, ohnehin vom Konsumenten verworfen(ITU, 2003, S.2). Hier ist also eine Speicherung von Nachrichten länger als 150ms nicht sinnvoll. Entsprechende Anwendungsvorgaben benötigen wiederum eine Kontrolle des Parameters durch die Anwendungen.

Anzahl der in Abonnements referenzierten Nachrichten

Um zu verhindern, dass Nachrichten durch die Speicherung in den Brokern bei Veränderungen eines Pfades erneut übertragen werden, werden die zuletzt übertragen Nachrichten in entsprechenden Abonnements referenziert. Dieser Parameter steuert die maximale Anzahl dieser Referenzen pro Abonnement.

Eine größere Zahl referenzierter Nachrichten führt zu weniger Doppeltübertragungen. Allerdings wächst mit jeder Referenz auch die Größe der Abonnements und somit der Kommunikationsaufwand. Eine sinnvolle Abstimmung mit der Anzahl gespeicherter Nachrichten ist hier wünschenswert, um den Kommunikationsaufwand durch Abonnements nicht unnötig zu erhöhen. Eine Einbindung der Anwendungen ist nicht nötig.

Schlussfolgerungen

Von den vier wesentlichen identifizierten Parametern zur Steuerung des Verhaltens der Algorithmen ist bei dreien eine Steuerung durch die Anwendung sinnvoll. Dies sind

- die Gültigkeitsdauer der Ankündigungen,

- die Anzahl zwischengespeicherter Nachrichten, sowie
- die Gültigkeitsdauer der Nachrichten im Cache.

Es wurden bei der Diskussion der Auswirkungen der vier betrachteten Parameter verschiedene theoretische Annahmen über das Verhalten der Algorithmen getroffen. Diese bedürfen einer Verifikation. Diese sind im Einzelnen:

- Eine längere Gültigkeitsdauer der Ankündigungen führt zu weniger Kommunikationsaufwand durch das Flooding der Ankündigungen.
- Eine Vergrößerung der Anzahl der zwischengespeicherten Nachrichten führt zum Verlust von weniger Nachrichten.
- Eine Erhöhung der Anzahl der in Abonnements referenzierten Nachrichten verringert die doppelte Übertragung von Nachrichten durch deren Zwischenspeicherung, erhöht aber den Kommunikationsaufwand durch größere Abonnements.

5.4.4. Zuordnung zu Anwendungsklassen

Neben der Beeinflussung einzelner Parameter, kann auch die Wahl des richtigen Algorithmus durch die Anwendungen beeinflusst werden. Eine Einteilung in die in Abschnitt 4.6.1 vorgestellten Anwendungsklassen und deren Einfluss auf die Performance der Algorithmen wird nachfolgend diskutiert.

Klasse 1 - eine einzelne kleine Nachricht

Zur Verbreitung einer einzelnen kleinen Nachricht ist es nicht sinnvoll, einen Pfad zu etablieren und nur für diese Nachricht zu nutzen. Bereits der Aufwand durch den Versand von Abonnements ist unnötig, wenn die Nachricht in eine Ankündigung verpackt und unter allen Geräten verteilt wird. Die Zwischenspeicherung und Weiterleitung an neue Geräte erlaubt eine zeitlich entkoppelte Zustellung der Nachricht an alle interessierten Senken.

Klasse 3 - eine einzelne große Nachricht

Mit steigender Nachrichtengröße wird auch der Versand einer einzelnen Nachricht an alle Geräte sehr schnell ineffizient. Daher ist die Etablierung effizienter Pfade und der Versand der Nachricht entlang dieser Pfade sinnvoll. Der Ein-Wege-Ansatz erzeugt mit wenigen Abonnements einen praktikablen Pfad pro Senke, während der Mehr-Wege-Ansatz in der Phase des Abonnements deutlich mehr Kommunikationsaufwand erzeugt. Sollte der Pfad während der Übertragung der einen Nachricht zusammenbrechen, erzeugt wiederum der Ein-Wege-Ansatz durch eine erneute Ankündigungsphase deutlich mehr Kommunikationsaufwand während der Mehr-Wege-Ansatz vermutlich für die kurze Zeit auf einen anderen nutzbaren Pfad ausweichen kann.

Beim Versand einer relativ kleinen Nachricht in einer nicht zu stark veränderlichen Topologie ist mit einem effizienteren Versand durch den Ein-Wege-Ansatz zu rechnen. Mit zunehmender Nachrichtengröße wird der Mehr-Wege-Ansatz durch die Wahl eines möglicherweise effizienteren Pfades günstiger. In stark veränderlichen Topologien sollte dieser ohnehin überlegen sein.

Klasse 2/4 - Strom kleiner und großer Nachrichten

Sollen über einen größeren Zeitraum viele Nachrichten versandt werden, ist die Wahl effizienter Pfade von enormer Bedeutung. In diesen Anwendungsklassen ist mit einer deutlichen Überlegenheit des Mehr-Wege-Ansatzes zu rechnen. Dies gilt insbesondere mit steigender Veränderlichkeit der Topologie.

Schlussfolgerungen

Ergebnis der Diskussion ist, das für den Versand einer einzelnen kleinen Nachricht der Versand in einer Ankündigung an alle Geräte bereits effizienter ist, als einen Pfad zu etablieren und dann nur einmal zu benutzen. In allen anderen Anwendungsklassen ist die Wahl des Mehr-Wege-Ansatzes höchstens minimal ineffizienter als der Ein-Wege-Ansatz, wahrscheinlich aber durch die Wahl effizienterer Pfade und weniger wiederholter Ankündigungsphasen deutlich effizienter.

Eine praktische Überprüfung ist hierbei insbesondere sinnvoll, um festzustellen

- wie groß der zusätzliche Kommunikationsaufwand durch mehr versandte Abonnements und eventuell Kündigungen tatsächlich ist und
- ob der Ein-Wege-Ansatz durch mehr Mobilität tatsächlich stärker an Effizienz verliert als der Mehr-Wege-Ansatz.

5.5. Zusammenfassung

In diesem Kapitel wurden zwei Routingalgorithmen vorgestellt, die geeignet sind, in der in Kapitel 4 vorgestellten Vermittlungstopologie Nachrichten von Quellen zu Senken zu vermitteln. Beide Algorithmen nutzen dazu drei Phasen, von denen die erste Phase – die Verbreitung von Ankündigungen per Flooding-Algorithmus in der gesamten Topologie – dazu dient, die Verfügbarkeit von Nachrichtentypen unter allen Geräten zu verbreiten.

Beim Ein-Wege-Ansatz wird in der zweiten Phase ein Pfad zu jeder Senke durch den Versand von Abonnements entlang dieses Pfades aktiviert und in der dritten Phase Nachrichten entlang dieses Pfades verbreitet. Beim Mehr-Wege-Ansatz werden in der zweiten Phase alle in der ersten Phase gefunden Pfade durch den Versand von Abonnements nach ihren Pfadkosten bis zur Senke sortiert und für den Versand von Nachrichten in der dritten Phase einer dieser Pfade ausgewählt.

Die Komplexität in Form von Speicherbedarf und Kommunikationsaufwand wurde für beide Algorithmen untersucht. Der Ein-Wege-Ansatz benötigt in jedem Fall weniger Speicherbedarf auf den Geräten als der Mehr-Wege-Ansatz. Der höhere Kommunikationsaufwand des Mehr-Wege-Ansatzes in der zweiten Phase wird dadurch kompensiert, dass für den Nachrichtenversand immer der optimale momentan erkannte Pfad genutzt wird, und dieser abhängig von der gewählten Metrik erheblich weniger Kommunikationsaufwand erzeugen kann. Außerdem erzeugt der Mehr-Wege-Ansatz durch seine Fähigkeit, zerstörte Pfade eine Zeit lang ohne Neuankündigung reparieren zu können, seltener den enorm hohen Kommunikationsaufwand der Ankündigungsphase.

Die Untersuchung der Skalierbarkeit beider Algorithmen ergab, dass ihr Speicherbedarf linear mit der Anzahl der Nachrichtentypen in der gesamten Umgebung steigt. Außerdem hängt der Kommunikationsaufwand wesentlich von der Anzahl der Nachrichtentypen in der Umgebung sowie der Differenz zwischen Kommunikationsverbindungen und Geräten, also der Form der Netzwerktopologie, ab. Es wurden verschiedene mögliche Erweiterungen skizziert, die Skalierbarkeit zu verbessern.

Das Verhalten der beiden Algorithmen wird wesentlich von vier Parametern gesteuert. Diese wurden auf ihre Auswirkungen hin untersucht und sinnvolle Belegungen beschrieben. Außerdem wurden die Algorithmen den in Abschnitt 4.6.1 vorgestellten Anwendungsklassen zugeordnet.

Bei den in diesem Abschnitt durchgeführten theoretischen Betrachtungen wurden sieben Punkte identifiziert, die einer praktischen Untersuchung bedürfen:

- Die Pfadnutzungsdauer in veränderlichen Topologien ist für den Mehr-Wege-Ansatz signifikant größer als für den Ein-Wege-Ansatz.
- Die Länge der gewählten Pfade ist bei Wahl einer Metrik, die mit der Pfadlänge positiv korreliert, für den Mehr-Wege-Ansatz erkennbar kleiner als für den Ein-Wege-Ansatz.
- Eine längere Gültigkeitsdauer der Ankündigungen führt zu weniger Kommunikationsaufwand durch das Flooding der Ankündigungen.
- Eine Vergrößerung der Anzahl zwischengespeicherter Nachrichten führt zum Verlust von weniger Nachrichten.
- Eine Erhöhung der Anzahl in Abonnements referenzierte Nachrichten verringert die doppelte Übertragung von Nachrichten durch deren Zwischenspeicherung, erhöht aber den Kommunikationsaufwand durch größere Abonnements.
- Wie groß ist der zusätzliche Kommunikationsaufwand des Mehr-Wege-Ansatzes durch zusätzlich versandte Abonnements und eventuell Kündigungen tatsächlich?
- Verliert der Ein-Wege-Ansatz durch mehr Mobilität tatsächlich stärker an Effizienz als der Mehr-Wege-Ansatz?

Die praktische Untersuchung dieser Annahmen per Simulation ausgewählter Szenarien ist Inhalt des nachfolgenden Kapitels.

Kapitel 6.

Simulative Evaluation

Inhalt

6.1 Ausgangssituation **124**
6.1.1 Zu überprüfende Aussagen 124
6.1.2 Bewertungskriterien 125
6.2 Methodik **129**
6.2.1 Simulationsumgebung 129
6.2.2 Modellierung 130
6.2.3 Szenarien 131
6.2.4 Validierung der Simulationsumgebung 133
6.3 Untersuchung des Einflusses der Parameter **135**
6.3.1 Versuch 1 – Gültigkeitsdauer der Ankündigungen 135
6.3.2 Versuch 2 – Anzahl zwischengespeicherter Nachrichten 135
6.3.3 Versuch 3 – Anzahl in Abonnements referenzierter Nachrichten 136
6.3.4 Schlussfolgerungen 136
6.4 Untersuchung der Algorithmen **137**
6.4.1 Versuch 4 – Auftreten von Pfadfehlern 137
6.4.2 Versuch 5 – Einfluss von Mobilität 138
6.4.3 Versuch 6 – Vergleich mit Flooding 138
6.4.4 Versuch 7 – Verhalten bei Anwendungen der Klasse 2 139
6.4.5 Versuch 8 – Verhalten bei Anwendungen der Klasse 3 139
6.4.6 Versuch 9 – Verhalten bei Anwendungen der Klasse 4 140
6.4.7 Versuch 10 – Horizontale Skalierbarkeit 140
6.4.8 Versuch 11 – Vertikale Skalierbarkeit 141
6.4.9 Schlussfolgerungen 142
6.5 Versuch 12 – Untersuchung in einem realitätsnahen Szenario **145**
6.5.1 Herausforderungen 145
6.5.2 Ergebnisse 147
6.6 Schwächen der Ansätze **151**
6.6.1 Ein-Wege-Ansatz 151
6.6.2 Mehr-Wege-Ansatz 152
6.7 Zusammenfassung **154**

Im vorhergehenden Kapitel wurden zwei Algorithmen zum Finden effizienter Pfade zum Versand von Nachrichten zwischen Quellen und Senken in einer Vermittlungstopologie beschrieben und theoretisch untersucht. Mehrere dabei gemachte Annahmen und Schlussfolgerungen bedürfen einer praktischen Bestätigung. In diesem Kapitel werden diese mit Hilfe der Simulation des Verhaltens der Algorithmen in zwei Szenarien untersucht.

Eingangs werden die zu untersuchenden Aussagen gruppiert und geeignete Bewertungskriterien für die Effizienz der Pfadsuche und der gefundenen Pfade vorgestellt. Anschließend werden die Simulationsumgebung, die Modellierung der Algorithmen und die Szenarien vorgestellt und verifiziert.

Darauf folgend werden die einzelnen Versuche charakterisiert und deren Ergebnisse analysiert. Dazu werden sie in drei Gruppen eingeteilt, Versuche zur Überprüfung des Einflusses von Parametern auf die Effizienz der Algorithmen, Versuche zur vergleichenden Bewertung der Effizienz der Algorithmen und ein Experiment zur Beurteilung der Funktion der Algorithmen in einem realitätsnahen Szenario.

Abschließend werden die Ergebnisse der Versuche in Bezug auf die ursprünglichen Annahmen sowie zusätzlich daraus gezogene Erkenntnisse zusammengefasst.

6.1. Ausgangssituation

In diesem Abschnitt werden aus den zu überprüfenden Annahmen[1] und den einleitend definierten Anforderungen für eine gute Problemlösung[2] Aussagen abgeleitet, die nachfolgend simulativ verifiziert werden können. Anschließend werden geeignete Kriterien für die Bewertung der Effizienz der Algorithmen und der gefundenen Pfade vorgestellt.

6.1.1. Zu überprüfende Aussagen

Da die Beurteilung der Eigenschaften der Algorithmen stark von der Konfiguration der Parameter abhängt, ist es zweckmäßig, zuerst die Versuche durchzuführen, die der Ermittlung geeigneter Konfigurationen dienen. Erst danach werden die Versuche durchgeführt, die der Einschätzung und dem Vergleich der Algorithmen untereinander dienen.

Bezüglich der Konfiguration der Parameter sind folgende drei Aussagen zu überprüfen:

Gültigkeitsdauer Eine längere Gültigkeitsdauer der Ankündigungen führt zu weniger Kommunikationsaufwand durch das Flooding der Ankündigungen.

Zwischenspeicher Eine Vergrößerung der Anzahl der zwischengespeicherten Nachrichten führt zum Verlust von weniger Nachrichten.

[1] siehe Abschnitt 5.5
[2] siehe Abschnitt 1.1

Referenzen Eine Erhöhung der Anzahl der in Abonnements referenzierter Nachrichten verringert die doppelte Übertragung von Nachrichten durch deren Zwischenspeicherung, erhöht aber den Kommunikationsaufwand durch größere Abonnements.

Jede dieser drei Aussagen wird durch einen Versuch abgebildet. Die Ergebnisse dienen als wichtige Grundlage für die Konfiguration der Algorithmen in den darauf folgenden Versuchen.

Es verbleiben vier weitere Aussagen:

Pfadbruch Die Pfadnutzungsdauer in veränderlichen Topologien ist für den Mehr-Wege-Ansatz signifikant länger als für den Ein-Wege-Ansatz.

Mobilität Der Ein-Wege-Ansatz verliert durch mehr Mobilität stärker an Effizienz als der Mehr-Wege-Ansatz.

Pfadlänge Die Länge der gewählten Pfade ist bei Wahl einer Metrik, die mit der Pfadlänge positiv korreliert, für den Mehr-Wege-Ansatz erkennbar kleiner als für den Ein-Wege-Ansatz.

Mehraufwand Wie stark steigt der Kommunikationsaufwand des Mehr-Wege-Ansatzes durch zusätzlich versandte Abonnements und eventuelle Kündigungen tatsächlich?

sowie die folgenden drei durch die Algorithmen beeinflussbaren Anforderungen an eine Problemlösung:

Zuverlässigkeit Die Zustellung von Nachrichten an die Konsumenten muss robust und zuverlässig erfolgen.

Effizienz Die Kommunikationstechnologie darf durch den Versand der Nachrichten und den zur Verteilung der Nachrichten nötigen Overhead der Algorithmen möglichst wenig belastet werden. Die Algorithmen müssen mit wenig Overhead möglichst effiziente Pfade finden.

Skalierbarkeit Sowohl die Anzahl der kommunizierenden Geräte, als auch die Anzahl der Nachrichtenzustände darf in angemessenen Grenzen, entsprechend der verwendeten Technologien, keinen Einfluss auf die Zuverlässigkeit der Middleware haben.

Während die Aussagen Pfadbruch und Mobilität in jeweils einem und die Skalierbarkeit in zwei eigenen Versuchen verifiziert werden, dienen die verbleibenden fünf Versuche der Überprüfung hinsichtlich der Aussage zur Pfadlänge sowie der Anforderungen Zuverlässigkeit und Effizienz. Die Aussage Mehraufwand zieht sich im Prinzip durch alle Versuche, lässt sich aber bei der Übertragung einer einzelnen Nachricht entsprechend Anwendungen der Klasse 3 am besten zeigen. Tabelle 6.1 gibt einen Überblick über alle Versuche und die dadurch zu überprüfenden Aussagen bzw. Anforderungen.

6.1.2. Bewertungskriterien

Bewertet werden die Zuverlässigkeit der Algorithmen, die Güte der gefundenen Pfade, die Effizienz der Algorithmen hinsichtlich des erzeugten Kommunikationsaufwandes, der er-

Nr.	Versuchsbezeichnung	darin geprüfte Aussagen
1	Gültigkeitsdauer der Ankündigungen	Gültigkeitsdauer
2	Anzahl zwischengespeicherter Nachrichten	Zwischenspeicher
3	Anzahl in Abonnements referenzierter Nachrichten	Referenzen
4	Auftreten von Pfadfehlern	Pfadbruch
5	Einfluss von Mobilität	Mobilität
6	Vergleich mit Flooding	Zuverlässigkeit, Effizienz
7	Verhalten bei Anwendungen der Klasse 2	Pfadlänge, Zuverlässigkeit, Effizienz
8	Verhalten bei Anwendungen der Klasse 3	Pfadlänge, Mehraufwand, Zuverlässigkeit, Effizienz
9	Verhalten bei Anwendungen der Klasse 4	Pfadlänge, Zuverlässigkeit, Effizienz
10	Horizontale Skalierbarkeit	Skalierbarkeit
11	Vertikale Skalierbarkeit	Skalierbarkeit
12	Untersuchung in einem realitätsnahen Szenario	Mehraufwand, Zuverlässigkeit, Effizienz

Tabelle 6.1.:
Übersicht über die Versuche und die dadurch überprüften Aussagen bzw. Anforderungen entsprechend der Übersicht in Abschnitt 6.1.1.

zeugte Overhead sowie spezielle Eigenschaften, wie die Pfadnutzungsdauer. All diese Eigenschaften lassen sich nicht direkt messen. Daher sind geeignete Indikatoren nötig, die möglichst genaue Rückschlüsse auf diese Eigenschaften zulassen.

Zuverlässigkeit

Die Zuverlässigkeit eines Algorithmus lässt sich am besten über den Anteil erfolgreich zugestellter Nachrichten, nachfolgend als **Vollständigkeit** bezeichnet, beurteilen. Diese kann sehr einfach absolut pro Senke bestimmt und quantitativ mit der Vollständigkeit anderer Algorithmen im gleichen Szenario verglichen werden. Um eine qualitative Aussage zu gewinnen, ist die Betrachtung in Relation zur Anzahl der von der betrachteten Senke im betrachteten Zeitraum theoretisch empfangbaren Nachrichten, basierend auf den im gleichen

Zeitraum überhaupt veröffentlichten Nachrichten, möglich. Allerdings bedarf der resultierende Wert unter Umständen einer nachträglichen Interpretation. Wurden vor dem betrachteten Zeitraum bereits Nachrichten veröffentlicht, so können diese aufgrund von Zwischenspeicherung die relative Vollständigkeit unbemerkt unter Umständen über den eigentlichen Maximalwert von 1,0 vergrößern.

Pfadgüte

Da die in Abschnitt 4.5 vorgestellte und in den Versuchen zum Einsatz kommende Metrik auf der geschätzten Übertragungsverzögerung zwischen benachbarten Geräten basiert, ist der beste Indikator für die Pfadgüte die Übertragungsverzögerung zwischen der Veröffentlichung der Nachricht durch die Quelle und deren Empfang durch die Senke. Um Verzögerungen durch eine eventuelle Zwischenspeicherung in auf dem Pfad liegenden Geräten zu vermeiden, werden Nachrichten bei der Zwischenspeicherung entsprechend markiert. Aus der Übertragungsverzögerung der von der Senke empfangenen nicht markierten Nachrichten wird dann die so genannte **Netto-Latenz** berechnet. Aus der Übertragungsverzögerung aller Nachrichten lässt sich auch eine Brutto-Latenz berechnen. Diese ist aber als Kriterium der Pfadgüte irrelevant.

Effizienz und Overhead

Die Effizienz der Algorithmen lässt sich quantitativ durch den insgesamt erzeugten Kommunikationsaufwand bewerten und mit anderen Algorithmen im gleichen Szenario vergleichen. Auch eine qualitative Bewertung anhand verschiedener erkennbarer und klar als Overhead charakterisierbarer Kommunikationsanteile ist möglich.

Die erzeugte **Netzlast** ergibt sich aus der Summe der von jedem Gerät empfangenen Datenmenge im betrachteten Zeitraum. Dabei wird ausschließlich die Datenmenge betrachtet, die zwischen der Netzwerkabstraktionsschicht und der Vermittlungsschicht[3] übertragen wurde. Zusätzlicher Overhead der Netzwerkabstraktionsschicht zum Finden neuer Nachbarn, zur Absicherung der Kommunikation zwischen benachbarten Geräten sowie zur Ermittlung und Überwachung der Kosteninformationen wird nicht berücksichtigt. Motivation hierfür ist zum Einen, dass die nicht betrachteten Datenmengen stark von der Implementierung der Netzwerkabstraktionsschicht abhängen und diese nicht Objekt der Untersuchung ist. Zum Anderen werden in der Netzwerkabstraktionsschicht ständig Daten übertragen. Es besteht somit nur eine schwache Relation zwischen dort übertragener Datenmenge und dem Ergebnis im betrachteten Zeitraum, sondern vielmehr zwischen jener Datenmenge und der Dauer des Versuchs. Um gleiche Rahmenbedingungen für verschiedene Versuche und untersuchte Algorithmen sicherzustellen, wird immer die gleiche Netzwerkabstraktionsschicht verwendet, die außerdem keine Optimierungen wie zum Beispiel die Nutzung von

[3] vergleiche Abschnitt 4.3.2

Nachrichtentyp	Algorithmus	Protokolloverhead
Nachricht	Ein-Wege-Ansatz	13 Byte
	Mehr-Wege-Ansatz	13 Byte
	Flooding	12 Byte
Nachrichtenfragment	Ein-Wege-Ansatz	16 Byte
	Mehr-Wege-Ansatz	18 Byte
	Flooding	16 Byte

Tabelle 6.2.:
Übersicht über den von den verwendeten Implementierungen der Algorithmen erzeugten Protokolloverhead zur Übertragung einer Nachricht zwischen Geräten.

Broadcast-Mechanismen in der zugrunde liegenden Kommunikationstechnologien verwendet.

Neben der Netzlast, die einen Vergleich der Effizienz von Algorithmen im gleichen Versuch zulässt, können verschiedene Arten von Overhead bewertet werden. Diese können als absolute Werte im gleichen Versuch, sowie in Relation zur Netzlast allgemeiner verglichen werden.

Phase-I-Overhead entsteht durch die Verteilung von Ankündigungen durch Flooding in der Vermittlungstopologie. Hierbei wird die Verfügbarkeit einer Quelle bestimmter Nachrichten im Netzwerk verbreitet. Auch wenn diese Information von Bedeutung ist, werden dabei keine Nutzdaten, also Nachrichten, übertragen.

Phase-II-Overhead entsteht durch den Versand von Abonnements zur Etablierung von Pfaden. Auch diese enthalten keine Nutzdaten.

Doppeltübertragungen werden bemerkt, wenn eine Nachricht oder ein Nachrichtenfragment, die im Zwischenspeicher eines Geräte bereits gespeichert sind, noch einmal empfangen werden. Es ist dabei zu beachten, dass die unbemerkte Doppeltübertragung möglich ist, wenn zum Beispiel die Zwischenspeicherung von Nachrichten im entsprechenden Versuch deaktiviert oder die Größe der Zwischenspeicher sehr klein ist. Eine entsprechende Interpretation dieses Wertes ist also notwendig.

Weiterhin entsteht Overhead durch die Signalisierung defekter Pfade und den Versand von Kündigungen für Abonnements. Dieser wird der Vollständigkeit halber als sonstiger Overhead gemessen, ist aber sehr gering und daher praktisch nicht von Bedeutung.

Eine weitere Form des Overheads ist der Protokolloverhead zur Übertragung von Nachrichten, der durch die Betrachtung der Nachrichtenformate bestimmbar ist. Er ist von untergeordneter Bedeutung, da er bei den verwendeten Implementierungen aller betrachteter Algorithmen sehr ähnlich ist. Tabelle 6.2 fasst die entsprechenden Werte zusammen.

spezielle Bewertungskriterien

Die Effizienz der beiden vorgestellten Algorithmen hängt wesentlich von der Anzahl der Ankündigungsphasen ab. Diese wiederum ist davon abhängig, wie lange gefundene Pfade aufrecht erhalten werden können, um Nachrichten zuzustellen. Gehen alle Pfade zu einer Senke verloren, so wird eine neue Ankündigungsphase ausgelöst. Die beschriebene Zeitspanne wird als **Pfadnutzungsdauer** bezeichnet. Um sie zu ermitteln, wird die Zeitspanne zwischen dem Zeitpunkt, zu dem der Pfad durch Empfang eines Abonnement durch die Quelle etabliert wurde, und dem Zeitpunkt, zu dem ein Pfad nicht mehr genutzt werden kann, gemessen. Für den zweiten Zeitpunkt gibt es drei Möglichkeiten: Eine Quelle bekommt einen zerstörten Pfad zu einer Senke signalisiert, die Gültigkeit einer Ankündigung ist abgelaufen oder der Versuch ist beendet.

Die minimale Pfadnutzungsdauer, 0, bedeutet, dass im gesamten Versuch kein Abonnement empfangen, und somit kein Pfad etabliert wurde. Die obere Grenze, die Dauer des Versuches bzw. die Gültigkeit der Abonnements im Versuch, bedeutet, dass ein Pfad etabliert wurde, und dieser nie zerstört wurde. Je größer der gemessene Zeitraum ist, desto besser hat der Algorithmus auf die Veränderlichkeit der Umgebung reagiert.

Um zu bewerten, wie schnell ein Algorithmus eine neue Senke in das System integriert, wird die **Anknüpfungslatenz** gemessen. Sie beschreibt die Zeitspanne zwischen dem Versand des Abonnements durch die neue Senke und deren Empfang der ersten auf das Abonnement passenden Nachricht. Im Gegensatz zur Netto-Latenz ist sie unabhängig vom Zeitpunkt der Veröffentlichung der Nachricht. Wünschenswert ist hier, dass die letzte im System veröffentlichte Nachricht aus dem Zwischenspeicher eines Gerätes so schnell wie möglich zur neuen Senke übermittelt wird und die Anknüpfungslatenz somit so kurz wie möglich ist. Erst wenn wieder neue Nachrichten veröffentlicht werden, ist die Netto-Latenz für die neue Senke von Bedeutung.

6.2. Methodik

In diesem Abschnitt werden die Simulationsumgebung, die Modellierung der untersuchten Systemarchitektur und Algorithmen sowie die dafür benutzten Szenarien charakterisiert. Außerdem wird ein Versuch zur Validierung wesentlicher Simulationsergebnisse durch den Vergleich mit theoretisch berechenbaren Werten beschrieben und ausgewertet.

6.2.1. Simulationsumgebung

Als Simulationsumgebung kommt OMNeT++ (VARGA und HORNIG, 2008), eine Simulationsumgebung für diskrete ereignisgesteuerte Simulation, in der Version 4.1 zum Einsatz. Die Modellierung erfolgt hier in einfachen Modulen, die zu zusammengesetzten Modulen zusammengefügt und ineinander verschachtelt werden können. Module kommunizieren frei

definierte Daten über Ein- und Ausgangs-Ports miteinander. Normalerweise erfolgt die Kommunikation über Kanäle zwischen den Ports. Daten können aber auch direkt an einen nicht verbundenen Eingangs-Port eines anderen Moduls versandt werden. Außerdem ist ein direkter Aufruf von entsprechend markierten Methoden der Module durch ein anderes Modul möglich. Verbunden mit einem objektorientierten Design ist dadurch ein Maximum an Flexibilität und Wiederverwendbarkeit gewährleistet. Durch die vollständige Programmierung der Simulationsumgebung und der Module in C++ ist OMNeT++ außerdem sehr leistungsfähig in Bezug auf die Größe der Modelle und die Ausführungsgeschwindigkeit der Simulationsläufe.

Zur Netzwerksimulation mit OMNeT++ kommt das INET-Framework (VARGA und HORNIG) in der Version 20100723 zum Einsatz. Es vereint Modelle für die wichtigsten Netzwerkprotokolle auf allen Schichten des OSI-Referenzmodells (STALLINGS, 2007, S. 42ff), drahtgebunden wie drahtlos, sowie Ausbreitungs- und Bewegungsmodelle.

6.2.2. Modellierung

Ein Gerät mit installiertem Broker wird in der Simulation durch 4 Schichten aus Modulen modelliert. Abbildung 6.1 gibt einen Überblick. Die unterste Schicht bilden die Netzwerkschnittstellen aus dem INET-Framework. Sie verbinden das Gerät mit den verschiedenen Netzwerktopologien. In den Versuchen kommen zwei Netzwerkschnittstellen zum Einsatz. Diese sind IEEE 802.3 (IEEE-SA STANDARDS BOARD, 2008) basierendes Ethernet mit LLC (IEEE-SA STANDARDS BOARD, 1998) sowie IEEE 802.11 (IEEE-SA STANDARDS BOARD, 2007) im Ad-Hoc-Modus.

Auf der Netzwerkschicht baut die Netzwerkabstraktion auf. Sie bildet die Funktion der Netzwerkabstraktionsschicht wie in Abschnitt 4.3.2 beschrieben nach. Es kommen auch hier zwei Module zum Einsatz, je eines pro zuvor erwähnter Netzwerkschnittstelle. Nach oben hin bietet jedes Modul der Netzwerkabstraktion einen Port zum Anschluss des Algorithmus.

Drei Algorithmen kommen in den Versuchen zum Einsatz und sind jeweils als ein Modul realisiert, Flooding als Referenz sowie der Ein- und der Mehr-Wege-Ansatz. Alle Geräte in einem Versuch nutzen den gleichen Algorithmus. Dieser kann per Versuchskonfiguration bequem ausgetauscht werden, so dass der gleiche Versuchsaufbau mit jedem Algorithmus simuliert werden kann. Das Modul jedes Algorithmus erlaubt den Anschluss beliebig vieler Netzwerkabstraktionsmodule sowie Anwendungen und bildet die Funktion der Vermittlungsschicht laut 4.3.2 sowie der Anwendungsabstraktionsschicht laut 4.3.2 ab.

Die oberste Schicht bilden die Module der Anwendungen. Sie modellieren jeweils eine Anwendung, Produzent, Konsument oder Verarbeiter. Außerdem befindet sich auf jedem Gerät eine Nachbartabelle. In dieser werden von den Modulen der Netzwerkabstraktion die gefundenen Nachbarn angemeldet und deren Pfadkosten ständig aktualisiert. Wichtige Veränderungen werden von ihr an den Algorithmus gemeldet und die aktuellen Pfadkosten können vom Algorithmus dort ständig abgerufen werden.

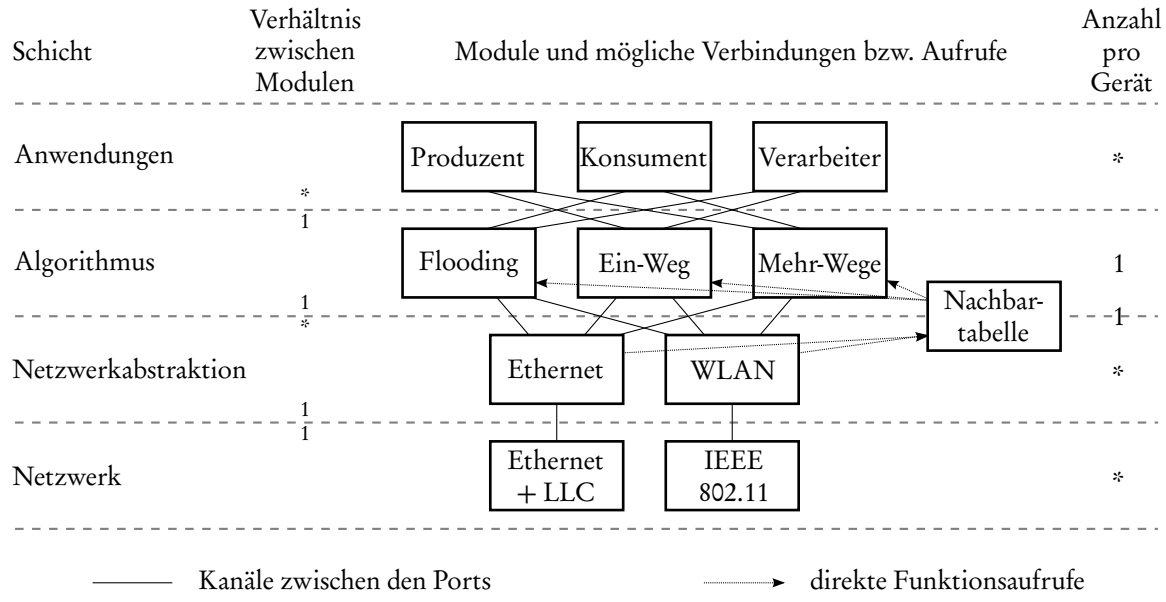


Abbildung 6.1.:
Übersicht über das Simulationsmodell. Jedes Gerät enthält eine Nachbartabelle und einen Algorithmus. Dazu kommen beliebig viele Anwendungen und pro zu benutzender Kommunikationsschnittstelle ein Netzwerkabstraktionsmodul. Die Nachbartabelle wird von den Netzwerkabstraktionsmodulen ständig aktualisiert und informiert den Algorithmus über Veränderungen.

6.2.3. Szenarien

Zur Untersuchung der Algorithmen per Simulation kommen zwei Szenarien zum Einsatz, die nachfolgend kurz charakterisiert werden. Eine detaillierte Beschreibung der darin abgebildeten Anforderungen an die Algorithmen findet sich nachfolgend im jeweiligen Abschnitt der Auswertung.

Testszenario

Der Grundgedanke hinter dem Testszenario ist die gezielte Untersuchung ausgewählter Eigenschaften des Algorithmus. Dazu ist eine Einstellung des Szenarios in den folgenden Kriterien notwendig:

- Größe der Topologie
- Grad der Veränderlichkeit der Topologie
- Größe und Anzahl der veröffentlichten Nachrichten
- Parameter des untersuchten Algorithmus

Das Testszenario besteht aus einer Menge mobiler Geräte, die per WLAN nach IEEE802.11 im Ad-Hoc-Modus kommunizieren können. Die Reichweite jedes Gerätes beträgt

ca. 300 [m] und sie sind auf einer quadratischen Fläche mit 400 [m] Kantenlänge angeordnet. Jedes Gerät verharrt eine zufällige Zeit, bewegt sich dann mit zufälliger Geschwindigkeit auf einen zufällig gewählten neuen Punkt auf der Fläche zu und verharrt dort dann wieder eine zufällige Zeit und so weiter.

Jeweils ein Produzent, ein Konsument, ein Verarbeiter und optional ein zweiter Verarbeiter bilden eine zusammenhängende Verarbeitungstopologie wobei nur der erste Verarbeiter die Nachrichten des Produzenten in Nachrichten für den Konsumenten umwandelt. Der zweite Verarbeiter erzeugt Nachrichten, für die kein Konsument verfügbar ist. Produzent $j[1 \leq j \leq p]$ ist auf Gerät j installiert, Verarbeiter j auf Gerät $j+1$, Konsument j auf Gerät $j+2$ und der optionale Verarbeiter j auf Gerät $j+3$. Somit sind für p Produzenten, p Verarbeiter, p Konsumenten und p optionale Verarbeiter genau $p+3$ Geräte nötig, um alle Anwendungen zu installieren.

Die Laufzeit des Versuches beträgt 1200 [s]. Die Veröffentlichung von Nachrichten durch den Produzenten beginnt nach [100,105) [s] und endet nach [1110,1120) [s]. Die tatsächlichen Zeiten werden für jeden Durchlauf zufällig gleichverteilt aus den Intervallen gewählt. Tabelle 6.3 fasst die Konfigurationsmöglichkeiten des Testszenarios zusammen.

Wenn nicht anders angegeben, werden von jedem Versuch pro Konfiguration und Algorithmus 20 Durchläufe mit verschiedenen Anfangswerten der Zufallsgeneratoren durchgeführt. Dabei wird sichergestellt, dass gleiche Durchläufe, die sich nur durch den Algorithmus oder dessen Konfiguration unterscheiden, auch identisch ablaufen. Werden zum Beispiel 20 Durchläufe einer Konfiguration des Szenarios mit beiden Algorithmen durchgeführt, so ist die Platzierung der Geräte, deren Bewegung sowie die Zeiten der Veröffentlichung von Nachrichten in Durchlauf 4 mit dem Ein-Wege-Ansatz identisch zu Durchlauf 4 mit dem Mehr-Wege-Ansatz.

Bahnhofszenario

Das zweite Szenario stellt das praktische Beispiel aus der Einleitung 1.3 nach. Es kommt eine Kombination aus drahtgebundenen und drahtlosen Technologien zum Einsatz. Die Fahrplaninformationen stellen mittelgroße Nachrichten dar, die durch die Verarbeitung noch einmal verkleinert werden. Die Bilddaten große Nachrichten, die ebenfalls durch Verarbeitung verkleinert werden. Die Frequenz der Veröffentlichung ist bei den Bilddaten sehr hoch, bei den Fahrplaninformationen gering.

Die Simulation läuft über eine Zeit von einer Stunde mit ca. 400 [s] Vorlaufzeit bevor die Produzenten anfangen, Nachrichten zu veröffentlichen. Bei den Fahrplaninformationen sind es [399,400) [s] und bei den Bilddaten [400,401) [s] jeweils gleichverteilt. In diesem Szenario lässt sich lediglich der benutzte Algorithmus austauschen und konfigurieren. Alle anderen Einstellungen sind in Tabelle B.13 im Anhang B.13 zusammengefasst und nicht veränderbar.

Parameter	Bez.	Einschränkungen	Std.
Größe der Topologie			
Anzahl der Geräte	n	$n > p + 1$	10
Anzahl der Produzenten	p	$p > 0$	1
Anzahl der Verarbeiter	v	$v = p$	p
Anzahl der Konsumenten	k	$k = p$	p
Anzahl zusätzlicher Verarbeiter	v'	$v' \leq p$	0
Grad der Veränderlichkeit der Topologie			
Geschwindigkeit der Geräte	$v_{\max}$		0 [m/s]
Länge der Pausen	t		∞
Größe und Anzahl der veröffentlichten Nachrichten			
Anzahl der Nachrichten pro Produzent	$a(p)$		
Frequenz der Veröffentlichung pro Produzent	$f(p)$		1 [Hz]
Größe der Nachrichten des Produzenten	$g(p)$		
Größe der Nachrichten des Verarbeiters	$g'(v)$		$g(p)$
Größe der Ankündigungen des Produzenten	$h(p)$		
Größe der Ankündigungen des Verarbeiters	$h'(v)$		$h(p)$

Tabelle 6.3.:
Übersicht über die Parameter des Testszenarios mit Bezeichner, eventuellen Einschränkungen und einem Standardwert, der genutzt wird, wenn kein anderer Wert angegeben ist. Der Standardwert für die Frequenz der Nachrichtenveröffentlichung ist der Durchschnitt. Die tatsächlich gewählten Zeitpunkte schwanken normalverteilt mit einer Standardabweichung von 5 [ms] um ungewollte Nebeneffekte zu vermeiden.

6.2.4. Validierung der Simulationsumgebung

In der Simulation wird auf drei Aspekte besonderer Wert gelegt. Diese sind

- die Veränderlichkeit der Topologie,
- die Messung sinnvoll interpretierbarer Übertragungszeiten, sowie
- die korrekte Ermittlung der erzeugten Netzlast.

Die Veränderlichkeit der Topologie lässt sich aufgrund ihrer Auswirkungen in den einzelnen Versuchen sehr gut erkennen. Pfade gehen verloren und einzelne Nachrichten werden nicht übermittelt. Die Realitätsnähe der Simulation der Veränderlichkeit ist hier weit weniger wichtig, als deren Vielseitigkeit zum Beispiel in der Häufigkeit und Dauer des Abrisses von Verbindungen.

Die Messung der Übertragungszeiten ist sehr schwer zu validieren. Ein Vergleich mit Messungen in einem realen Szenario kann nicht erfolgreich sein, da in der Simulation bewusst enorme Vereinfachungen der Realität durchgeführt werden. Es wird nur eine Positionierung

Parameter	Wert
Anzahl der Nachrichten pro Produzent	100
Größe der Nachrichten des Produzenten	100
Größe der Ankündigungen des Produzenten	20

Tabelle 6.4.:
Übersicht über die Parameter des Versuches zur Validierung der Simulationsumgebung.

und Bewegung in zwei Dimensionen simuliert. Das Ausbreitungsmodell basiert lediglich auf dem Pfadverlust. Aussagen basierend auf den absoluten Messwerten sind hier demzufolge nicht sinnvoll. Lediglich ein Vergleich zwischen Messwerten für verschiedene Algorithmen im gleichen Versuch ist möglich.

Die übertragenen Datenmengen müssen korrekt simuliert werden, um verlässliche Aussagen über die erzeugte Netzlast zu erhalten. Hier ist eine Validierung anhand eines Vergleiches zu den theoretisch errechneten Werte möglich. Dazu wird das Testszenario leicht abgeändert. Die Kantenlänge der Fläche wird auf 200 [m] reduziert. Die Geräte werden zufällig platziert und bewegen sich nicht. Es entsteht also eine zufällige Anordnung der Geräte. Das Netzwerk, das sie bilden, bildet immer einen vollständig Graphen, da die Länge der Diagonale mit 282 [m] kleiner ist als die Reichweite der einzelnen Geräte. Dieser Graph hat durch die 10 Geräte somit 10 Ecken. Da er vollständig verbunden ist, hat er demzufolge 45 Kanten und es werden 81 Nachrichtenkopien für das Verbreiten einer Nachricht per Flooding benötigt. Somit lassen sich die nachfolgenden theoretische Aussagen über die Netzlast ableiten und in einem Versuch validieren. Die dafür notwendigen Parameter sind in Tabelle 6.4 zusammengefasst.

Phase 1 81 Kopien der Ausgangsankündigung sowie der verarbeiteten Ankündigung mit einer Größe von je 54 Byte Header und 20 Byte Nutzlast werden versandt:

$$81 \cdot 2 \cdot (54\,[B] + 20\,[B]) = 11988\,[B].$$

Phase 2, Ein-Wege-Ansatz Die Größe eines Abonnements beträgt 14 Byte, wenigstens 2 Abonnements sind nötig:

$$x \cdot 14\,[B] | x \in \mathbb{N}, x \geq 2.$$

Phase 3, Ein-Wege-Ansatz 100 Nachrichten mit einer Größe von 13 Byte Header, 4 Byte Anwendungsheader und 100 Byte Nutzlast werden über x Hops versandt:

$$100 \cdot (100\,[B] + 13\,[B] + 4\,[B]) \cdot x = x \cdot 11700\,[B].$$

Phase 2, Mehr-Wege-Ansatz Die Größe eines Abonnements beträgt hier 23 Byte. Es sind deutlich mehr Abonnements nötig als beim Ein-Wege-Ansatz:

$$y \cdot 23\,[B] | y \in \mathbb{N}.$$

Phase 3, Mehr-Wege-Ansatz Die Nachrichtengröße ist identisch zum Ein-Wege-Ansatz. Die Pfade sind immer so kurz wie möglich, also 2 Hops lang:

$$23400[B].$$

Phase 1/2, Flooding Beim Flooding gibt es keine 1. und 2. Phase:

$$0.$$

Phase 3, Flooding Die Nachrichtengröße ist identisch zu den vorhergehenden Ansätzen, aber es werden 2 mal 81 Kopien versandt:

$$81 \cdot 2 \cdot 11700\,[B] = 1895400\,[B].$$

Bei 100 Durchläufen mit jeweils zufälligen Positionen der Knoten konnten alle Aussagen bestätigt werden. Die detailliert Auswertung des Versuches befindet sich im Anhang B.1.

6.3. Untersuchung des Einflusses der Parameter

In den ersten drei Versuchen werden die Auswirkungen der Veränderung bestimmter Parameter der beiden Algorithmen untersucht. Diese sind im Einzelnen die Gültigkeitsdauer der Ankündigungen, die Anzahl zwischengespeicherter Nachrichten sowie die Anzahl der in Abonnements referenzierten Nachrichten. Die entsprechenden Versuche werden im Anschluss kurz skizziert und deren Ergebnisse zusammengefasst. Um die Diagramme erkennbar zu gestalten und den Text nicht durch zu viele Tabellen zu zerstückeln befinden sich alle Diagramme und Konfigurationstabellen im Anhang B. Anschließend werden aus den Ergebnissen für die folgenden Versuche sinnvolle Belegungen der untersuchten Parameter geschlussfolgert.

6.3.1. Versuch 1 – Gültigkeitsdauer der Ankündigungen

Tabelle B.2 stellt die Konfiguration des Szenarios für diesen Versuch dar. Der Versuch wird mit verschiedenen Werten für die Gültigkeitsdauer der Ankündigungen für beide Algorithmen jeweils 20 mal simuliert. Dabei wird die Zwischenspeicherung von Nachrichten und somit jegliche zeitliche Entkopplung deaktiviert, um aus der Vollständigkeit auf die exakte Anzahl der nicht direkt zugestellten Nachrichten schließen zu können.

Die Ergebnisse bestätigen die Aussage, dass der Overhead durch die Ankündigungen mit steigender Gültigkeitsdauer derselben sinkt, deutlich. Abbildung B.6 illustriert dies. Außerdem ist festzustellen, dass mit steigender Gültigkeitsdauer auch der Overhead durch Abonnements sinkt, was ebenfalls zu erwarten war. Die Ursache für das starke Abfallen der Vollständigkeit des Ein-Wege-Ansatzes mit steigender Gültigkeitsdauer der Ankündigungen wird im Abschnitt 6.6 mit den Schwächen der einzelnen Ansätze diskutiert.

6.3.2. Versuch 2 – Anzahl zwischengespeicherter Nachrichten

Bei gleicher Konfiguration des Szenarios wird die Größe des Zwischenspeichers verändert. Die Gültigkeitsdauer der Ankündigungen wird auf 50 [s] gesetzt, da dies zum häufigen Neuaufbau der Pfade führt und somit das Aufkommen von Doppeltübertragungen als ungewollter Nebeneffekt besonders gut zu beobachten ist.

In den Ergebnissen in Abbildung B.11 erkennt man deutlich, dass die Vollständigkeit die ursprüngliche Aussage bestätigend mit steigender Größe des Zwischenspeichers ansteigt. Ursache hierfür ist, dass durch die Zwischenspeicherung und dadurch wiederholte Übertragung von Nachrichten vormals verlorene Nachrichten doch noch zugestellt werden. Dieser Effekt kehrt sich ab einer bestimmten Größe des Zwischenspeichers um. Hierfür ist die beinahe linear zur Größe des Zwischenspeichers steigende Anzahl an mehrfach übertragenen Nachrichten und der damit verbundene Anstieg der Netzlast verantwortlich. Durch die Überlastung und den kurzzeitigen Ausfall einzelner Verbindungen wird die Veränderlichkeit der Topologie künstlich erhöht, was zu einem Absinken der Vollständigkeit führt.

6.3.3. Versuch 3 – Anzahl in Abonnements referenzierter Nachrichten

Durch eine Erhöhung der Anzahl in Abonnements referenzierter Nachrichten sinkt laut der nächsten zu prüfenden Aussage die Anzahl doppelt übertragener Nachrichten wieder. Dazu wird die Konfiguration des Szenarios unverändert gelassen. Die Gültigkeitsdauer für Ankündigungen wird wiederum auf 50 [s] gesetzt und die Größe des Zwischenspeichers auf 20 Nachrichten. Die Anzahl der in Abonnements referenzierten Nachrichten wird variiert.

Die Aussage wird durch die Ergebnisse des Versuchs bestätigt. Es stellt sich sogar ein klares Optimum ein. Die Anzahl doppelt übertragener Nachrichten sinkt nahezu linear mit der Anzahl der in Abonnements referenzierten Nachrichten und erreicht einen Wert nahe 0 wenn die Anzahl der referenzierten Nachrichten genau der Größe des Zwischenspeichers entspricht. Eine weitere Erhöhung der Anzahl referenzierter Nachrichten bringt in Bezug auf die Anzahl doppelt übertragener Nachrichten keine weitere Verbesserung. Im Gegenteil, sie erhöht den Overhead durch größere Abonnements. Allerdings hat dieser aufgrund der geringen Größe von Abonnements im Vergleich mit Nachrichten und der geringen Anzahl von Abonnements im Vergleich mit Ankündigungen kaum einen messbaren Effekt auf die insgesamt erzeugte Netzlast.

6.3.4. Schlussfolgerungen

Für die Parameter der beiden Algorithmen ergeben sich entsprechend der bestätigten Aussagen und sonstigen Ergebnisse der drei zuvor geschilderten Versuche die folgenden Schlussfolgerungen.

Gültigkeitsdauer der Ankündigungen

Die Gültigkeitsdauer der Ankündigungen in einer veränderlichen Umgebung sollte für den Ein-Wege-Ansatz eher niedrig gewählt werden. Im betrachteten Szenario drohte schon bei einer Gültigkeitsdauer von 100 [s] ein nicht kompensierbarer Nachrichtenverlust. Für die folgenden Versuche wird, wenn es nicht aus anderen Gründen zwingend notwendig ist, eine Gültigkeitsdauer von 50 [s] genutzt.

Für den Mehr-Wege-Ansatz hat die Gültigkeitsdauer keinen Einfluss auf die Vollständigkeit. Eine hohe Gültigkeit vermindert hier den Overhead in der Ankündigungsphase enorm. Daher ist die Wahl einer langen Gültigkeitsdauer hier zu empfehlen. Die Begrenzung sollte durch die Anforderungen der Anwendungen erfolgen. In den folgenden Versuchen wird, wenn es nicht aus anderen Gründen zwingend notwendig eine Gültigkeitsdauer von 1000 [s] genutzt.

Größe des Zwischenspeichers

Durch die Zwischenspeicherung von Nachrichten können Nachrichtenverluste bei der Übertragung in einer veränderlichen Topologie abgemildert werden. Je mehr Nachrichten gespeichert werden, desto größer ist dieser Effekt. Eine Begrenzung ist hier durch die auf den einzelnen Geräten zur Verfügung stehende Speichermenge und die Anforderungen der Anwendungen sinnvoll.

In den folgenden Versuchen werden bei der Übertragung mehrerer Nachrichten 20 Nachrichten zwischengespeichert. Da die Veröffentlichungsfrequenz von Nachrichten in den entsprechenden Versuchen bei ca. 1 [Hz] liegt, überbrücken diese Nachrichten einen Zeitraum von etwa 20 [s]. Eine Gültigkeit der Nachrichten im Zwischenspeicher von 30 [s] sorgt dafür, dass die Nachrichten in diesem Zeitraum sicher gültig sind und im Fall einer Änderung des Pfades trotzdem nach akzeptabler Zeit aus dem Zwischenspeicher entfernt werden.

Anzahl der referenzierten Nachrichten

Bei der Zwischenspeicherung von 20 Nachrichten in den entsprechenden Versuchen werden im Folgenden auch 20 Nachrichten referenziert. Weniger referenzierte Nachrichten führen zu einem starken Anstieg der Anzahl doppelt übertragener Nachrichten, während mehr referenzierte Nachrichten keinen weiteren Vorteil bringen.

6.4. Untersuchung der Algorithmen

Durch die acht folgenden Versuche werden die eingangs abgeleiteten Aussagen über die Algorithmen – Pfadbruch, Mobilität, Pfadlänge und Mehraufwand – sowie die Erfüllung der Anforderungen – Zuverlässigkeit, Effizienz und Skalierbarkeit – überprüft. Wiederum werden die Motivation für die Versuche und die Versuche selbst kurz skizziert und die wesentlichen Ergebnisse zusammengefasst. Die referenzierten Diagramme und Konfigurationstabellen finden sich wiederum im Anhang B.

6.4.1. Versuch 4 – Auftreten von Pfadfehlern

Ziel dieses Versuches ist es, die Algorithmen hinsichtlich der Häufigkeit und der durchschnittlichen Zeit bis zum Auftreten von Pfadfehlern zu vergleichen. Dazu bleibt die

Konfiguration des Szenarios unverändert. Die Gültigkeit der Ankündigungen beträgt mit 10000 [s] mehr als die Laufzeit des Versuches, somit werden die Ankündigungen nur durch auftretende und erkannte Pfadfehler verlängert. Der Zwischenspeicher für Ankündigungen wird deaktiviert um die Netzlast der Algorithmen nicht unnötig zu erhöhen. Dies stört nicht, da die erreichte Vollständigkeit für diesen Versuch nicht relevant ist.

Um das Konfidenzintervall für die ermittelten Werte zu verkleinern, werden 100 Durchläufe des Versuches simuliert. Wie in Abbildung B.17 zu erkennen ist die durchschnittliche Pfadnutzungsdauer beim Mehr-Wege-Ansatz in etwa doppelt so hoch wie beim Ein-Wege-Ansatz, was die eingangs formulierte Aussage bestätigt.

6.4.2. Versuch 5 – Einfluss von Mobilität

Ziel dieses Versuches ist der Vergleich der Effizienz beider Algorithmen abhängig vom Grad der Veränderlichkeit der Topologie. Tabelle B.6 fasst die Konfiguration des Szenarios für Versuch 5 zusammen. Die Geschwindigkeit der Geräte wird pro Bewegungsvorgang normalverteilt um den gegebenen Durchschnittswert mit einer Standardabweichung von 100 [ms] gewählt. Abbildung B.19 fasst die Ergebnisse für die erzeugte Netzlast und die erreichte Vollständigkeit beider Algorithmen zusammen.

Für den Mehr-Wege-Ansatz steigt die Netzlast mit steigender Mobilität der Geräte leicht an während sie für den Ein-Wege-Ansatz konstant bleibt. Im Gegensatz dazu erreicht der Mehr-Wege-Ansatz unabhängig vom Grad der Mobilität eine Vollständigkeit von nahezu 100% während die Vollständigkeit des Ein-Wege-Ansatzes kontinuierlich bis unter 80% abfällt. Die relative Netzlast pro übertragener Nachricht steigt somit beim Ein-Wege-Ansatz genau wie beim Mehr-Wege-Ansatz um ca. 25% bei einer Erhöhung der Geschwindigkeit der Geräte von durchschnittlich 1 [m/s] auf durchschnittlich 10 [m/s].

Die Aussage, dass der Ein-Wege-Ansatz bei steigender Mobilität stärker an Effizienz verliert als der Mehr-Wege-Ansatz, kann somit nicht bestätigt werden. Beide verlieren ähnlich stark an Effizienz, während der Mehr-Wege-Ansatz aber deutlich robuster auf Veränderungen der Mobilität wie auch auf Mobilität überhaupt reagiert.

6.4.3. Versuch 6 – Vergleich mit Flooding

Ziel dieses Versuches ist die Beurteilung der Zuverlässigkeit und Effizienz der beiden Algorithmen. Dazu wird das Verbreiten relativ kleiner Nachrichten per Flooding-Algorithmus als Referenz herangezogen. Die entsprechende Konfiguration des Szenarios ist in Tabelle B.7 dargestellt.

Abbildung B.23 skizziert die wesentlichen Ergebnisse des Versuchs. Flooding ist klar der robusteste Ansatz. Es erreicht trotz der veränderlichen Topologie ständig eine Vollständigkeit von 100%. Der Mehr-Wege-Ansatz erreicht nahezu das gleiche Ergebnis, während der Ein-Wege-Ansatz hier klar unterlegen ist. In Bezug auf die erzeugte Netzlast sind beide An-

sätze dem Flooding weit überlegen, wobei auch hier der Mehr-Wege-Ansatz besser abschneidet.

Der Mehr-Wege-Ansatz ist dem Flooding bezogen auf die Robustheit nur geringfügig unterlegen, während er bei der Effizienz um einen Faktor von 70 überlegen ist. Mit steigender Nachrichtengröße ist demzufolge beim Flooding weit eher mit einer Überlastung des Netzwerkes zu rechnen.

6.4.4. Versuch 7 – Verhalten bei Anwendungen der Klasse 2

In diesem Versuch werden Zuverlässigkeit und Effizienz des Ansatzes und die Qualität der gewählten Pfade für beide Ansätze in Bezug auf Anwendungen der Klasse 2 untersucht. Wie in Tabelle B.8 dargestellt, wird ein Strom kleiner Nachrichten veröffentlicht. Die Geräte bewegen sich nicht, um den Einfluss von Topologieveränderungen auszuschließen. Da demzufolge nicht mit Pfadbrüchen zu rechnen ist, benutzen beide Algorithmen eine Gültigkeitsdauer der Ankündigungen von 1000 [s]. Es ist also in beiden Fällen nur eine Ankündigungsphase notwendig.

Abbildung B.26 fasst die Ergebnisse des Versuches zusammen. Beide Algorithmen sind in der Lage, alle Nachrichten erfolgreich zuzustellen. Der Mehr-Wege-Ansatz erzeugt etwas mehr Netzlast durch Ankündigungen und Abonnements. Dies wird aber durch die Wahl kürzerer Pfade in nahezu jedem Durchlauf kompensiert. Diese wird anhand der nahezu diskreten Aufteilung der Netzlast in Blöcke von ca. 1500 [kiB] deutlich. Meist kommt der Mehr-Wege-Ansatz mit zwei Hops, Quelle – Verarbeiter – Senke aus. In vier Durchläufen werden drei Hops und in einem Durchlauf vier Hops benötigt. Der Ein-Wege-Ansatz benötigt in nur sechs Durchläufen die identische Anzahl an Hops. Bei allen anderen Durchläufen sind es wenigstens ein Hop mehr. Da die verwendete Metrik auf der Übertragungszeit basiert, ergibt sich auch bei der Netto-Latenz ein ähnliches Bild.

Wenn mehrere Pfade verfügbar sind, lohnt sich also bei Anwendungen der Klasse 2 der zusätzliche Aufwand des Mehr-Wege-Ansatzes, da er die Nachrichten jeweils auf den besten verfügbaren Pfaden vermittelt.

6.4.5. Versuch 8 – Verhalten bei Anwendungen der Klasse 3

Analog zum vorhergehenden Versuch werden Zuverlässigkeit und Effizienz des Ansatzes und die Qualität der gewählten Pfade beider Ansätze in Bezug auf Anwendungen der Klasse 3 untersucht. Tabelle B.9 fasst die Konfiguration des Szenarios zusammen. Vom Produzenten wird eine einzelne Nachricht von 51200 Byte veröffentlicht und vom Verarbeiter in eine Nachricht von 35840 Byte umgewandelt. Die Senke kündigt das Abonnement sofort nach Erhalt der Nachricht, um zusätzlichen Aufwand zur Aufrechterhaltung des etablierten Pfades darüber hinaus zu vermeiden. Die Gültigkeit der Ankündigungen wurde für beide Algorithmen auf 50 [s] eingestellt. Da der Konsument in diesem Versuch nur ca. 20 [s] aktiv

ist und sich dann von seinem Broker abmeldet, ist trotz dessen nur eine Ankündigungsphase notwendig.

Die Versuchsergebnisse, zusammengefasst in Abbildung B.29, suggerieren eine starke Ähnlichkeit zum Verhalten bei Anwendungen der Klasse 2. Mit beiden Ansätzen wird die Nachricht jeweils vollständig übertragen. Der Mehr-Wege-Ansatz erzeugt für die Etablierung des Pfades bis zu ca. 3 [kiB] mehr Netzlast durch zusätzliche Abonnements und Kündigungen, kompensiert dies aber in 16 von 20 Durchläufen durch eine deutliche Reduktion der Netzlast in Folge der Wahl eines kürzeren Pfades.

Die Netto-Latenz lässt sich bei der Übertragung nur einer Nachricht nicht bestimmen, da die Nachricht für die Zeit, bis wenigstens eine Senke gefunden ist, in der Quelle zwischengespeichert werden muss. Stattdessen bietet sich der Vergleich der Anknüpfungslatenz, dargestellt in Abbildung B.30 an. Diese setzt sich aus der Abonnement-Phase und der Übertragung der Nachricht zusammen. In Folge dessen ist in den Fällen, in denen der Mehr-Wege-Ansatz einen kürzere Pfad nutzt, die Anknüpfungslatenz entsprechend geringer. In den anderen Fällen ist sie leicht erhöht, da die Übertragung einer größeren Anzahl von Abonnements die Übertragung der Nachrichtenfragmente behindert.

Der Mehraufwand durch zusätzliche Abonnements und Kündigungen ist im Verhältnis zur Netzlast durch Ankündigungen und Nachrichtenübertragungen vernachlässigbar gering. Der Mehr-Wege-Ansatz ist dem Ein-Wege-Ansatz in Effizienz und Pfadlänge auch in diesem Versuch überlegen. Die Zuverlässigkeit unterscheidet sich nicht, da die Übertragung einer einzelnen Nachricht so schnell geht, dass auch in einer veränderlichen Umgebung die Wahrscheinlichkeit für eine Pfadunterbrechung sehr gering ist. Dies kann sich mit steigender Nachrichtengröße verändern.

6.4.6. Versuch 9 – Verhalten bei Anwendungen der Klasse 4

Um Zuverlässigkeit, Effizienz und den Einfluss der Pfadgüte beider Algorithmen für Anwendungen der Klasse 4 zu vergleichen wird wiederum eine statische Konfiguration des Szenarios entsprechend Tabelle B.10 mit einer Ankündigungsgültigkeit von 1000 [s] gewählt. Dadurch ist eine Vergleichbarkeit der Ergebnisse beider Ansätze gewährleistet.

Mit wenigen Ausnahmen übertragen beide Ansätze 100% der Nachrichten erfolgreich. Die Netzlast wird, wie in Abbildung B.32 zu sehen, aufgrund der Natur der Anwendung von der Nutzlast der Nachrichten dominiert. Der Overhead ist sehr gering. Dadurch sind die Auswirkungen der Wahl optimaler Pfade sehr deutlich. Sowohl in Bezug auf die Netzlast, als auch auf die Netto-Latenz (Abbildung B.33) ist der Mehr-Wege-Ansatz hier klar im Vorteil.

6.4.7. Versuch 10 – Horizontale Skalierbarkeit

Ziel dieses Versuches ist der Vergleich beider Algorithmen in Bezug auf die Abhängigkeit der Zuverlässigkeit von der Größe der Netzwerktopologie. Dabei ist entsprechend der theo-

retischen Untersuchung in Abschnitt 5.4.2 die Anzahl der Kommunikationsverbindungen entscheidend. Dementsprechend ist die Konfiguration des Versuches, dargestellt in Tabelle B.11, so gestaltet, dass die Anzahl der Geräte auf der gleichen Grundfläche erhöht wird. Dadurch ist sichergestellt, dass mit jedem zusätzlichen Gerät eine große Anzahl zusätzlicher Nachbarschaftsbeziehungen und somit Kommunikationsverbindungen hinzukommen. Der Versuch wird je Konfiguration 100 Mal durchgeführt, da insbesondere in den Übergangsbereichen die Ergebnisse stark schwanken und mit nur 20 Versuchen keine aussagekräftigen Mittelwerte errechnet werden können.

Abbildung B.34 zeit das Verhalten der beiden Algorithmen in Bezug auf die Vollständigkeit in Abhängigkeit von der Anzahl der Geräte in der Netzwerktopologie. Es ist deutlich zu sehen, dass mit steigender Gerätezahl und somit steigender Anzahl an vorhandenen Kommunikationsverbindungen die Vollständigkeit der Algorithmen abnimmt. Dies gestaltet sich derart, dass die Algorithmen bis zu einer bestimmten Gerätezahl in der Lage sind, alle Nachrichten zu übermitteln. Für den Ein-Wege-Ansatz liegt diese bei 13 Geräten, für den Mehr-Wege-Ansatz bei 16 Geräten.

Die Vollständigkeit des Ein-Wege-Ansatzes fällt dann relativ stark ab und bereits bei 25 Geräten wird kaum noch eine Nachricht übertragen. Für den Mehr-Wege-Ansatz fällt die Vollständigkeit bis zu 25 Geräten nur relativ leicht ab. Es werden noch deutlich über 80 % der Nachrichten übertragen. Erst danach beginnt ein starker Abfall der Vollständigkeit bis auf 0 bei 34 Geräten. Der Ein-Wege-Ansatz ließe sich in diesem Szenario also bis zu einer Grenze von etwa 16 Geräten sinnvoll betreiben während der Mehr-Wege-Ansatz bis zu einer Grenze von etwa 25 Geräten noch nutzbar ist. Der Mehr-Wege-Ansatz skaliert deutlich besser als der Ein-Wege-Ansatz in Bezug auf die Anzahl der Geräte auf engem Raum.

6.4.8. Versuch 11 – Vertikale Skalierbarkeit

Ziel dieses Versuches ist die Untersuchung des Einflusses der Größe der Verarbeitungstopologie auf die erreichte Vollständigkeit beider Ansätze. Tabelle B.12 fasst die Konfiguration des Szenarios zusammen. Die Anzahl der Produzenten und damit auch die der Verarbeiter und Konsumenten wird schrittweise erhöht. Außerdem wird pro Produzent ein zusätzlicher Verarbeiter aktiviert, der die Nachrichten des Produzenten in Nachrichten umzuwandeln in der Lage ist, für die keine geeigneten Konsumenten existieren.

Für jeden Produzenten bedeutet dies, dass drei Nachrichtentypen existieren, der vom Produzenten veröffentlichte, der vom Verarbeiter erzeugte sowie der vom zusätzlichen Verarbeiter erzeugte. In der Ankündigungsphase sind alle drei Nachrichtentypen relevant, da deren Verfügbarkeit in der Vermittlungstopologie propagiert werden muss. In den Phasen 2 und 3 ist der dritte Nachrichtentyp nicht von Bedeutung, da für ihn kein Konsument existiert. Der Versuch wird je Konfiguration 100 Mal durchgeführt.

Der Abfall der Vollständigkeit ist, wie in Abbildung B.39 zu sehen, weit weniger stark als bei der Abhängigkeit von der Netzwerktopologie. Dies war zu erwarten, da durch eine

Vergrößerung der Netzwerktopologie die Anzahl der Nachrichten pro Ankündigungsphase punktuell stark ansteigt und schließlich zur Überlastung einzelner Kommunikationsverbindungen führt. Eine Erhöhung der Anzahl der Verarbeitungstopologien durch eine Erhöhung der Anzahl der Produzenten führt zu einer Erhöhung der Anzahl der Ankündigungsphasen, wobei die Ankündigungsphasen zweier Produzenten nicht exakt gleichzeitig stattfinden. Somit steigt die Belastung der Netzwerktopologie zeitlich weniger punktuell und es kommt nicht so schnell zu einer Überlastung der Kommunikationsverbindungen.

Es wird allerdings auch hier deutlich, dass die Kurve für den Ein-Wege-Ansatz stärker abfällt als für den Mehr-Wege-Ansatz. Auch in Bezug auf die Abhängigkeit der Vollständigkeit von der Anzahl der Nachrichtentypen skaliert der Mehr-Wege-Ansatz besser.

6.4.9. Schlussfolgerungen

Ziel der vorangehenden 8 Versuche war die Bestätigung oder Widerlegung bzw. Beantwortung der in Abschnitt 6.1 aufgeworfenen Thesen und Fragestellungen in einem entsprechend konfigurierbaren Testszenario. Die Ergebnisse werden nachfolgend entsprechend den Thesen sortiert und diskutiert.

Pfadbruch

In Versuch 4 konnte gezeigt werden, dass ein Pfad im Mehr-Wege-Ansatz deutlich länger erfolgreich benutzt werden kann als im Ein-Wege-Ansatz. Im Versuch war die Pfadnutzungsdauer für den Mehr-Wege-Ansatz in etwa doppelt so lang.

Für die theoretische Betrachtung der Effizienz und Skalierbarkeit bedeutet dies einen erheblichen Vorteil für den Mehr-Wege-Ansatz, da bei der Wahl geeigneter Parameter in veränderlichen Umgebungen seltener Ankündigungsphasen ausgelöst werden müssen und diese einen erheblichen negativen Einfluss auf Effizienz und Skalierbarkeit haben.

Mobilität

Bei steigender Mobilität wurde für den Ein-Wege-Ansatz ein stärkerer Abfall an Effizienz vorausgesagt, als für den Mehr-Wege-Ansatz. Dies konnte im Versuch nicht bestätigt werden. Beide Ansätze verlieren mit ca. 25% relativem Anstieg der Netzlast pro erfolgreich übertragener Nachricht ähnlich stark an Effizienz. Während der Ein-Wege-Ansatz bei gleich bleibender Belastung des Netzwerkes kontinuierlich weniger Nachrichten erfolgreich zustellen kann, steigt beim Mehr-Wege-Ansatz die Netzlast bei konstant bleibender Vollständigkeit. Die absolute Effizienz des Mehr-Wege-Ansatzes als Netzlast pro erfolgreich übertragener Nachricht ist über den gesamten Versuch deutlich besser als für den Ein-Wege-Ansatz.

Die Ursachen für diese Beobachtung sind sehr komplex. Aus der Betrachtung des Phase-I-Overheads wird deutlich, dass der Anstieg für beide Algorithmen absolut betrachtet identisch bei etwa 300 [kiB] über die gesamte Spanne liegt. Der Phase-II-Overhead hat mit we-

niger als 10 [kiB] kaum Einfluss. Es ergibt sich also ein Minus von ca. 300 [kiB] für die Last durch den Nachrichtenversand für den Ein-Wege-Ansatz, der durch nicht zugestellte Nachrichten zustande kommt. Für den Mehr-Wege-Ansatz ergibt sich hier ein Plus von ca. 400 [kiB]. Als Ursache bleibt hier ein deutlich höherer Aufwand, der durch den Versand von durch den Ein-Wege-Ansatz gar nicht mehr übertragenen Nachrichten verursacht wird. Dieser entsteht vermutlich durch längere Pfade und notwendige Mehrfachübertragungen.

Es zeigt sich für den Ein-Wege-Ansatz somit kein stärkerer Abfall der Effizienz mit steigender Mobilität. Statt dessen sinkt die Zuverlässigkeit deutlich stärker als für den Mehr-Wege-Ansatz.

Pfadlänge

Die Übertragungszeit als Grundlage der gewählten Metrik korreliert positiv mit der Pfadlänge, da bei jedem zusätzlichen Übertragungsschritt eine Verzögerung durch die Verarbeitung der Nachricht auf dem entsprechenden Gerät entsteht. Entsprechend zeigt sich bei den Versuchen 7, 8 und 9 sehr deutlich, dass der Mehr-Wege-Ansatz immer einen gleich langen oder häufig kürzeren Pfad zur Übertragung jeder Nachricht wählt, als der Ein-Wege-Ansatz.

Die These wird demzufolge durch die entsprechenden Ergebnisse bestätigt. Für die anderen Versuche bedeutet dies eine geringere absolute Netzlast für den Mehr-Wege-Ansatz, die sich durchgängig beobachten lässt.

Auch bei Wahl einer Metrik, die nicht positiv mit der Pfadlänge korreliert, bedeutet diese Beobachtung, dass der Mehr-Wege-Ansatz die entsprechend der Pfadkosten besseren Pfade findet.

Mehraufwand

Der tatsächliche Mehraufwand durch den Versand von Abonnements und Kündigungen für den Mehr-Wege-Ansatz ist extrem gering. Für die meisten Versuche ist er mit wenigen kiB bei einer Gesamtlast von mehreren MiB nicht wahrnehmbar.

In Versuch 8 wurden sowohl Abonnements als auch Kündigungen bei einer verhältnismäßig geringen Last zur Nachrichtenübertragung versandt. Zum Beispiel im 9. Durchlauf erzeugen die Ankündigungen eine Netzlast von ca. 40 [kiB], die Abonnements und Kündigungen etwa 3 [kiB] und die Nachrichtenübermittlung knapp 100 [kiB]. Dem Overhead durch Ankündigungen von ca. 40% ist somit ein Overhead durch Abonnements und Kündigungen von ca. 3% entgegenzusetzen.

Im Versuch 8 zeigt sich aber ebenfalls sehr deutlich, dass gerade bei den Durchläufen, in denen der Overhead durch Abonnements und Kündigungen mit ein bis drei Prozent überhaupt wahrnehmbar ist, 40 bis fast 200 Prozent Netzlast für die Nachrichtenübertragung durch die Wahl besserer Pfade eingespart wurden.

Anforderungen

Neben den zuvor diskutierten Thesen und Fragestellungen war ein weiteres Ziel der Versuche, die Ansätze hinsichtlich der Anforderungen Zuverlässigkeit, Effizienz und Skalierbarkeit zu vergleichen.

In allen Anforderungen ist der Mehr-Wege-Ansatz dem Ein-Wege-Ansatz deutlich überlegen. In vergleichbaren Konfigurationen des Szenarios werden mehr Nachrichten erfolgreich übertragen und dabei weniger Netzlast pro übertragener Nachricht erzeugt.

In den Versuchen 10 und 11 wird außerdem deutlich, dass der Mehr-Wege-Ansatz sowohl horizontal, also in Bezug auf die Größe der Netzwerktopologie, als auch vertikal, also in Bezug auf die Größe der Verarbeitungstopologie, besser skaliert.

Zusammenfassung

Im untersuchten Szenario ließen sich die Thesen Pfadbruch und Pfadlänge klar bestätigen. Die These Mobilität ließ sich nicht bestätigen. Die Höhe des tatsächlichen Mehraufwandes durch Abonnements und Kündigungen hat sich als nicht relevant herausgestellt. Außerdem konnte gezeigt werden, dass der Mehr-Wege-Ansatz in Bezug auf die Anforderungen Zuverlässigkeit, Effizienz und Skalierbarkeit dem Ein-Wege-Ansatz deutlich überlegen ist. Beide Ansätze sind jedoch selbst in einer Konfiguration des Szenarios, in dem Flooding als Algorithmus möglich ist, dem Flooding-Algorithmus in Effizienz weit überlegen. Der Ein-Wege-Ansatz ist allerdings nicht so robust wie Flooding und der Mehr-Wege-Ansatz.

Anhand dieser Ergebnisse lässt sich vermuten, dass die theoretischen Aussagen zur Komplexität und Skalierbarkeit aus den Abschnitten 5.4.1 und 5.4.2 für Szenarien mit mehreren verfügbaren Pfaden zwischen Quelle und Senke in einer statischen oder veränderlichen Umgebung mit einer Instanz eines Verarbeiters und einem Konsumenten pro Produzent zutreffen. Der Mehr-Wege-Ansatz verhält sich insbesondere durch die seltener nötigen Ankündigungsphasen und die besseren gefundenen Pfade effizienter und robuster als der Ein-Wege-Ansatz.

Aussagen über Szenarien mit wenigen oder nur einem verfügbaren Pfad, mehreren Instanzen des gleichen Verarbeiters oder mehreren Senken für gleiche Nachrichten lassen sich nicht ohne weiteres ableiten. Hier sind weitere Simulationen nötig. Um einen Eindruck vom Verhalten beider Algorithmen in einem komplexeren Szenario mit mehreren Verarbeiterinstanzen und Konsumenten sowie weniger möglichen Pfaden zu erhalten, wird außerdem ein Versuch in dem eingangs beschriebenen Bahnhofszenario durchgeführt und nachfolgend ausgewertet.

6.5. Versuch 12 – Untersuchung in einem realitätsnahen Szenario

Für die Untersuchung der Algorithmen im Bahnhofszenario werden nachfolgend die zentralen Herausforderungen in diesem Szenario erläutert. Anschließend werden die Ergebnisse des Versuchs vorgestellt und diskutiert.

6.5.1. Herausforderungen

Abbildung 6.2 gibt einen Überblick über das in Abschnitt 6.2.3 kurz beschriebene Bahnhofszenario. Es werden verschiedene Anforderungen an die Algorithmen gestellt. In einem eher stationären Umfeld aus den in drei LANs verbundenen Geräten in den Gebäuden sowie den per Ethernet an das mittlere LAN angebundenen Accesspoints auf den Bahnsteigen stellen sich die zwei folgenden Herausforderungen.

Fahrplaninformationen

Die Fahrplaninformationen, Nachrichten mittlerer Größe[4], werden zentral (`timetableSource` an `etherFront1`) einmal pro Minute im vorderen Gebäude zur Verfügung gestellt und müssen an zwei Konsumenten, einen im vorderen (`timetableSink1` an `etherFront2`) und einen im hinteren (`timetableSink2` an `EtherBack2`) Gebäude, zugestellt werden.

Überwachungsaufnahmen

Die Bilder der fünf Überwachungskameras (`surveilanceSource1...5`) auf den Bahnsteigen müssen von den Accesspoints (`apPlatform1...5`), an denen die Kameras angeschlossen sind, zu einem im zentralen Gebäude befindlichen Konsumenten (`surveilanceSink1` an `etherPlatform`) übertragen werden. Dabei handelt es sich um sehr große Nachrichten, die mit hoher Frequenz veröffentlicht werden.

Zwei weitere Herausforderungen stellen sich im mobilen Umfeld aus zwei drahtlosen WLANs nach IEEE 802.11 im Ad-Hoc-Modus. Eines bildet sich aus den Accesspoints auf den Bahnsteigen und den dort befindlichen mobilen Geräten, die sich auf definierten Strecken auf dem Bahnsteig bewegen. Ein zweites Ad-Hoc-WLAN auf einem anderen Kanal bildet sich zwischen den Accesspoints im vorderen und hinteren Gebäude und den sich jeweils dort vorbei bewegenden mobilen Geräten. Die Reichweiten der einzelnen Geräte sind durch Kreise angedeutet.

[4] Die genauen Größen der Nachrichten sowie deren Veröffentlichungsfrequenz sind in Tabelle B.13 auf Seite A-59 zusammengefasst

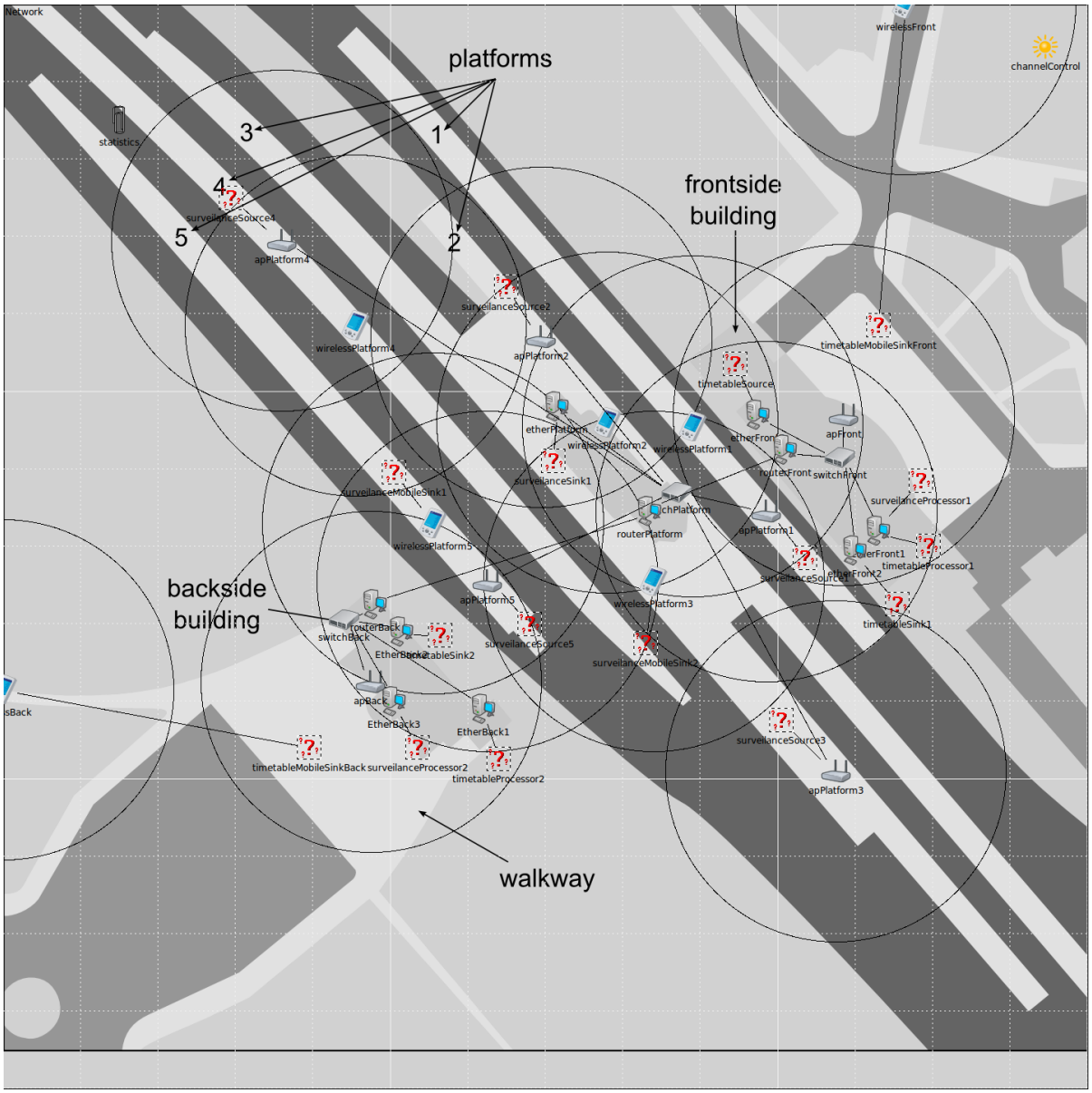


Abbildung 6.2.:
Skizze des Bahnhofszenarios. Das Gelände ist einem Bahnhof mit fünf Bahnsteigen nachempfunden. In den drei Gebäuden befinden sich jeweils verschiedene mit Ethernet verbundene Geräte. Die fünf Accesspoints auf den Bahnsteigen und die fünf dort befindlichen mobilen Geräte kommunizieren auf einem WLAN-Kanal im Ad-Hoc-Modus. Die beiden Accesspoints an den Zugängen und die Geräte außerhalb des Bahnhofs kommunizieren auf einem anderen Kanal ebenfalls im Ad-Hoc-Modus. Die Kreise um die Accesspoints und die mobilen Geräte skizzieren die simulierte Reichweite. Die unterschiedlichen Anwendungen sind durch die Kästchen mit den roten Fragezeichen symbolisiert und jeweils auf dem durch eine Linie verbundenen Gerät installiert. „ChannelControl“ und „Statistics“ sind zentrale Module für die Simulation und haben keine Bedeutung im Szenario.

Mobile Überwachungsaufnahmen

Die erste Herausforderung bezieht sich auf die Auslieferung der Überwachungsdaten der Kameras auf den Bahnsteigen 1 und 2 an den Konsumenten (`surveilanceMobileSink1`) auf dem mobilen Gerät (`wirelessPlatform5`) auf Bahnsteig 5 sowie der Daten der Kameras 4 und 5 an den Konsumenten (`surveilanceMobileSink2`) auf dem mobilen Gerät (`wirelessPlatform3`) auf Bahnsteig 3 mit möglichst geringem Nachrichtenverlust. Dabei bewegen sich die Geräte mit den Konsumenten jeweils während des Empfangszeitfensters aus dem Bereich, der durch Accesspoints abgedeckt ist hinaus und es muss automatisch auf einen Pfad über ein anderes in der Nähe befindliches mobiles Gerät ausgewichen werden. Die Daten müssen auf dem Transfer zwischen den Überwachungskameras und den jeweiligen mobilen Konsumenten in ein anderes Format mit 10% der Nachrichtengröße umgewandelt werden. Eine von zwei Instanzen des dafür nötigen Verarbeiters (`surveilanceProcessor1` an `etherFront1` sowie `surveilanceProcessor2` an `EtherBack3`) muss automatisch in den Übermittlungspfad eingebunden werden.

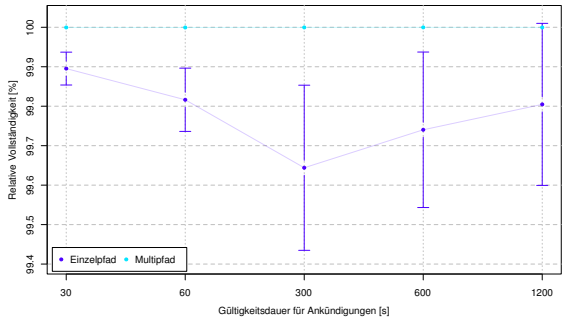
Mobile Fahrplaninformationen

Die zweite Herausforderung bezieht sich auf die möglichst schnelle Einbindung der beiden mobilen Konsumenten (`timetableMobileSink1-2`) für Fahrplaninformationen. Jeweils am vorderen und hinteren Gebäude begibt sich ein mobiles Gerät (`wirelessFront` respektive `wirelessBack`) in den Empfangsbereich des jeweiligen Accesspoints und äußert sein Interesse an Fahrplaninformationen. Das Ziel ist es trotz der geringen Veröffentlichungsfrequenz möglichst schnell den aktuellen Fahrplan an die Geräte zuzustellen. Dieser muss dabei von einer von zwei möglichen Instanzen (`timetableProcessor1` an `etherFront1` sowie `timetableProcessor2` an `EtherBack1`) eines Verarbeiters in ein für die Konsumenten verständliches Format umgewandelt werden. Die Konsumenten fragen jeweils nach vier Fahrplaninformationen. Dabei soll wenigstens die erste so schnell wie möglich zugestellt und wenigstens die letzte so aktuell wie möglich sein.

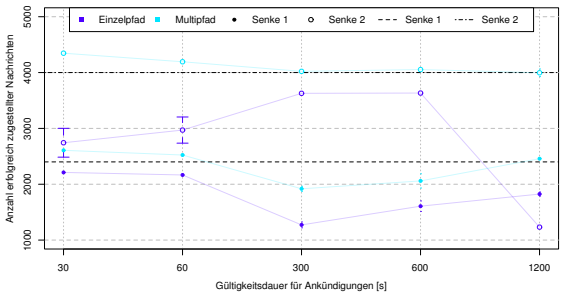
Der Versuch wird jeweils mit dem Ein-Wege-Ansatz und dem Mehr-Wege-Ansatz mit einer Gültigkeit der Ankündigungen von 30, 60, 300, 600 und 1200 [s] durchgeführt. Pro Gerät werden maximal 20 Nachrichten für jeweils maximal 60 [s] zwischengespeichert. Es werden bis zu 100 Nachrichten in Abonnements referenziert um Doppeltübertragungen zu vermeiden. Pro Konfiguration werden 100 Durchläufe simuliert.

6.5.2. Ergebnisse

Die Ergebnisse werden nachfolgend in die Bereiche Vollständigkeit, Netzlast und Übertragungszeiten untergliedert zusammengefasst und entsprechend Aussagen über die Robustheit, Effizienz und das Zeitverhalten der Algorithmen abgeleitet.



(a) Versuch 12 – Vollständigkeit statische Bilddaten



(b) Versuch 12 – Vollständigkeit mobile Bilddaten

Abbildung 6.3.:
Die erzielte Vollständigkeit bei der Übertragung der Bilder der Überwachungskameras in Abhängigkeit von der Länge der Gültigkeit der Ankündigungen mit 95 % Konfidenzintervallen für Versuch 11. Die halb transparenten Linien dienen nur der einfacheren Zuordnung der zusammengehörigen Messpunkte und tragen keinerlei Informationen. Die Abbildungen in voller Größe finden sich in Anhang B.13.

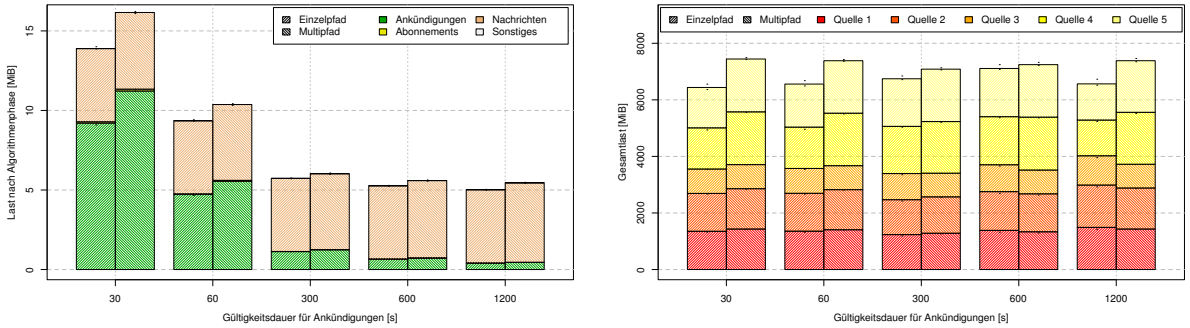
Vollständigkeit

Die Analyse der Vollständigkeit erlaubt Rückschlüsse auf die Zuverlässigkeit, die wichtigste Eigenschaft der Algorithmen im Szenario. Daher wird sie zuerst betrachtet.

Im Bereich der Fahrplaninformationen sind beide Algorithmen in der Lage, alle Nachrichten erfolgreich zu übermitteln. Dies gilt sowohl für den statischen, als auch für den mobilen Anteil.

Anders gestaltet sich dies bei den wesentlich größeren Nachrichten- und Datenmengen, die die Übertragung der Überwachungsdaten mit sich bringt. Abbildung 6.3(a) fasst die Ergebnisse für den statischen Anteil zusammen. Bei der Übertragung an den statischen Empfänger wird bereits deutlich, dass nur der Mehr-Wege-Ansatz in der Lage ist, 100 % aller Nachrichten zu übertragen. Hier sind die Nachrichtenverluste des Ein-Wege-Ansatzes mit bis zu 0,3 % aber noch sehr gering. Außerdem zeigt sich, dass die Vollständigkeit des Mehr-Wege-Ansatzes wiederum nicht von der Wahl der Gültigkeitsdauer der Ankündigungen abhängt, was für den Ein-Wege-Ansatz nicht der Fall ist.

Bei der Übertragung der Daten der entsprechenden Überwachungskameras an die mobilen Senken zeichnen die Ergebnisse ein deutlicheres Bild, zusammengefasst in Abbildung 6.3(b). Die gestrichelte bzw. gestrichpunktete Linie gibt das Soll an Nachrichten vor, also die Anzahl der Nachrichten, die jeweils von den beiden Quellen in dem Zeitraum veröffentlicht wurden, in dem die entsprechende Senke aktiv war. Die tatsächliche Anzahl zugestellter Nachrichten kann dieses Soll übersteigen weil zusätzlich Nachrichten aus den Zwischenspeichern der an der Übertragung beteiligten Geräte empfangen werden können. Auf jeden Fall ist es möglich, und Ziel beider Algorithmen, das Soll nicht zu unterschreiten. Während dies dem Mehr-Wege-Ansatz in fast allen Konfigurationen gelingt, erreicht der Ein-Wege-Ansatz



(a) Versuch 12 – Netzlast durch Fahrplandaten

(b) Versuch 12 – Netzlast durch Bilddaten

Abbildung 6.4.:
Die erzeugte Netzlast in Abhängigkeit von der Länge der Gültigkeit der Ankündigungen mit 95 % Konfidenzintervallen für Versuch 11. Die Konfidenzintervalle der einzelnen Teilbalken nehmen dabei eine fixe Grundlinie an und sind dann am oberen Ende durch zwei Punkte angedeutet. In der linken Abbildung sind die Segmente nach der Art der Nachrichten in die Phasen untergliedert. In der rechten Abbildung sind die Phasen summiert den fünf Produzenten zugeordnet, an denen die Nachrichten ihren Ursprung haben. Die Abbildungen in voller Größe finden sich in Anhang B.13.

im besten Fall ein Ergebnis knapp unter dem Soll, dies allerdings in keiner Konfiguration für beide Senken gleichzeitig.

Netzlast

Die Effizienz der Nachrichtenübertragung beider Ansätze lässt sich am Besten über die Netzlast bewerten. Hierbei ist eine Beurteilung bezogen auf die Nachrichtenquelle und den Nachrichtentyp möglich.

Abbildung 6.4(a) zeigt die Netzlast durch Fahrplaninformationen untergliedert in die Nachrichtentypen Ankündigung, Abonnement, Nachricht und Sonstiges. Dabei wird deutlich, dass die Netzlast für Abonnements und sonstige Nachrichten vernachlässigbar gering ist. Für die eigentliche Nachrichtenübertragung erzeugt der Mehr-Wege-Ansatz geringfügig mehr Netzlast. Außerdem wird hier deutlich, dass die Ankündigungsphase einen erheblichen Anteil an der erzeugten Netzlast hat. Dies ist insbesondere für Gültigkeitsdauern der Ankündigungen von 30 [s] und 60 [s] der Fall. Bei einer Veröffentlichungsfrequenz der Nachrichten von 60 [s] ist dies nicht überraschend, da hier eine erhebliche Anzahl von Ankündigungsphasen (jede zweite bei 30 [s]) vergeht, ohne dass eine einzige Nachricht übertragen wurde. Die Wahl einer sehr großen Ankündigungsgültigkeit ist für Fahrplaninformationen daher zweckmäßig.

Im Bereich der Übertragung der Überwachungsbilder sind sowohl Abonnements als auch Ankündigungen von der erzeugten Netzlast her irrelevant. Die enorme Größe und Häufigkeit der Nachrichten führt zu einem Anteil von nahezu 100 % der Netzlast für die Nachrich-

tenübertragung. Daher werden die Balken in Abbildung 6.4(b) nicht nach Nachrichtentypen, sondern nach Nachrichtenproduzenten untergliedert. Eine Abhängigkeit der Netzlast von der Ankündigungsgültigkeit lässt somit in Abhängigkeit von der Anzahl der erfolgreich übertragenen Nachrichten direkt auf die Effizienz der gewählten Pfade schließen. Die Anzahl der Ankündigungsphasen und die dadurch erzeugte Netzlast hat keinen signifikanten Anteil an der Netzlast.

Für beide Ansätze wird deutlich, dass die Nachrichten des Produzenten 3 mit Abstand am wenigsten Netzlast erzeugen. Dies liegt daran, dass sie nur an einen statischen Konsumenten übertragen werden. Die Nachrichten der anderen Produzenten werden jeweils auch an einen mobilen Konsumenten übertragen, wobei dieser für die Produzenten 4 und 5 länger aktiv ist und somit mehr Nachrichten empfängt, was sich in einem höheren Datenvolumen widerspiegelt.

Insbesondere bei den Nachrichten der Produzenten 4 und 5 erzeugt der Mehr-Wege-Ansatz deutlich mehr Netzlast. Dies liegt im Wesentlichen an der zum Teil deutlich größeren Anzahl erfolgreich zugestellter Nachrichten in diesem Bereich. Überraschend ist, dass trotz über 20 % mehr erfolgreich zugestellter Nachrichten der Produzenten 1 und 2 für eine Ankündigungsgültigkeit von 600 bzw. 1200 [s] der Mehr-Wege-Ansatz hier geringfügig weniger Netzlast erzeugt. Es werden hier offenbar durch die lange Ankündigungsgültigkeit vom Ein-Wege-Ansatz ineffizientere Pfade gewählt.

Übertragungszeiten

Für die Übertragungszeiten ergibt sich im Wesentlichen die Aussage: Beide Ansätze schaffen es, die Daten in kurzer Zeit an die Konsumenten zu übertragen. Die Fahrplaninformationen werden in 6-8 [ms] an die statischen und in 10-15 [ms] an die mobilen Konsumenten übertragen. Die Einbindung eines neuen mobilen Konsumenten, also die Zeit zwischen seinem Abonnement und der ersten von ihm empfangenen Nachricht beträgt 5-10 [ms]. Eine Abhängigkeit von der Wahl der Ankündigungsgültigkeitsdauer lässt sich nicht ausmachen.

Die Daten der Überwachungskameras werden vom Mehr-Wege-Ansatz in konstant ca. 30 [ms] übertragen. Beim Ein-Wege-Ansatz variiert dieser Zeitraum von etwas besser bei einer geringen bis zu etwas schlechter bei einer hohen Ankündigungsgültigkeitsdauer. Interessant ist der Fakt, dass die Brutto-Übertragungszeit für den Mehr-Wege-Ansatz identisch zur Netto-Übertragungszeit ist, während sie für den Ein-Wege-Ansatz etwas höher ist. Dies lässt darauf schließen, dass der Mehr-Wege-Ansatz seine perfekte Vollständigkeit nahezu ohne die mehrfache Übertragung von Nachrichten aus den Zwischenspeichern erreicht, während der Ein-Wege-Ansatz hiervon regen Gebrauch macht.

Die durchschnittliche Übertragungszeit von Überwachungsbildern an mobile Konsumenten beträgt 80-150 [ms]. Dabei benötigt der Mehr-Wege-Ansatz deutlich mehr Zeit. dies liegt vermutlich daran, dass die Nachrichten, die der Ein-Wege-Ansatz nicht mehr zustellen kann, eine besonders hohe Übertragungszeit benötigen. Mit Bestimmtheit lässt sich sagen, dass

diese Werte sich aufgrund der sehr unterschiedlichen Vollständigkeit in diesem Bereich nicht aussagekräftig miteinander vergleichen lassen.

Schlussfolgerungen

Die Ergebnisse der Simulation beider Ansätze im deutlich komplexeren Bahnhofszenario sind nicht mehr so eindeutig wie in den Testszenarien zuvor. Dem Mehr-Wege-Ansatz kann zwar auch hier bei anspruchsvolleren Nachrichtenübertragungen eine deutlich höhere Zuverlässigkeit bescheinigt werden, aber bei der Netzlast und vor allem bei den Übertragungszeiten ergibt sich ein weitaus ausgeglicheneres Bild. Nicht zuletzt auch durch die Unterschiede in der Vollständigkeit erzeugt der Mehr-Wege-Ansatz geringfügig mehr Netzlast und benötigt zum Teil mehr Zeit zur Übertragung von Nachrichten. Vor allem bei der Übertragungszeit ergibt sich aber ein sehr ausgeglichenes Bild zwischen beiden Ansätzen. Es lässt sich schlussfolgern, dass sowohl der Mehr-Wege-Ansatz als auch mit geringen Einschränkungen in der Zuverlässigkeit der Ein-Wege-Ansatz den Herausforderungen dieses Szenarios gewachsen sind.

Insbesondere für die höhere Netzlast des Mehr-Wege-Ansatzes in diesem Szenario lässt sich noch eine andere Ursache finden, die im nächsten Abschnitt näher erläutert wird.

6.6. Schwächen der Ansätze

In diesem Abschnitt werden Schwächen der beiden Ansätze analysiert, die sich in den verschiedenen Versuchen herausgestellt haben.

6.6.1. Ein-Wege-Ansatz

Abbildung 6.5 zeigt die Ergebnisse von Versuch 1 als Histogramm für die jeweilige Ankündigungsgültigkeitsdauer. Es ist klar zu erkennen, dass bestimmte Ergebnisse häufiger vorkommen als andere. Dies stellt sich wie folgt dar: Bei einer Gültigkeitsdauer der Ankündigungen von 50 [s] geht fast immer ein Vielfaches von 50 Nachrichten verloren, bei 100 [s] ist es ein Vielfaches von 100, und so weiter. Bei einer Veröffentlichungsfrequenz von einer Nachricht pro Sekunde lässt sich daraus schlussfolgern, dass vornehmlich alle oder keine Nachricht pro Ankündigungsphase verloren gehen. Dies wird in Versuch 1 besonders deutlich, weil hier die Zwischenspeicherung von Nachrichten auf den Geräten abgeschaltet ist.

Die Ursache liegt in einer grundsätzlichen Schwäche des Algorithmus. Es wird immer der zum Zeitpunkt des Empfangs des Abonnements kostengünstigste bekannte Pfad zur Quelle durch Weiterleitung des Abonnements an den entsprechenden Nachbarn freigeschaltet. Erst wenn die Quelle das Abonnement erhält, ist ihr die Senke bekannt und der Pfad wird aktiv gepflegt und bei Verlust repariert. Ist aber durch schnelle Veränderungen in der Topologie

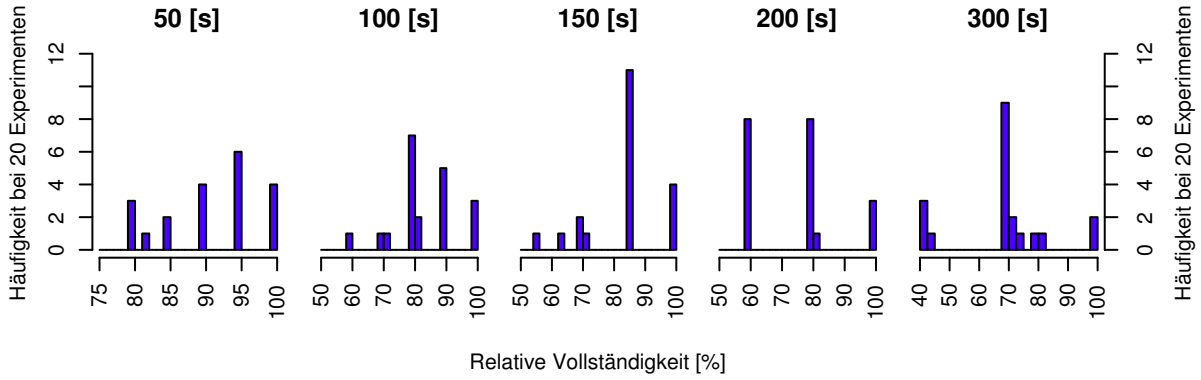


Abbildung 6.5.:
Histogramme der Vollständigkeit von Versuch 1. Jedes Histogramm repräsentiert die Ergebnisse der Durchläufe mit der darüber angegebenen Ankündigungsgültigkeitsdauer in Sekunden. Über der grob diskretisierten Vollständigkeit ist die Anzahl der Durchläufe mit dem entsprechenden Ergebnis aufgetragen.

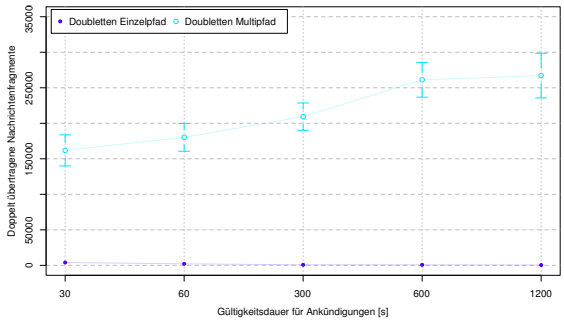
der Nachbar, von dem die Ankündigung mit dem kostengünstigsten Pfad empfangen wurde, nicht mehr verfügbar, so kann das Abonnement am entsprechenden Gerät nicht mehr weitergeleitet werden und es wird für diese Ankündigungsphase kein Pfad etabliert und entsprechend keine Nachricht zugestellt.

Die Einfachheit des Ein-Wege-Ansatzes durch die Freischaltung nur eines Pfades ist im Wesentlichen die Ursache für dessen schlechtere Robustheit bzw. Zuverlässigkeit. Geht das eine Abonnement auf dem Weg von der Senke zur Quelle verloren, wird der aktive Pfad nicht etabliert und kann somit im Anschluss auch nicht gepflegt werden.

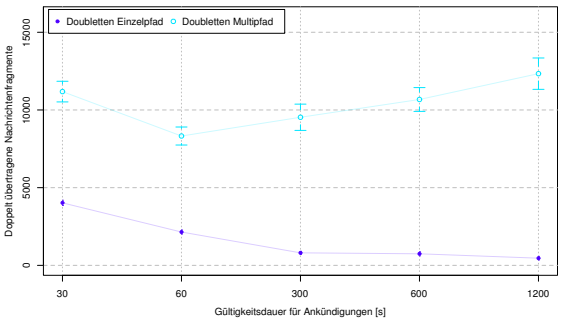
6.6.2. Mehr-Wege-Ansatz

Eine wesentliche Schwäche des Mehr-Wege-Ansatzes, wie er in dieser Arbeit vorgestellt wurde, wird bei der Betrachtung der Anzahl doppelt übertragener Nachrichten in Versuch 12 deutlich. Diese ist in Abbildung 6.6(a) dargestellt. In Versuch 12 wurden insgesamt etwa 3 Millionen Nachrichtenfragmente auf jeweils mehreren Hops pro Fragment übertragen. Trotzdem ist die Doppeltübertragung von 200.000 Fragmenten eine signifikante Menge, die eine Ursachenforschung rechtfertigt.

Ein hoher Anteil an Doppeltübertragungen in Relation zur Anzahl zu übertragener Fragmente ist bei allen Fahrplandaten sowie bei den Überwachungsbildern, während eine der beiden mobilen Senken aktiv ist, zu beobachten. Die Ursache liegt in der strikten Anwendung des Distanzvektorverfahrens. Jedes Gerät kennt den von ihm aus besten Pfad zur Senke und überträgt Nachrichten entsprechend auf diesem Pfad. Ist aber mehr als eine Senke verfügbar, so kann es vorkommen, dass die kostengünstigsten Pfade zu beiden Senken an einer Stelle, Gerät x, auseinander laufen. Auf dem kostengünstigsten Pfad zu Senke 1 hinter Gerät x ist



(a) Versuch 12 – Doppeltübertragungen unbereinigt



(b) Versuch 12 – Doppeltübertragungen bereinigt

Abbildung 6.6.:
Die Anzahl der doppelt übertragenen Nachrichtenfragmente in Abhängigkeit von der Länge der Gültigkeit der Ankündigungen mit 95 % Konfidenzintervallen für Versuch 12. Die halb transparenten Linien dienen nur der einfacheren Zuordnung der zusammengehörigen Messpunkte und tragen keinerlei Informationen. Die Abbildungen in voller Größe finden sich in Anhang B.13.

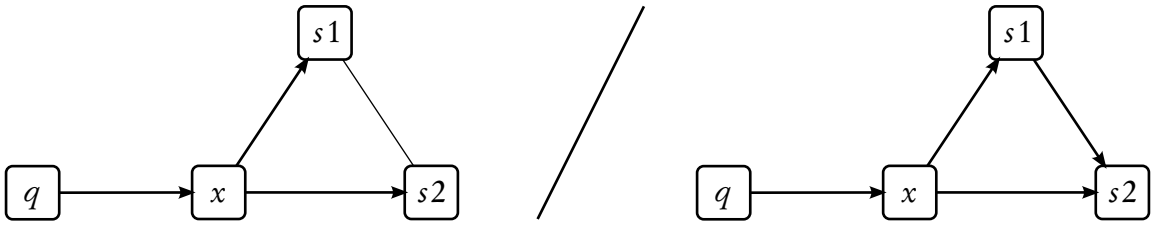


Abbildung 6.7.:
Es sind 4 Geräte dargestellt, die Quelle q, die Senken $s1$ und $s2$ sowie das Gerät x. q schickt eine Nachricht an x, da für sie die kostengünstigsten Pfade zu $s1$ und $s2$ über x laufen. Die Nachricht wird von x an $s1$ und $s2$ weitergeleitet, da sie jeweils die nächsten Geräte auf den kostengünstigsten Pfaden zu $s1$ respektive $s2$ sind. Es ergibt sich die links dargestellte Situation. $s1$ hat aber auch ein Abonnement von $s2$ erhalten und leitet somit die Nachricht auf dem günstigsten Pfad zu $s2$, nämlich an $s2$ weiter. Dieser Schritt ist algorithmisch aus der lokalen Perspektive von $s2$ korrekt, führt aber zu einer unnötigen Doppeltübertragung.

nicht mehr bekannt, dass bereits ein anderer Pfad zu Senke 2 gewählt wurde. Somit werden dort wiederum der jeweils kostengünstigste Pfad zu Senke 1 und Senke 2 gewählt. Diese laufen später wieder auf einem Gerät zusammen und dort wird die Doppeltübertragung einer Nachricht bzw. eines Fragmentes bemerkt. Abbildung 6.7 skizziert dieses Verhalten anhand eines sehr einfachen Beispiels.

Derart doppelt übertragene Nachrichtenfragmente lassen sich markieren und aus der Betrachtung der doppelt übertragenen Nachrichten für Versuch 12 heraus rechnen. Dann ergibt

sich die in Abbildung 6.6(b) dargestellte Situation. Die vorgestellte Schwäche verursacht also etwa 95 % der ursprünglich registrierten Doppeltübertragungen.

Grundsätzlich gibt es zwei Möglichkeiten, diesem Problem zu begegnen. Entweder wird jeder Nachricht eine Positivliste der von ihr noch zu erreichenden Senken mitgegeben oder eine Negativliste der nicht mehr zu erreichenden Senken. In beiden Fällen entscheidet Gerät x nicht mehr nur den nächsten Schritt der Nachrichten in Richtung der Senken, sondern auch die überhaupt von der Nachricht noch zu erreichenden Senken. Beide Möglichkeiten haben Nachteile.

Bei einer Positivliste müsste die Quelle bereits alle bekannten Senken an die Nachricht anhängen und die Nachricht wird auch nur an diese Senken zugestellt. Eine neu in das System integrierte Senke müsste demzufolge noch im System zwischengespeicherte und für die Senke gegebenenfalls interessante Nachrichten immer an der Quelle abgreifen, was wiederum zu Doppeltübertragungen führt. Andere Geräte dürfen die bei ihnen zwischengespeicherten Nachrichten nicht an die neue Senke weiterleiten, da diese nicht in deren Positivliste vermerkt ist.

Bei einer Negativliste tritt dieser Nachteil nicht auf. Allerdings kann es hier passieren, dass die Nachricht gleichzeitig aus dem Zwischenspeicher mehrerer Geräte abgegriffen und somit wiederum unnötig oft übertragen wird. Außerdem wird die Nachricht bzw. ein Nachrichtenfragment durch das Anhängen weiterer Senken an die Negativliste immer größer und muss gegebenenfalls weiter fragmentiert werden, wodurch der Overhead der Nachrichtenübertragung ansteigt.

Im Rahmen dieser Arbeit wurde keine der beiden Möglichkeiten umgesetzt, da der Mehr-Wege-Ansatz trotz der hohen Anzahl an Doppeltübertragungen im Bahnhofszenario bereits überzeugen konnte und somit eine weitere Verbesserung zwar wünschenswert aber für den Zweck der erfolgreichen Simulation in den beschriebenen Szenarien nicht notwendig ist.

6.7. Zusammenfassung

In diesem Kapitel wurden verschiedene Annahmen und Fragestellungen zum Einfluss bestimmter Parameter der Ansätze, zur Reaktion der Ansätze auf bestimmte Systemeigenschaften sowie zu bestimmten Anforderungen an dieselben formuliert. Im Anschluss wurden Versuche in einem Testszenario entworfen und die Ergebnisse der entsprechenden Simulationen diskutiert. Zur Überprüfung der gewonnenen Erkenntnisse wurde das Verhalten der Ansätze in einem komplexeren, der Realität ähnlicheren Szenario simuliert und die Ergebnisse entsprechend ausgewertet.

Die Ergebnisse zeigen in der Simulation im Testszenario eine deutliche Überlegenheit des Mehr-Wege-Ansatzes gegenüber dem Ein-Wege-Ansatz. Die Vorteile im Einzelnen sind

- eine höhere Robustheit, die sich durch eine deutlich höhere Zuverlässigkeit unter verschiedensten Bedingungen etablierte,

- eine weitgehende Unabhängigkeit der Zuverlässigkeit des Mehr-Wege-Ansatzes von der gewählten Gültigkeitsdauer der Ankündigungen,
- die Wahl besserer Pfade entsprechend der Pfadkosten durch die gewählte Metrik,
- ein besseres Timing durch die Wahl besserer Pfade,
- eine bessere Skalierbarkeit sowohl gegenüber einer Steigerung der Komplexität der Netzwerktopologie als auch der Verarbeitungstopologie, sowie
- ein effizienterer Umgang mit den Netzwerkressourcen durch weniger erzeugte Netzlast pro erfolgreich übertragener Nachricht.

Diese Ergebnisse sind allerdings grundsätzlich nur für das untersuchte Szenario mit den entsprechenden Eigenschaften gültig. Sie lassen sich vermutlich auf ähnliche Szenarien mit vielen verschiedenen verfügbaren Pfaden und der Nutzung einer einzigen Quelle, eines Verarbeiter und einer Senke übertragen.

Bei der Überprüfung der Ergebnisse in dem komplexeren, der Realität näheren Bahnhofszenario lassen sich die überlegene Robustheit, die weitgehende Unabhängigkeit der Zuverlässigkeit von der gewählten Gültigkeitsdauer der Ankündigungen sowie Anzeichen für die Wahl besserer Pfade entsprechend der Pfadkosten bestätigen.

Außerdem wurden Schwächen beider Algorithmen aufgedeckt. In Ein-Wege-Ansatz lässt sich durch einen möglichen Pfadbruch vor dessen vollständiger Etablierung die zum Teil deutlich schlechtere Zuverlässigkeit erklären. Im Mehr-Wege-Ansatz ist insbesondere die deutlich höhere Netzlast im Bahnhofszenario durch die hohe Anzahl an Doppeltübertragungen auf dem Weg zu mehreren Senken zu erklären. Hierfür wurden zwei mögliche Erweiterungen des Ansatzes diskutiert, diese Schwäche zu beheben.

Aufgrund der Modellierung und der Eigenschaften der Simulation sind die Ergebnisse in Bezug auf das Zeitverhalten nicht direkt auf die Realität zu übertragen. Aber insbesondere die höhere Zuverlässigkeit des Mehr-Wege-Ansatzes und deren weitestgehende Unabhängigkeit von der gewählten Ankündigungsgültigkeitsdauer, die sich auch im Bahnhofszenario bestätigt haben, sowie dessen bessere Skalierbarkeit sind deutliche Indizien für eine Überlegenheit gegenüber dem Ein-Wege-Ansatz und einfachen Ansätzen wie Flooding und damit einer grundsätzlichen Nutzbarkeit in einer realen Implementierung.

Kapitel 7.

Zusammenfassung und Ausblick

Inhalt

7.1 Ergebnisse der Arbeit **158**
7.1.1 Taxonomie zur Untersuchung von Kommunikationsparadigmen 158
7.1.2 Konzept der Vermittlungstopologie 159
7.1.3 Entwicklung zweier geeigneter Algorithmen 162
7.1.4 Untersuchung und Vergleich der Algorithmen 164
7.2 Offene Forschungsfragen **166**
7.2.1 Weitere Optimierung der vorgestellten Ansätze 166
7.2.2 Sicherheit 167
7.2.3 Persistenz 167
7.2.4 Integration bestehender Anwendungen 168
7.3 Schlusswort **168**

In diesem Kapitel werden die Ergebnisse der Arbeit zusammengefasst und ein Ausblick auf offene Forschungsfragen gegeben.

7.1. Ergebnisse der Arbeit

Ziel dieser Arbeit war es, einen Beitrag zur Lösung des Problems der inhaltlich entkoppelten, nachrichtenbasierten Kommunikation in heterogenen, veränderlichen Umgebungen zu leisten. Dies fand in vier Schritten statt. Im ersten Teil der Arbeit wurde das Problem anhand einer geeigneten Taxonomie der Entkopplung eingeordnet und bestehende Kommunikationsparadigmen auf deren Eignung zur Lösung des Problems hin untersucht. Anschließend wurde das Konzept der Vermittlungstopologie begleitet von einer geeigneten Systemarchitektur zur Abbildung auf heterogene, veränderliche Netzwerktopologien sowie einem Vorschlag für eine geeignete Metrik zur Bewertung gefundener Pfade vorgestellt.

Im dritten Teil der Arbeit wurden zwei geeignete Algorithmen entwickelt, mit denen Produzenten und Konsumenten in einer Vermittlungstopologie zusammengeführt und Nachrichten zwischen beiden übertragen werden können. Das Verhalten beider Algorithmen im Vergleich zueinander und zu einem verfügbaren Referenzalgorithmus wurde im letzten Abschnitt anhand von Simulation in einem speziell dafür zugeschnittenen sowie einem realitätsnäheren Szenario untersucht.

Die Ergebnisse und die jeweiligen Beiträge zum Stand der Forschung werden nachfolgend für jeden dieser vier Abschnitte dargestellt.

7.1.1. Taxonomie zur Untersuchung von Kommunikationsparadigmen

Die Beiträge des ersten Teils sind

- die in der Arbeit entwickelte Taxonomie der Entkopplung sowie
- die Untersuchung und Einordnung bestehender Kommunikationsparadigmen und der Problemstellung anhand dieser Taxonomie.

Die Taxonomie besteht dabei aus fünf im Wesentlichen voneinander unabhängigen Dimensionen:

- räumliche Entkopplung,
- zeitliche Entkopplung,
- Entkopplung des Produzenten von der Middleware,
- Entkopplung des Konsumenten von der Middleware, sowie
- inhaltliche Entkopplung.

Die Einordnung des Problems und bestehender Kommunikationsparadigmen anhand der Taxonomie kommt zu dem Ergebnis, dass die nachrichtenbasierten Paradigmen Publish/Subscribe und Tuple Spaces jeweils die geforderten Entkopplungseigenschaften in den

Bereichen räumlicher, inhaltlicher und Entkopplung vom Produzenten gewährleisten können.

Bei der zeitlichen Entkopplung fordert die Problemstellung sowohl die gegenwärtige Entkopplung, also die Zustellung aller Nachrichten an alle der Middleware zum Zeitpunkt des Empfangs der Nachricht bekannten Konsumenten, sowie eingeschränkt auch die zukünftige Entkopplung, die Zustellung der jeweils aktuell letzten Nachrichten an zukünftig bekannt werdende Konsumenten.

Die gegenwärtige Entkopplung ist durch das Publish/Subscribe-Paradigma realisierbar. Die zukünftige Entkopplung in der geforderten Ausführung erfordert eine Erweiterung des Paradigmas. Tuple Spaces hingegen sehen eine zukünftige zeitliche Entkopplung vor, während eine gegenwärtige Entkopplung das Paradigma verletzen würde oder wenigstens einschneidende Erweiterungen notwendig macht.

In Bezug auf die Entkopplung vom Konsumenten erfordert die Problemstellung sowohl die kontinuierliche Anfrage, also den Empfang aller verfügbaren Nachrichten nach einer Registrierung für den entsprechenden Typ von Nachrichten, als auch die Registrierung mit Rückfrage, also die Registrierung mit einer sehr allgemeinen Anfrage und einer Verfeinerung derselben anhand der durch die Middleware mitgeteilten, auf diese Anfrage passenden Nachrichtentypen.

Während Publish/Subscribe die kontinuierliche Anfrage problemlos ermöglicht, ist eine entsprechende Erweiterung von Tuple Spaces nötig aber nicht problematisch. Die Registrierung mit Rückfrage lässt sich, ohne das Publish/Subscribe-Paradigma zu verletzen realisieren. Mit Tuple Spaces ist dies nicht ohne weiteres möglich.

Zuletzt begründen die speziellen Eigenschaften beider Paradigmen, für Tuple Spaces die Datenpersistenz und für Publish/Subscribe die besondere Eignung für Nachrichtenströme, die Aussage, dass Publish/Subscribe das für die Lösung der Problemstellung am besten geeignete Kommunikationsparadigma ist.

7.1.2. Konzept der Vermittlungstopologie

Die Beiträge dieses Teils sind

- die Vermittlungstopologie zur Abbildung der Problemstellung auf ein Routingproblem,
- die Systemarchitektur zur Abbildung der Vermittlungstopologie auf eine heterogene, veränderliche Netzwerktopologie,
- die Analyse inhaltsbasierter Routingalgorithmen auf ihre Eignung zur Gewährleistung der gewünschten inhaltlich entkoppelten Kommunikation in der Vermittlungstopologie,
- die Beschreibung von Anforderungen an eine geeignete Metrik zur Bewertung der Pfade und die Vorstellung einer solchen, sowie

- die Definition von Anforderungen an entsprechende Anwendungen zur Nutzung der konzipierten Lösung.

Die Ergebnisse werden im Einzelnen in den folgenden Abschnitten kurz zusammengefasst.

Vermittlungstopologie

Die Vermittlungstopologie entsteht durch die Kombination der Netzwerktopologie, bestehend aus den Nachbarschaftsbeziehungen der Geräte auf Basis von Kommunikationsverbindungen, mit der Verarbeitungstopologie, bestehend aus den Nachbarschaftsbeziehungen zwischen Nachrichtentypen auf Basis der Verfügbarkeit geeigneter Verarbeiter.

Eine Untersuchung der Komplexitätseigenschaften der Vermittlungstopologie ergibt einen steigenden Einfluss auf die Komplexität in der folgenden Reihenfolge:

- zusätzlicher Verarbeiter
- zusätzliche Nachbarschaftsbeziehung
- zusätzliches Gerät mit den dazugehörigen Nachbarschaftsbeziehungen zu bestehenden Geräten
- zusätzlicher Nachrichtentyp mit den dazugehörigen Nachbarschaftsbeziehungen zu bestehenden Nachrichtentypen

Zustandsbehaftete und zustandslose Datenverarbeitung werden definiert und daraus geschlossen, dass zustandsbehaftete Verarbeiter alle für sie relevanten Ausgangsnachrichten erhalten müssen, um ihren Zustand korrekt daraus ableiten zu können. Auf verfügbare zustandslose Verarbeiter können die zu verarbeitenden Nachrichten hingegen wahllos verteilt werden.

Systemarchitektur

Pro Gerät wird ein Broker implementiert, der die Knoten aller Nachrichtenzustände für dieses Gerät entsprechend der Vermittlungstopologie verwaltet. Von den Netzwerkschnittstellen wird jeder Broker durch eine Netzwerkabstraktionsschicht entkoppelt, von den Anwendungen durch eine Anwendungsabstraktionsschicht. Dazwischen befindet sich die Vermittlungsschicht, in der der Routingalgorithmus arbeitet.

Die Systemarchitektur erlaubt eine vollständig verteilte Architektur, bei der jede Anwendung nur mit ihrem lokalen Broker kommuniziert und eine maximale Entkopplung der Geräte untereinander gewährleistet ist.

Durch die Abstraktion des Problems als Pfadsuche in der Vermittlungstopologie kombiniert mit der Systemarchitektur wird die Pfadsuche durch das Routing vollständig von der Heterogenität der Kommunikationsverbindungen sowie der inhaltlichen Vielfalt der Nachrichten entkoppelt. Der Routingalgorithmus muss auf diese Anforderungen folglich keine Rücksicht mehr nehmen.

Eignung von grundlegenden Routingalgorithmen

Ein hoher Grad an Effizienz verbunden mit der integrierten Verbreitung der Verfügbarkeit aller im System veröffentlichten Nachrichtentypen unter allen Brokern zeichnet das inhaltsbasierte Routing mit Ankündigungen als vielversprechendste Grundlage für einen Routingalgorithmus aus. Es erlaubt sowohl effizientes Routing in der Vermittlungstopologie, als auch Registrierung mit Rückfrage als Entkopplungseigenschaft.

Metriken

eine Metrik für das Routing in einer Vermittlungstopologie muss sowohl die Kommunikation über Netzwerkverbindungen, als auch die Nachrichtenverarbeitung vergleichbar bewerten können. Daher eignet sich die Zeit in Form der Übertragungszeit sowie der Verarbeitungszeit als Basis für eine solche Metrik hervorragend. Außerdem muss sie die Eigenschaft berücksichtigen, dass in der Vermittlungstopologie nicht nur unterschiedlich große Nachrichten übermittelt werden, sondern sich deren Anzahl und Größe während der Übertragung zwischen Produzent und Konsument verändern können.

Eine Metrik, mit der Zeit als Basis und einem Faktor zur Gewichtung einzelner Verbindungen durch die Middleware, die diese Eigenschaften erfüllt, sowie geeignete Aggregationsoperationen wurden in der Arbeit vorgestellt.

Anforderungen an die Anwendungen

Zur Diskussion unterschiedlicher Anforderungen an die Wahl entsprechender Routingalgorithmen, wurde für Anwendungen eine Klassifizierung anhand zweier Merkmale vorgestellt. Diese sind die Anzahl von der Anwendung veröffentlichter Nachrichten mit den diskreten Ausprägungen Einzelnachricht und Nachrichtenstrom sowie die Größe der Nachrichten mit den diskreten Ausprägungen in eine Ankündigung passend sowie größer.

Besonderheiten der Schnittstellen für Anwendungen und die speziellen Anforderungen an die kommunizierten Inhalte wurden diskutiert. Dabei erlaubt das Konzept für die Inhalte im Extremfall eine vollständige Inhaltsdefinition durch die Anwendung sowohl, was Ankündigungen, als auch Filter zur Einschränkung von Abonnements und die Nachrichten selbst angeht. Lediglich die maximale Größe von Ankündigungen wird aus Effizienzgründen vom Routingalgorithmus bzw. der Middleware vorgegeben. Weitere Einschränkungen an die Definition von Formaten für die Inhalte durch Routingalgorithmen sind möglich, um deren Effizienz zu erhöhen.

7.1.3. Entwicklung zweier geeigneter Algorithmen

Der Beitrag dieses Teils besteht in der Entwicklung und theoretischen Analyse zweier Routingansätze zur Verbreitung der Verfügbarkeit von Nachrichten und der Übermittlung derselben von Produzenten zu Konsumenten in einer Vermittlungstopologie.

Die Ergebnisse bestehen im Einzelnen aus den Ansätzen selbst, der Diskussion der Komplexitätseigenschaften sowie der Skalierungsfähigkeit beider Ansätze, der Diskussion des Einflusses der Anwendungen auf die Parameter der Algorithmen und zuletzt der Ableitung von in der Simulation zu prüfenden Aussagen und Fragestellungen.

Routingansätze

Beiden Routingansätzen ist gemein, dass sie aus drei Phasen bestehen. In der ersten Phase wird die Verfügbarkeit von Nachrichten eines bestimmten Typs durch Ankündigungen in der Vermittlungstopologie verbreitet. Für diese Phase kommt der Flooding-Algorithmus zur Verbreitung der Ankündigungen zum Einsatz. In der zweiten Phase werden Nachrichtentypen durch interessierte Konsumenten abonniert und in der dritten Phase die entsprechenden Nachrichten übermittelt.

Die Ankündigungen werden von der Nachrichtenquelle mit einer Gültigkeit versehen, so dass ein explizites ungültig machen nicht nötig ist. Für den Fall, dass mehrere Nachrichten mit unterschiedlicher Gültigkeit durch einen Verarbeiter zu einer Ankündigung eines anderen Typs aggregiert werden, wurden verschiedene Verfahren zur Berechnung der Gültigkeit der generierten Ankündigung diskutiert. Außerdem wurde die Notwendigkeit der Zwischenspeicherung von Ankündigungen auf möglichst vielen Brokern zur Gewährleistung einer zügigen Anbindung neuer Nachrichtenkonsumenten herausgestellt.

Speziell für den Ein-Wege-Ansatz ist die Etablierung genau eines Pfades von den benötigten Produzenten zu einem Konsument in der Abonnementphase. Dieser wird anschließend zur Übermittlung aller Nachrichten genutzt. Wird er durch Veränderungen in der Vermittlungstopologie unbenutzbar, muss ein neuer Pfad etabliert werden. Ein entsprechendes Vorgehen zur Signalisierung von Pfadbrüchen in Richtung der Quelle wurde vorgestellt.

Im Gegensatz dazu werden beim Mehr-Wege-Ansatz in der Abonnementphase alle möglichen Pfade anhand ihrer Pfadkosten bewertet und erst bei der Nachrichtenübermittlung der nach Pfadkosten optimale Pfad ausgewählt. Bei Verlust dieses optimalen Pfades stehen automatisch alternative Pfade zur Verfügung. Signalisierungsmechanismen, die gewährleisten, dass nach einem Pfadverlust wieder der global optimale Pfad genutzt wird, wurden diskutiert.

Komplexität

Die Komplexitätseigenschaften in Form von Speicherbedarf und Kommunikationsaufwand beider Algorithmen wurde formal untersucht. Der Speicherbedarf pro Broker für beide An-

sätze hängt ausschließlich von den Eigenschaften der Vermittlungstopologie ab. Er ist für den Ein-Wege-Ansatz deutlich geringer als für dem Mehr-Wege-Ansatz, da jener sowohl in der Ankündigungs-, als auch in der Abonnementphase mehr Statusinformationen speichern muss.

Für den Kommunikationsaufwand beider Algorithmen konnten zusätzlich zwei wesentliche Eigenschaften derselben identifiziert werden, die hier einen erheblichen Einfluss haben. Dies sind die Häufigkeit notwendiger Ankündigungsphasen sowie die Länge der letztendlich zur Nachrichtenübertragung genutzten Pfade. Während der Mehr-Wege-Ansatz in der Ankündigungs- wie Abonnementphase bezogen auf die Eigenschaften der Vermittlungstopologie mehr Kommunikationsaufwand erzeugt, ist hier mit einer deutlich günstigeren Ausprägung der beiden erwähnten Eigenschaften zu rechnen, so dass der in Summe erzeugte Kommunikationsaufwand erheblich niedriger ausfallen könnte, als für den Ein-Wege-Ansatz.

Skalierungsfähigkeit

Die Diskussion der Skalierungsfähigkeit beider Algorithmen ergab einen erheblichen Einfluss der Anzahl der Nachrichtentypen in der Verarbeitungstopologie, gefolgt von der Differenz zwischen der Anzahl der Kommunikationsverbindungen und der Geräte in der Netzwerktopologie. Einen geringen Einfluss haben die Anzahl der Verarbeiter, Quellen und Senken. Da das bestimmende Element für die Skalierungsfähigkeit die Nutzung des Flooding-Algorithmus in der Ankündigungsphase ist, gelten diese Angaben für beide Ansätze gleichermaßen.

Verschiedene Vorschläge zur Begrenzung der einflussreichsten Faktoren wurden diskutiert. Die Anzahl der Nachrichtentypen lässt sich durch die Einschränkung der Ausbreitung derselben in der Vermittlungstopologie begrenzen. Für spezielle, besonders ungünstige Formen der Netzwerktopologie wurden verschiedene Optimierungsmöglichkeiten vorgestellt.

Anwendungseinfluss auf Parameter

Die Steuerung der Algorithmen erfolgt im Wesentlichen über vier Parameter. Dies sind die

- Gültigkeitsdauer der Ankündigungen,
- die Anzahl zwischengespeicherter Nachrichten in Brokern,
- deren Gültigkeitsdauer im Zwischenspeicher, sowie
- die Anzahl der in Abonnements referenzierten Nachrichten.

Für drei dieser Parameter ist zur optimalen Einstellung eine Steuerung durch die Anwendungen sinnvoll. Dies sind die Gültigkeitsdauer sowie die beiden Einstellungen für die Zwischenspeicherung.

Aussagen und Fragestellungen zur simulativen Überprüfung

Aus den beschriebenen Diskussionen wurden die sieben folgenden Aussagen bzw. Fragestellungen extrahiert, die entweder als Annahmen oder Schlussfolgerungen in die Diskussion eingeflossen sind und einer Überprüfung bedürfen.

- Die Pfadnutzungsdauer in veränderlichen Topologien ist für den Mehr-Wege-Ansatz signifikant länger als für den Ein-Wege-Ansatz.
- Die Länge der gewählten Pfade ist bei Wahl einer Metrik, die mit der Pfadlänge positiv korreliert, für den Mehr-Wege-Ansatz erkennbar kleiner als für den Ein-Wege-Ansatz.
- Eine längere Gültigkeitsdauer der Ankündigungen führt zu weniger Kommunikationsaufwand durch die Verbreitung der Ankündigungen per Flooding.
- Eine Vergrößerung der Anzahl der zwischengespeicherten Nachrichten führt zum Verlust von weniger Nachrichten.
- Eine Erhöhung der Anzahl in Abonnements referenzierter Nachrichten verringert die doppelte Übertragung von Nachrichten durch deren Zwischenspeicherung, erhöht aber den Kommunikationsaufwand durch größere Abonnements.
- Wie groß ist der zusätzliche Kommunikationsaufwand durch zusätzlich versandte Abonnements und eventuell Kündigungen im Mehr-Wege-Ansatz tatsächlich?
- Verliert der Ein-Wege-Ansatz durch mehr Mobilität tatsächlich stärker an Effizienz als der Mehr-Wege-Ansatz?

7.1.4. Untersuchung und Vergleich der Algorithmen

Der Beitrag des letzten Teils besteht in der Bestätigung bzw. Widerlegung der aus der theoretischen Analyse gewonnenen Aussagen sowie der Beantwortung der entsprechenden Fragestellung anhand von Simulation der Ansätze in einem speziell darauf zugeschnittenen Szenario. Dazu kommen die Beurteilung der beiden Ansätze durch den Vergleich untereinander und mit einem Referenzansatz sowie die Ermittlung von Stärken und Schwächen der Ansätze auch durch die Simulation in einem komplexeren, realitätsnäheren Szenario.

Die Ergebnisse im Einzelnen werden untergliedert nach der Beurteilung der gemachten Aussagen bzw. Fragestellungen, der Einordnung der Algorithmen sowie der ermittelten Schwachpunkte.

Beurteilung der Aussagen und Fragestellungen

Ein spezielles Test-Szenario zur Beurteilung bestimmter Verhaltenseigenschaften der Ansätze wurde entwickelt. Dabei standen die Konfigurierbarkeit

- der Größe der Topologie,
- des Grad der Veränderlichkeit der Topologie,
- der Größe und Anzahl veröffentlichter Nachrichten, sowie

- der Parameter der untersuchten Ansätze und Algorithmen

im Mittelpunkt.

Die untersuchten Eigenschaften der Ansätze wurden so gewählt, dass Aussagen über deren Zuverlässigkeit, die Güte gewählter Pfade sowie deren Effizienz möglich sind. Im Einzelnen sind dies die Vollständigkeit und die Netto-Latenz der Nachrichtenübermittlung, die dadurch erzeugte Netzlast, der Overhead in verschiedenen Phasen und die Anzahl der Doppeltübertragungen sowie spezielle Eigenschaften wie die Pfadnutzungsdauer und die Anknüpfungslatenz.

Zur Validierung der Simulationsumgebung wurden die Ergebnisse in einer speziellen Konfiguration des Test-Szenarios, in der sich bestimmte ermittelte Eigenschaften theoretisch exakt berechnen lassen, überprüft.

Bis auf eine Ausnahme konnten alle in der theoretischen Analyse gemachten Aussagen durch die Simulation im Test-Szenario bestätigt werden. Die letzte Aussage konnte nicht bestätigt werden. Sowohl Ein- als auch Mehr-Wege-Ansatz verlieren bei steigender Mobilität gleich stark an Effizienz. Dabei ist allerdings die Zuverlässigkeit und die absolute Effizienz des Mehr-Wege-Ansatzes dem Ein-Wege-Ansatz stets überlegen.

Die Frage nach dem tatsächlichen Anstieg des Kommunikationsaufwandes durch im Mehr-Wege-Ansatz zusätzlich versandte Abonnements und Kündigungen konnte dahingehend beantwortet werden, dass dieser zwar messbar aber vernachlässigbar gering ist. Bereits eine Verbesserung der Länge der zur Nachrichtenübermittlung genutzten Pfade um einen Hop führt zu einer weit größeren Verringerung des Kommunikationsaufwandes.

Einordnung der Algorithmen

Verglichen mit der Nutzung des Flooding-Algorithmus zur Verbreitung von Nachrichten in einem Szenario mit sehr geringer Nachrichtengröße und Frequenz, wo dies möglich ist, sind beide Ansätze an Effizient weit überlegen. Lediglich der Ein-Wege-Ansatz ist hier weniger robust als Flooding.

Der Mehr-Wege-Ansatz ist dem Ein-Wege-Ansatz unter den meisten Bedingungen insbesondere in der Zuverlässigkeit und Effizienz deutlich überlegen. Außerdem hängt seine Zuverlässigkeit im Gegensatz zum Ein-Wege-Ansatz deutlich weniger von der Wahl der Gültigkeitsdauer der Ankündigungen ab. Der Mehr-Wege-Ansatz skaliert sowohl bezogen auf die Größe der Netzwerktopologie als auch die der Verarbeitungstopologie besser als der Ein-Wege-Ansatz. Durch den Mehr-Wege-Ansatz werden bessere Pfade gefunden und genutzt als durch den Ein-Wege-Ansatz.

In einem zweiten, komplexeren und der Realität näheren Szenario relativieren sich diese Vorteile allerdings. Insbesondere die Überlegenheit im Bereich der Effizienz ist weit weniger deutlich.

Ermittelte Schwachpunkte

Für den Ein-Wege-Ansatz konnte eine wesentliche Schwäche identifiziert werden. Bei der Unterbrechung bestimmter Kommunikationsverbindungen in der sehr kurzen Zeit zwischen der Ankündigungsphase und der Abonnementphase kann es passieren, dass kein Pfad etabliert wird und somit während der gesamten Gültigkeitsdauer der Ankündigung keine Nachricht übertragen wird.

Während diese Schwäche nur mit umfangreichen Veränderungen am Ansatz zu umgehen ist, konnten für eine ebenfalls identifizierte Schwäche des Mehr-Wege-Ansatzes zwei Lösungsvorschläge entworfen werden. Diese Schwäche führt zur Doppeltübertragung von Nachrichten, wenn mehrere Konsumenten für den gleichen Nachrichtentyp verfügbar und hinter der Verzweigung der optimalen Pfade zu beiden Konsumenten andere Pfade zu dem jeweils anderen Konsumenten verfügbar sind. Diese Schwäche lässt sich durch das Anhängen einer Positivliste noch zu erreichender oder eine Negativliste nicht mehr zu erreichender Konsumenten an die Nachrichten überwinden.

Neben diesen identifizierten Schwächen der Ansätze gibt es auch Bereiche, die dem Umfang der Arbeit geschuldet vollständig ausgespart werden mussten. Einige dieser werden nachfolgend als offene Forschungsfragen vorgestellt.

7.2. Offene Forschungsfragen

Selbstverständlich ist es in einer einzelnen Arbeit nicht möglich, ein so umfassendes Problem wie die inhaltlich entkoppelte Kommunikation zwischen Produzenten und Konsumenten in einer heterogenen, veränderlichen Umgebung in all seinen Details zu lösen. Daher werden im folgenden Abschnitt weitere, sich aus den Ergebnissen der Arbeit ergebene Forschungsfragen vorgestellt.

Diese sind in die vier Bereiche, weitere Optimierung der Ansätze, Gewährleistung umfassender Sicherheitseigenschaften, Ergänzung um Persistenz von Daten im System sowie Erweiterungen zur Integration bestehender Anwendungen in ein entsprechendes System, untergliedert.

7.2.1. Weitere Optimierung der vorgestellten Ansätze

Für verschiedene Schwächen bzw. bewusst einfache Implementierungsentscheidungen der Ansätze wurden in den entsprechenden Abschnitten bereits Optimierungen vorgestellt. Die tatsächliche Untersuchung des Optimierungspotentials wurde bewusst ausgespart. Neben dessen praktischer Evaluation ergeben sich weitere Fragestellungen:

- Lässt sich die Effizient der Ansätze durch weitere Optimierungen in der Ankündigungsphase ohne einen Verlust an Robustheit und Zuverlässigkeit steigern?

- Lassen sich größere und komplexere Szenarien durch weitere Optimierung der Ansätze oder geschickte Konfiguration realisieren und somit die Skalierbarkeit der Ansätze verbessern?
- Gibt es alternative, effizientere Ansätze, die das Problem ohne Verlust an Robustheit und Flexibilität lösen können?
- Kann durch die Aufgabe der Flexibilität in bestimmten Aspekten, zum Beispiel bei der Unabhängigkeit von der Kommunikationstechnologie, eine deutliche Verbesserung der Effizienz erreicht werden?

7.2.2. Sicherheit

Insbesondere die räumliche und inhaltliche Entkopplung stellen völlig neue Anforderungen an Sicherheitsmechanismen. Davon sind sowohl die Datensicherheit, als auch die Systemsicherheit betroffen.

Neben dem Schutz vor dem Ausspähen von Daten ist insbesondere die Vermeidung unzulässiger Datenmanipulation in einem System, in dem von Produzenten und Konsument unbemerkte Datenverarbeitung die Regel ist, eine besondere Herausforderung. Ähnlich verhält es sich mit dem Schutz vor der Einspeisung falscher Daten in einem System, in dem die Produzenten von den Konsumenten räumlich entkoppelt sind.

In Bezug auf die Systemsicherheit ist insbesondere die Gewährleistung, dass nur Geräte - Produzenten, Konsumenten, Verarbeiter, Broker - am System teilnehmen, deren Teilnahme auch erwünscht ist. Eine entsprechende Authentifizierung von Teilnehmern oder gar ein umfangreiches Rechtesystem darf die Flexibilität und lose Kopplung der Teilnehmer nicht beschränken.

7.2.3. Persistenz

Die Anforderungsanalyse in dieser Arbeit fand hauptsächlich unter dem besonderen Augenmerk der spontanen intelligenten Umgebungen statt, in denen viele Informationen ständig aktualisiert werden. Somit müssen Nachrichten nur für kurze Zeit im System verfügbar sein und werden dann durch aktuellere Nachrichten ersetzt.

In anderen Anwendungsbereichen kann es erwünscht sein, eine längere oder gar ständige Persistenz von Daten in der Middleware zu realisieren. Diese ist durch eine temporäre Zwischenspeicherung auf möglichst vielen Geräten, wie sie im vorgestellten System zur Gewährleistung von Robustheit und Effizienz durchgeführt wird, nicht sicherzustellen.

Hier ergeben sich Fragestellungen nach dem Ort der Datenspeicherung, der Steuerung der Datenspeicherung und dem Zugriff auf alte Daten. Auf wie vielen und welchen Geräten ist eine Speicherung von Daten sinnvoll, um eine möglichst zuverlässige Datenpersistenz zu realisieren? Wie lange speichert man diese Daten und welche Geräte oder Anwendungen kontrollieren die Dauer der Datenspeicherung oder stellen die Löschung von Daten sicher?

In welcher Form speichert man Daten? Speichert man nur Ausgangsdaten oder speichert man auch daraus generierte Daten weil nicht sicher ist, ob zu einem späteren Zeitpunkt noch geeignete Verarbeiter im System verfügbar sind?

Eine zentrale Fragestellung ist auch, wie man überhaupt auf alte Daten zugreift. Eine ständige Verbreitung alter Daten in Form von Ankündigung und Abonnement zur Gewährleistung aller Entkopplungseigenschaften wird kaum möglich sein, da die Anzahl der verfügbaren Nachrichtentypen mit der Zeit das System überlasten würden. Eine Änderung oder Erweiterung des Kommunikationsparadigmas in Richtung von Tuple-Spaces wird hier nötig sein.

7.2.4. Integration bestehender Anwendungen

Ein weiterer Punkt, der in dieser Arbeit ausgespart wurde, ist die mögliche Integration bestehender Anwendungen in ein entsprechend entkoppeltes System. Viele solcher Anwendungen basieren auf einer eng gekoppelten, meist TCP/IP-basierenden Client-Server- oder Peer-to-Peer-Kommunikation.

Entsprechende Produzenten veröffentliche ihre Daten oder ihre Verfügbarkeit auf zentralen Servern, von denen Konsumenten dann die Daten empfangen oder von der Verfügbarkeit von Produzenten erfahren und direkten Kontakt mit diesen suchen. Hier könnte die Middleware erweitert werden, einen entsprechenden zentralen Server zu simulieren, aber die eigentliche Kommunikation zwischen Produzenten und Konsumenten dann entkoppelt realisieren.

Insbesondere für die Integration bestehender Anwendungen kann es nötig sein, die Grenzen der unidirektionalen Vermittlung von Nachrichten aufzubrechen. In einem ersten Schritt könnte ein Rückkanal vom Konsumenten zum Produzenten sinnvoll sein, in dem Signalisierung von Erfolg oder Misserfolg kommuniziert werden kann. Ein zweiter Schritt könnte die Anforderung weiterer Informationen über einen Rückkanal sein. Selbstverständlich soll dadurch nicht auf die entkoppelte Natur der Kommunikation verzichtet werden.

7.3. Schlusswort

Neben einer ausgiebigen Untersuchung der vorgestellten Problematik wurde in dieser Arbeit ein umfangreiches Konzept zur Realisierung einer inhaltlich entkoppelten Kommunikation in heterogenen, veränderlichen Umgebungen entwickelt. Es wurden zwei algorithmische Ansätze zur Lösung des Problems entwickelt und untersucht sowie deren Eignung in zwei umfangreiche Simulationsstudien gezeigt.

Bei der Konzeption der Algorithmen wurden umfangreiche Optimierungsvorschläge unterbreitet, aber für die Simulation stets der am wenigsten optimale Weg der Implementierung

gewählt. Somit stellen die Ergebnisse der Untersuchung viel mehr einen Proof-of-Concept als eine Beurteilung des maximal möglichen dar.

Trotzdem sollen einige Schwächen der Arbeit nicht unerwähnt bleiben. Eine vollständige formale Untersuchung der entwickelten Algorithmen ist im Rahmen einer Arbeit dieses Umfangs ebenso wenig möglich, wie eine umfassende Untersuchung deren Verhaltens in allen denkbaren Szenarien. Es wurden für die untersuchten Eigenschaften sowohl geeignete, als auch anspruchsvolle Szenarien ausgewählt.

Eine Pauschalisierung oder Verallgemeinerung der Ergebnisse auf alle möglichen Anforderungen der Realität ist nicht möglich. Zum Beispiel wurden auch aufgrund der Einschränkungen der Simulationsumgebung keine wirklich komplexen Szenarien simuliert. Auch der Bereich der zustandsbehafteten Datenverarbeitung wurde bei der simulativen Untersuchung der Ansätze ausgespart.

Eine Implementierung einer frühen Version des Ein-Wege-Ansatzes auf der Basis der vorgestellten Konzepte fand im Rahmen einer begleitenden Arbeit (SEUCHTER, 2009) statt. Es konnte in einem einfachen Szenario gezeigt werden, dass die Konzepte in der Realität grundsätzlich funktionieren. Die aus der Arbeit gewonnenen Erkenntnisse sind an verschiedenen Stellen in die, in dieser Arbeit weiterentwickelten Algorithmen eingeflossen.

Symbolverzeichnis

Notation	Beschreibung	Seiten
	Grundlagen	
$G = (E, K, w)$	attributierter, symmetrischer, zusammenhängender, schleifenfreier, gerichteter Graph zur Abbildung einer Netzwerktopologie	18
$e \in E$	Ecke des Graphen - entspricht Kommunikationsteilnehmer bzw. Gerät	18, 20
$\|E\|$	Anzahl der Ecken im Graph - Anzahl der Geräte	18–21
$K \subset E \times E$	Kanten des Graphen - (e, f) und (f, e) bilden eine bidirektionale Kommunikationsverbindung zwischen $e \in E$ und $f \in E$	18
$\|K\|$	Anzahl der Kanten im Graph	18, 19
$w(k)$	Kantengewicht der Kante $k \in K$ - bildet die Kommunikationskosten zwischen e und f mit $k = (e, f)$ ab	18, 20
n	Die Anzahl der Kommunikationsverbindungen im Netzwerk, dass durch G repräsentiert wird	18, 19
N_e	Menge der benachbarten Ecken zu $e \in E$	18
$\|N_e\|$	Anzahl der benachbarten Ecken zu $e \in E$	20
t	Anzahl der insgesamt versandten Nachrichtenkopien bis der untersuchte Algorithmus terminiert	19–21
$q \in E$	Quelle - Ecke an der die Nachricht eingespeist wird	20
$t(q)$	Anzahl der insgesamt versandten Nachrichtenkopien bis der untersuchte Algorithmus terminiert abhängig von der Quelle $q \in E$, an der die Nachricht ursprünglich eingespeist wurde	20, 21
$\|P(q, e)\|$	Anzahl der möglichen Pfade zwischen $q \in E$ und $e \in E$	20
e	Eulersche Zahl	21

Notation	Beschreibung	Seiten

Notation	Beschreibung	Seiten
	Vermittlungstopologie	
G	Menge der Geräte in der Netzwerktopologie	67–69, 115, 116
T	Menge der Nachrichtentypen in der Verarbeitungstopologie	67–69, 78, 115, 116
E	Menge der Ecken in der Vermittlungstopologie, die als potentielle Konsumenten in Frage kommen, $E = G \times T$	67–69, 111, 112, 115, 116
B	Menge der Kanten in der Netzwerktopologie	68, 69, 80, 115, 116
V	Menge der Verarbeitungsschritte in der Verarbeitungstopologie - alle Kanten die auf einen Nachrichtentypen $t \in T$ zeigen	68, 69
K	Menge der Kanten in der Vermittlungstopologie, die zur Pfadsuche von Bedeutung sind - alle Kanten, die auf eine Ecke $e \in E$ zeigen, $K \subseteq (V \times G) \cup (G \times G)$	68, 69, 111, 112, 115, 116
	Routingmetrik	
b	Gerichtete Kante in der Netzwerktopologie, $b \in B$	80
s	Größe der zu übertragenden Nachricht	80, 81
l_b	Ausbreitungsverzögerung - Anteil der Übertragungskosten (Zeit) über die Kante b, der nicht von der Nachrichtengröße abhängt	80, 81
$e_b(s)$	Übertragungsverzögerung - Anteil der Übertragungskosten (Zeit) über die Kante b, der von der Nachrichtengröße s abhängt	80, 81
a_b	Durchschnittliche Anzahl notwendiger Übertragungen für eine erfolgreiche Übertragung eines Paketes über die Kante b	80, 81
w_b	Gewichtung der Kosten für die Kante b	81
m_b	Technologiebedingte maximal übertragbare Paketgröße auf der Kante b	81

Notation	Beschreibung	Seiten

Notation	Beschreibung	Seiten
$c_b(s)$	Gewichtete, größenabhängige zu erwartende Übertragungszeit auf der Kante b	81
ι	Kante in der Vermittlungstopologie, die eine Instanz eines Verarbeiters repräsentiert, $i \in I \vert I \subseteq V \times G$	81
l_ι	Fixer Anteil der Verarbeitungsverzögerung auf der Kante ι	81, 82
$e_\iota(s)$	Größenabhängiger Anteil der Verarbeitungsverzögerung auf der Kante ι	81, 82
w_ι	Gewichtung der Kante ι	81, 82
$c_\iota(s)$	Gewichtete, größenabhängige zu erwartende Verarbeitungszeit für die Kante ι	82
s_0	Die vom Produzenten mitgeteilte zu erwartende Größe einer Nachricht	83
f_ι	Größenveränderungsfaktor - beschreibt die zu erwartende Größenveränderung einer oder mehrerer Nachrichten durch die Verarbeitung auf der Kante ι	83
s_ι	Die nach der Verarbeitung auf der Kante ι zu erwartende Größe einer Nachricht	83

Algorithmen

aid	Die AID einer Ankündigung - repräsentiert deren Nachrichtentyp	95, 113
val	Der Gültigkeitszeitraum einer Ankündigung.	95, 97
seq	Die Sequenznummer einer Ankündigung	95, 97, 98
size	Schätzwert für die Größe der mit einer Ankündigung assoziierten Nachrichten	95, 106
cost	Pfadkosten des Pfades, den die Ankündigung bereits zurückgelegt hat	95, 96, 101, 104, 106
Ank_i	eine von einer Ecke empfangene Ankündigung - i repräsentiert die Anzahl der bereits empfangenen identischen Ankündigungen	101, 104, 107
Ank_0	die erste von einer Ecke empfange Ankündigung	101, 104–106, 108

Notation	Beschreibung	Seiten

Notation	Beschreibung	Seiten
$absd'$	der kostengünstigste Absender beim Ein-Wege-Ansatz	101, 102, 113
$cost'$	die geringsten empfangenen Pfadkosten beim Ein-Wege-Ansatz	101, 113
sid	die SID eines Abonnements	102, 105, 106, 108, 113
next	die nächste auf einem Pfad liegende Ecke	102, 113
last	die vorhergehende auf einem Pfad liegende Ecke	102, 113
$absd_i$	der von einer Ecke gespeicherte Absender der von ihr empfangenen Ankündigung A_i	104, 105, 107, 108, 113
$cost_i$	die von einer Ecke gespeicherten Pfadkosten aus der von ihr empfangenen Ankündigung A_i	104, 105, 107, 113
$cost_0$	die von einer Ecke gespeicherten Pfadkosten aus der von ihr empfangenen Ankündigung A_0	105, 106
mincost	die Pfadkosten, die nicht erreicht werden dürfen, damit eine Ankündigung von einer Ecke akzeptiert wird	104–108, 113
Abo_e	Ein von der Ecke *e* empfangenes Abonnement	105, 106
kost	Im Abonnement gespeicherter Pfadkostenwert	105, 106
e_j	Ecke, von der ein Abonnement erhalten wurde	106
$pathcost_e$	lokale Pfadkosten von einer Ecke bis zur Senke für den günstigsten Pfad auf dem die Ecke *e* der nächste Schritt ist	105, 106, 108, 113

Komplexität

s	Größe einer Nachricht, die in einer Phase übermittelt werden muss	111
s_{Ank}	Größe einer Ankündigung	111, 112
s_{Abo}	Größe eines Abonnements	111, 112
s_N	Größe einer Nachricht	111, 112
h	Häufigkeit der Initiierung einer Nachrichtenübermittlung für eine Phase	111

Notation	Beschreibung	Seiten

Notation	Beschreibung	Seiten
h_{Ank}	Häufigkeit der Initiierung einer Nachrichtenübermittlung für die Phase der Ankündigung	111, 112
h_{Abo}	Häufigkeit der Initiierung einer Nachrichtenübermittlung für die Phase des Abonnements	111, 112
h_N	Häufigkeit der Initiierung einer Nachrichtenübermittlung für die Phase des Nachrichtenversands	111, 112
a	Anzahl der für eine vollständige Nachrichtenübermittlung zu übertragenen Kopien der für die jeweilige Phase nötigen Nachricht	111
a_{Ank}	Anzahl der in der Ankündigungsphase zu übertragenen Kopien der Ankündigung	111, 112, 115, 116
a_{Abo}	Anzahl der in der Abonnementphase zu übertragenen Kopien von Abonnements	111, 112
a_N	Anzahl der in der Vermittlungsphase zu übertragenen Nachrichtenkopien	111, 112
t_g	Gültigkeit einer Ankündigung	112
t_p	Pfadnutzungsdauer - Zeit zwischen Abonnement und Pfadbruch	112, 114
t_N	Zeit zwischen der Veröffentlichung zweier Nachrichten	111, 112
q	Anzahl der angemeldeten Quellen	111, 112, 115, 116
v	Anzahl der Instanzen installierter Verarbeiter	111, 112, 115, 116
$\sum_{j \in SID}$	Summe über alle Senken	112
$\sum_{p \in P_j}$	Summe über alle verfügbaren Pfade zur Senke j	112
$\lvert p_j \rvert$	Pfadlänge des kürzesten Pfades zur Senke j	112, 114
$\lvert p \rvert$	Pfadlänge des Pfades p	112
x	Faktor, durch den sich die Anzahl der notwendigen Übertragung durch das Versenden einer Nachricht auf einer Verbindung in Richtung zweier Senken verringert: $0 < x \leq 1$	112

Notation	Beschreibung	Seiten

Literaturverzeichnis

EMILE AARTS, RICK HARWIG UND MARTIN SCHUURMANS; PETER J. DENNING (HRSG.), Kap. Ambient intelligence In The invisible future: the seamless integration of technology into everyday life. New York, NY, USA: McGraw-Hill, Inc., 2002, 235–250.

DANIEL J ABADI, YANIF AHMAD, MAGDALENA BALAZINSKA, UGUR CETINTEMEL, MITCH CHERNIACK, JEONG-HYON HWANG, WOLFGANG LINDNER, ANURAG S MASKEY, ALEXANDER RASIN, ESTHER RYVKINA, NESIME TATBUL, YING XING UND STAN ZDONIK, The Design of the Borealis Stream Processing Engine. in: *Second Biennial Conference on Innovative Data Systems Research (CIDR)*. Asilomar, CA, Januar 2005.

YANIF AHMAD, UGUR ÇETINTEMEL, JOHN JANNOTTI, ALEXANDER ZGOLINSKI UND STANLEY B. ZDONIK, Network Awareness in Internet-Scale Stream Processing. *IEEE Data Eng. Bull.* 28 2005:1, 63–69.

ERWIN AITENBICHLER, Development Tools for Mundo Smart Environments. in: GERD KORTUEM (HRSG.): *Workshop on Software Engineering Challenges for Ubiquitous Computing*. Lancaster University, Juni 2006, 89–90.

ERWIN AITENBICHLER, JUSSI KANGASHARJU UND MAX MÜHLHAUSER, MundoCore: A light-weight infrastructure for pervasive computing. *Pervasive and Mobile Computing*, 3 August 2007:4, 332–361.

LACHLAN ALDRED, WIL M. P. VAN DER AALST, MARLON DUMAS UND ARTHUR H. M. TER HOFSTEDE, Dimensions of coupling in middleware. *Concurrency and Computation: Practice and Experience*, Februar 2009, 2260–2269.

LACHLAN ALDRED, WIL M.P. VAN DER AALST, MARLON DUMAS UND ARTHUR H.M. TER HOFSTEDE, On the Notion of Coupling in Communication Middleware. in: *On the Move to Meaningful Internet Systems: CoopIS, DOA, and ODBASE*. Band 3761/2005, Springer, 2005, 1015–1033.

MEHMET ALTINEL UND MICHAEL J. FRANKLIN, Efficient Filtering of XML Documents for Selective Dissemination of Information. in: *Proceedings of the 26th International*

Conference on Very Large Data Bases (VLDB). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, 53–64.

BARUCH AWERBUCH UND ROBERT G. GALLAGER, A new distributed algorithm to find breadth first search trees. *IEEE Trans. Inf. Theor.* 33 1987:3, 315–322.

ROBERTO BALDONI, ROBERTO BERALDI, GIANPAOLO CUGOLA, MATTEO MIGLIAVACCA UND LEONARDO QUERZONI, Structure-less content-based routing in mobile ad hoc networks. in: *Proceedings of the International Conference on Pervasive Services (ICPS)*. Santorini, Greece, Juli 2005, 37–46.

GURUDUTH BANAVAR, MARC KAPLAN, ROBERT E. STROM, DANIEL C. STURMAN, KELLY SHAW UND WEI TAO, Information Flow based Event Distribution Middleware. *International Conference on Distributed Computing Systems*, 1999, 114–121.

JOHN BATES, JEAN BACON, KEN MOODY UND MARK SPITERI, Using events for the scalable federation of heterogeneous components. in: *EW 8: Proceedings of the 8th ACM SIGOPS European workshop on Support for composing distributed applications*. Sintra, Portugal: ACM, 1998, 58–65.

M.E.M. CAMPISTA, P.M. ESPOSITO, I.M. MORAES, L.H.M. COSTA, O.C.M. DUARTE, D.G. PASSOS, C.V.N. DE ALBUQUERQUE, D.C.M. SAADE UND M.G. RUBINSTEIN, Routing Metrics and Protocols for Wireless Mesh Networks. *Network, IEEE*, 22 Januar 2008:1, 6–12.

ANTONIO CARZANIGA, DAVID S. ROSENBLUM UND ALEXANDER L. WOLF, Achieving scalability and expressiveness in an Internet-scale event notification service. in: *PODC '00: Proceedings of the nineteenth annual ACM symposium on Principles of distributed computing*. Portland, Oregon, United States: ACM, 2000, 219–227.

ANTONIO CARZANIGA, DAVID S. ROSENBLUM UND ALEXANDER L. WOLF, Design and evaluation of a wide-area event notification service. *ACM Trans. Comput. Syst.* 19 2001:3, 332–383.

MIGUEL CASTRO, PETER DRUSCHEL, ANNE-MARIE KERMARREC UND ANTONY ROWSTRON, SCRIBE: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communications*, 20 2002, 100–110.

SHARMA CHAKRAVARTHY, V. KRISHNAPRASAD, EMAN ANWAR UND S.-K. KIM, Composite Events for Active Databases: Semantics, Contexts and Detection. in: *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994, 606–617.

SHARMA CHAKRAVARTHY UND DEEPAK MISHRA, Snoop: an expressive event specification language for active databases. *Data Knowl. Eng.* 14 1994:1, 1–26.

GUANLING CHEN, MING LI UND DAVID KOTZ, Data-centric middleware for context-aware pervasive computing. *Pervasive Mob. Comput.* 4 2008:2, 216–253.

LIANG CHEN, KOLAGATLA REDDY UND GAGAN AGRAWAL, GATES: A Grid-Based Middleware for Processing Distributed Data Streams. in: *Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing (HPDC)*. Washington, DC, USA: IEEE Computer Society, 2004, 192–201.

YUAN CHEN, KARSTEN SCHWAN UND DONG ZHOU, Opportunistic channels: mobility-aware event delivery. in: *Proceedings of the ACM/IFIP/USENIX International Conference on Middleware*. Rio de Janeiro, Brazil: Springer-Verlag New York, Inc., 2003, 182–201.

MITCH CHERNIACK, HARI BALAKRISHNAN, MAGDALENA BALAZINSKA, DONALD CARNEY, UGUR CETINTEMEL, YING XING UND STAN ZDONIK, Scalable Distributed Stream Processing. in: *First Biennial Conference on Innovative Data Systems Research (CIDR)*. Asilomar, CA, Januar 2003.

DAVID D. CLARK, *IP Datagram Reassembly Algorithms*. Fremont, CA, USA, November 1982 (815). – RFC.

R. COMROE UND JR. COSTELLO, D., ARQ Schemes for Data Transmission in Mobile Radio Systems. *Selected Areas in Communications, IEEE Journal on*, 2 Juli 1984:4, 472–481.

DIANE J. COOK UND SAJAL K. DAS (HRSG.), *Smart Environments - Technologies, Protocols, and Applications*. Hoboken, New Jersey: Wiley, 2005.

PAOLO COSTA UND GIAN PIETRO PICCO, Semi-Probabilistic Content-Based Publish-Subscribe. in: *ICDCS '05: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2005, 575–585.

GIANPAOLO CUGOLA, ELISABETTA DI NITTO UND ALFONSO FUGGETTA, The JEDI Event-Based Infrastructure and Its Application to the Development of the OPSS WFMS. *IEEE Trans. Softw. Eng.* 27 2001:9, 827–850.

GIANPAOLO CUGOLA UND GIAN PIETRO PICCO, REDS: a reconfigurable dispatching system. in: *Proceedings of the 6th international workshop on Software engineering and middleware (SEM)*. New York, NY, USA: ACM, 2006, 9–16.

DAVE WINER, *XML-RPC Specification*. ⟨URL: `http://www.xmlrpc.com/spec`⟩ – Zugriff am 3. September 2010.

U. DAYAL, B. BLAUSTEIN, A. BUCHMANN, U. CHAKRAVARTHY, M. HSU, R. LEDIN, D. MCCARTHY, A. ROSENTHAL, S. SARIN, M. J. CAREY, M. LIVNY UND R. JAUHARI, The HiPAC project: combining active databases and timing constraints. *SIGMOD Rec.* 17 1988:1, 51–70.

DOUGLAS S. J. DE COUTO, DANIEL AGUAYO, JOHN BICKET UND ROBERT MORRIS, A high-throughput path metric for multi-hop wireless routing. in: *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*. San Diego, CA, USA: ACM, 2003, 134–146.

ALAN DEMERS, DAN GREENE, CARL HAUSER, WES IRISH, JOHN LARSON, SCOTT SHENKER, HOWARD STURGIS, DAN SWINEHART UND DOUG TERRY, Epidemic algorithms for replicated database maintenance. in: *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing (PODC)*. Vancouver, British Columbia, Canada: ACM, 1987, 1–12.

ALAN J. DEMERS, JOHANNES GEHRKE, MINGSHENG HONG, MIREK RIEDEWALD UND WALKER M. WHITE, Towards Expressive Publish/Subscribe Systems. in: YANNIS E. IOANNIDIS, MARC H. SCHOLL, JOACHIM W. SCHMIDT, FLORIAN MATTHES, MICHAEL HATZOPOULOS, KLEMENS BÖHM, ALFONS KEMPER, TORSTEN GRUST UND CHRISTIAN BÖHM (HRSG.): *10th International Conference on Extending Database Technology (EDBT) - Advances in Database Technology*. Band 3896, Munich, Germany: Springer, März 2006, 627–644.

RICHARD DRAVES, JITENDRA PADHYE UND BRIAN ZILL, Routing in multi-radio, multi-hop wireless mesh networks. in: *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*. Philadelphia, PA, USA: ACM, 2004, 114–128.

GREG EISENHAUER, FABIÁN E. BUSTAMANTE UND KARSTEN SCHWAN, A middleware toolkit for client-initiated service specialization. *SIGOPS Oper. Syst. Rev.* 35 2001:2, 7–20.

PATRICK TH. EUGSTER, PASCAL A. FELBER, RACHID GUERRAOUI UND ANNE-MARIE KERMARREC, The many faces of publish/subscribe. *ACM Comput. Surv.* 35 2003:2, 114–131.

PATRICK TH. EUGSTER UND RACHID GUERRAOUI, Content-based publish/subscribe with structural reflection. in: *Proceedings of the 6th conference on USENIX Conference*

on Object-Oriented Technologies and Systems (COOTS). San Antonio, Texas: USENIX Association, 2001, 10–10.

LUDGER FIEGE, FELIX C. GÄRTNER, OLIVER KASTEN UND ANDREAS ZEIDLER, Supporting mobility in content-based publish/subscribe middleware. in: *Proceedings of the ACM/IFIP/USENIX International Conference on Middleware*. Rio de Janeiro, Brazil: Springer-Verlag New York, Inc., 2003, 103–122.

ERIC FREEMAN, SUSANNE HUPFER UND KEN ARNOLD, *JavaSpaces principles, patterns, and practice*. Addison-Wesley, 1999.

ERICH GAMMA, RICHARD HELM, RALPH E. JOHNSON UND JOHN VLISSIDES, *Design Patterns - Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

STELLA GATZIU UND KLAUS R. DITTRICH, Detecting Composite Events in Active Database Systems Using Petri Nets. in: J. WIDOM UND S. CHAKRAVARTHY (HRSG.): *Fourth International Workshop on Research Issues in Data Engineering: Active Database Systems*. Houston, Texas: IEEE Computer Society, Februar 1994, 2–9.

DAVID GELERNTER, Generative communication in Linda. *ACM Trans. Program. Lang. Syst.* 7 1985:1, 80–112.

DINA GOLDIN, MINGJUN SONG, AYFERI KUTLU, HUAYAN GAO UND HARDIK DAVE; ANTHONY STEFANIDIS UND SILVIA NITTEL (HRSG.), Kap. Georouting and Delta-Gathering: Efficient Data Propagation Techniques for GeoSensor Networks In Geosensor Networks. CRC Press, 2004, 73–96.

ZYGMUNT J. HAAS, JOSEPH Y. HALPERN UND LI LI, Gossip-based ad hoc routing. *IEEE/ACM Trans. Netw.* 14 2006:3, 479–491.

UWE HANSMANN, LOTHAR MERK, MARTIN S. NICKLOUS UND THOMAS STOBER, *Pervasive Computing: The Mobile World*. Springer, 2003, Springer Professional Computing.

JOHN HEIDEMANN, FABIO SILVA, CHALERMEK INTANAGONWIWAT, RAMESH GOVINDAN, DEBORAH ESTRIN UND DEEPAK GANESAN, Building efficient wireless sensor networks with low-level naming. in: *Proceedings of the eighteenth ACM symposium on Operating systems principles (SOSP)*. Banff, Alberta, Canada: ACM, 2001, 146–159.

THOMAS HEIDER, *Goal-based Interaction with Smart Environments: A Unified Distributed System Architecture*. Dissertation, Universität Rostock, 2009.

WENDI RABINER HEINZELMAN, ANANTHA CHANDRAKASAN UND HARI BALAKRISHNAN, Energy-Efficient Communication Protocol for Wireless Microsensor Networks. in: *Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS)*. Band 8, Washington, DC, USA: IEEE Computer Society, 2000, 8020.

WENDI RABINER HEINZELMAN, JOANNA KULIK UND HARI BALAKRISHNAN, Adaptive protocols for information dissemination in wireless sensor networks. in: *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking (MobiCom)*. Seattle, Washington, United States: ACM, 1999, 174–185.

MICHAEL HELLENSCHMIDT, Distributed implementation of a self-organizing appliance middleware. in: *Proceedings of the joint conference on Smart objects and ambient intelligence (sOc-EUSAI)*. Grenoble, France: ACM, 2005, 201–206.

MICHAEL HELLENSCHMIDT UND THOMAS KIRSTE, Self-organization for multi-component multi-media environments. in: *Proceedings of the UniComp Workshop on Ubiquitous Display Environments*. Nottingham, England, September 2004a.

MICHAEL HELLENSCHMIDT UND THOMAS KIRSTE, SodaPop: A Software Infrastructure Supporting Self-Organization in Intelligent Environments. in: *Proceedings of the 2nd IEEE Conference on Industrial Informatics (INDIN)*. Berlin, Germany, Juni 2004b.

YONGQIANG HUANG UND HECTOR GARCIA-MOLINA, Publish/subscribe in a mobile environment. *Wireless Networks*, 10 2004:6, 643–652.

RYAN HUEBSCH, JOSEPH M. HELLERSTEIN, NICK LANHAM, BOON THAU LOO, SCOTT SHENKER UND ION STOICA, Querying the internet with PIER. in: *Proceedings of the 29th international conference on Very large data bases (VLDB)*. VLDB Endowment, 2003, 321–332.

IBM, *IBM developerWorks : WebSphere MQ*. ⟨URL: `http://www.ibm.com/developerworks/websphere/zones/businessintegration/wmq.html`⟩ – Zugriff am 3. September 2010.

IBM, *TSpaces - Computer Science Research at Almaden*. ⟨URL: `http://www.almaden.ibm.com/cs/TSpaces/Version3/`⟩ – Zugriff am 3. September 2010.

IEEE-SA STANDARDS BOARD, *IEEE Std 802.2, 1998 Edition (R2003)*. New York, NY, USA, Mai 1998 – Technischer Bericht.

IEEE-SA STANDARDS BOARD, *IEEE Std 802.11-2007*. New York, NY, USA, Juni 2007 – Technischer Bericht.

IEEE-SA STANDARDS BOARD, *IEEE Std 802.3-2008 (Revision of IEEE Std 802.3-2005)*. New York, NY, USA, Dezember 2008 – Technischer Bericht.

CHALERMEK INTANAGONWIWAT, RAMESH GOVINDAN UND DEBORAH ESTRIN, Directed diffusion: a scalable and robust communication paradigm for sensor networks. in: *Proceedings of the 6th annual international conference on Mobile computing and networking (MobiCom)*. Boston, Massachusetts, United States: ACM, 2000, 56–67.

ITU, *ITU-T Recommendation G.114 – One-way transmission time*. Genf, Schweiz, März 2003 – Technischer Bericht.

NANYAN JIANG, ANDRES QUIROZ, CRISTINA SCHMIDT UND MANISH PARASHAR, Meteor: a middleware infrastructure for content-based decoupled interactions in pervasive grid environments. *Concurr. Comput. : Pract. Exper.* 20 2008:12, 1455–1484.

NANYAN JIANG, CRISTINA SCHMIDT, VINCENT MATOSSIAN UND MANISH PARASHAR, Enabling Applications in Sensor-based Pervasive Environments. in: *First International Conference on Broadband Networks (BROADNETS)*. San Jose, CA, USA, Oktober 2004.

D. JOHNSON, C. PERKINS UND J. ARKKO, *Mobility Support in IPv6*. Fremont, CA, USA, Juni 2004 (3775). – RFC.

NICOLAI M. JOSUTTIS, *SOA in Practice: The Art of Distributed System Design (Theory in Practice)*. 1. Auflage. Sabastopol, CA, USA: O'Reilly Media Inc., 2007.

THOMAS KIRSTE, DynAMITE - Dynamisch Adaptive Multimodale IT Ensembles. in: *Tagungsband: Forschungsoffensive „Software Engineering 2006", Eröffnungskonferenz*. Berlin, Germany, Juli 2004.

JON KLEINBERG UND EVA TARDOS, *Algorithm Design*. international Auflage. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2006.

OLIVER KUTTER, JENS NEUMANN UND THOMAS SCHMITZ, Extending Universal Plug And Play To Support Self-Organizing Device Ensembles. in: *International Workshop on Software Architectures for Self-Organization, and Software Techniques for Embedded and Pervasive Systems (SASO+STEPS)*. Munich, Germany: Springer, Mai 2005.

MATTHIEU LECLERCQ, VIVIEN QUÉMA UND JEAN-BERNARD STEFANI, DREAM: a component framework for the construction of resource-aware, reconfigurable MOMs. in: *ARM '04: Proceedings of the 3rd workshop on Adaptive and reflective middleware*. Toronto, Ontario, Canada: ACM, 2004, 250–255.

RENÉ MEIER UND VINNY CAHILL, STEAM: Event-Based Middleware for Wireless Ad Hoc Network. in: *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCSW)*. Washington, DC, USA: IEEE Computer Society, 2002, 639–644.

MESSAGE PASSING INTERFACE FORUM, *MPI: A Message-Passing Interface Standard - Version 2.2*. ⟨URL: `http://www.mpi-forum.org/docs/mpi-2.2/mpi22-report.pdf`⟩ – Zugriff am 3. September 2010.

MICROSOFT CORPORATION, *Distributed Component Object Model (DCOM) Remote Protocol Specification*. ⟨URL: `http://msdn.microsoft.com/library/cc201989.aspx`⟩ – Zugriff am 3. September 2010.

MICROSOFT CORPORATION, *Message Queuing (MSMQ): Message Queuing Binary Protocol Specification*. ⟨URL: `http://msdn.microsoft.com/en-us/library/cc235772(PROT.10).aspx`⟩ – Zugriff am 3. September 2010.

G. MÜHL, L. FIEGE, F. C. GÄRTNER UND A. BUCHMANN, Evaluating Advanced Routing Algorithms for Content-Based Publish/Subscribe Systems. in: *Proceedings of the 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS)*. Washington, DC, USA: IEEE Computer Society, 2002, 167.

GERO MÜHL, *Large-Scale Content-Based Publish/Subscribe Systems*. Dissertation, Darmstadt University of Technology, 2002.

GERO MÜHL, LUDGER FIEGE UND PETER PIETZUCH, *Distributed Event-Based Systems*. Springer, 2006.

OBJECT MANAGEMENT GROUP, *Common Object Request Broker Architecture (CORBA) Specification, Version 3.1*. ⟨URL: `http://www.omg.org/spec/CORBA/3.1/`⟩ – Zugriff am 15. März 2010.

BRIAN OKI, MANFRED PFLUEGL, ALEX SIEGEL UND DALE SKEEN, The Information Bus: an architecture for extensible distributed systems. in: *Proceedings of the fourteenth ACM symposium on Operating systems principles (SOSP)*. Asheville, North Carolina, United States: ACM, 1993, 58–68.

C. PARTRIDGE, T. MENDEZ UND W. MILLIKEN, *Host Anycasting Service*. Fremont, CA, USA, November 1993 (1546). – RFC.

LARRY L. PETERSON UND BRUCE S. DAVIE, *Computer Networks: A Systems Approach*. 4. Auflage. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007.

MILENKO PETROVIC, VINOD MUTHUSAMY UND HANS-ARNO JACOBSEN, Content-based routing in mobile ad hoc networks. in: *Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous)*. San Diego, CA, USA, Juli 2005, 45–55.

PETER PIETZUCH, JONATHAN LEDLIE, JEFFREY SHNEIDMAN, MEMA ROUSSOPOULOS, MATT WELSH UND MARGO SELTZER, Network-Aware Operator Placement for Stream-Processing Systems. in: *ICDE '06: Proceedings of the 22nd International Conference on Data Engineering*. Washington, DC, USA: IEEE Computer Society, 2006, 49.

PETER R. PIETZUCH, BRIAN SHAND UND JEAN BACON, A framework for event composition in distributed systems. in: *Proceedings of the ACM/IFIP/USENIX International Conference on Middleware*. Rio de Janeiro, Brazil: Springer-Verlag New York, Inc., 2003, 62–82.

J. POSTEL, *User Datagram Protocol*. Fremont, CA, USA, August 1980 (768). – RFC.

J. POSTEL, *Transmission Control Protocol*. Fremont, CA, USA, September 1981 (793). – RFC.

GREGORY J. POTTIE UND WILLIAM J. KAISER, Wireless integrated network sensors. *Commun. ACM*, 43 2000:5, 51–58.

DANIEL RONZANI, The Battle of Concepts: Ubiquitous Computing, Pervasive Computing and Ambient Intelligence in Mass Media. *Ubiquitous Computing and Communication Journal* 4 Januar 2009:2.

ANTONY ROWSTRON UND PETER DRUSCHEL, Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. in: *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*. Heidelberg, Germany, November 2001, 329–350.

BILL SEGALL, DAVID ARNOLD, JULIAN BOOT, MICHAEL HENDERSON UND TED PHELPS, Content Based Routing with Elvin4. in: *Proc. of the. Australian UNIX and Open Systems User Group Conference*. Band 2, Canberra, Australia, Juni 2000.

ROLAND SEUCHTER, *Experimentelle Evaluierung eines Ansatzes zur semantisch entkoppelten Kommunikation in dynamischen, heterogenen Netzwerken mittels Publish/Subscribe*. Diplomarbeit Universität Rostock, 2009.

THIRUNAVUKKARASU SIVAHARAN, GORDON BLAIR UND GEOFF COULSON, GREEN: A Configurable and Re-configurable Publish-Subscribe Middleware for Pervasive Computing. in: *On the Move to Meaningful Internet Systems: CoopIS, DOA, and ODBASE*. Band 3761/2005, Springer, 2005, 732–749.

UTKARSH SRIVASTAVA, KAMESH MUNAGALA UND JENNIFER WIDOM, Operator placement for in-network stream query processing. in: *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS)*. Baltimore, Maryland: ACM, 2005, 250–258.

WILLIAM STALLINGS, *Data and Computer Communications*. 8. Auflage. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2007.

R. STEWART, *Stream Control Transmission Protocol*. Fremont, CA, USA, September 2007 (4960). – RFC.

ION STOICA, ROBERT MORRIS, DAVID KARGER, M. FRANS KAASHOEK UND HARI BALAKRISHNAN, Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. in: *Proceedings of the ACM SIGCOMM Conference*. San Diego, California, August 2001.

INC SUN MICROSYSTEMS, *RPC: Remote Procedure Call Protocol Specification Version 2*. Fremont, CA, USA, Juni 1988 (1057). – RFC.

SUN MICROSYSTEMS, INC., *Java Message Service Specification*. ⟨URL: `http://java.sun.com/products/jms/`⟩ – Zugriff am 15. März 2010.

SUN MICROSYSTEMS, INC., *Java Remote Method Invocation Specification*. ⟨URL: `http://java.sun.com/javase/6/docs/platform/rmi/spec/rmiTOC.html`⟩ – Zugriff am 15. März 2010.

ANDREW S. TANENBAUM UND MAARTEN VAN STEEN, *Distributed Systems: Principles and Paradigms*. 2. Auflage. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2007.

SASU TARKOMA, *Efficient Content-based Routing, Mobility-aware Topologies, and Temporal Subspace Matching*. Dissertation, University of Helsinki, Faculty of Science, Department of Computer Science and Helsinki Institute for Information Technology, Helsinki, Finland, 2006.

WESLEY W. TERPSTRA, STEFAN BEHNEL, LUDGER FIEGE, ANDREAS ZEIDLER UND ALEJANDRO P. BUCHMANN, A peer-to-peer approach to content-based publish/subscribe. in: *DEBS '03: Proceedings of the 2nd international workshop on Distributed event-based systems*. San Diego, California: ACM, 2003, 1–8.

THE OPEN GROUP, *DCE 1.1: Remote Procedure Call*. ⟨URL: `http://www.opengroup.org/bookstore/catalog/c706.htm`⟩ – Zugriff am 3. September 2010.

ANDRAS VARGA UND RUDOLF HORNIG, *INET Framework*. ⟨URL: `http://inet.omnetpp.org/`⟩ – Zugriff am 4. Januar 2011.

ANDRÁS VARGA UND RUDOLF HORNIG, An overview of the OMNeT++ simulation environment. in: *Simutools '08: Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*. Marseille, France: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, 1–10.

MARK WEISER, Some computer science issues in ubiquitous computing. *Commun. ACM*, 36 1993:7, 75–84.

GANG XU, JIANGANG MA UND TAO HUANG, A XML-based composite event approach. in: *Proceedings of the first international workshop on Interoperability of heterogeneous information systems (IHIS)*. Bremen, Germany: ACM, 2005, 63–70.

Anhang A.

Komplexität des Flooding-Algorithmus

Es sei auf die Pfadgewichte verzichtet, da sie für die Komplexität nicht von Bedeutung sind. Daher sei ein zusammenhängender, symmetrischer, schleifenfreier, gerichteter Graph $G = (E,K)$ mit $K \subset E \times E$ gegeben. K ist irreflexiv. Als N_e seien alle Nachbarn der Ecke $e \in E$ bezeichnet. Damit ist es als $N_e := \{f \in E | (e,f) \in K \vee (f,e) \in K\}$ definiert.

Da G symmetrisch ist, existieren für jede Kommunikationsverbindung zwischen zwei Ecken e und f genau zwei Kanten: (e,f) und (f,e). Außerdem existieren für diese Kommunikationsverbindung genau zwei Elemente in den Nachbarschaftsmengen, nämlich $e \in N_f$ und $f \in N_e$. Diese jeweils zwei Einträge beziehen sich lediglich auf die eine Kommunikationsverbindung und auf keine andere. Daher muss gelten:

$$\sum_{e \in E} |N_e| = |K| \tag{A.1}$$

Das Verteilen einer einzelnen Nachricht per Flooding-Algorithmus lässt sich wie folgt beschreiben:

1. Die Quelle $q \in E$ generiert $|N_q|$ Kopien der Nachricht und sendet eine an jeden Nachbarn $n \in N_q$.
2. Jede andere Ecke $e \in E \setminus \{q\}$, die eine Kopie von einem ihrer Nachbarn $v \in N_e$ erhält, prüft, ob sie bereits vorher eine Kopie der Nachricht erhalten hat. Wenn nein, generiert sie $|N_e| - 1$ Kopien und sendet eine an jeden anderen Nachbarn $n \in N_e \setminus \{v\}$. Wenn ja, so tut sie nichts.

Da der Graph zusammenhängend ist, muss jede Ecke genau einmal eine Kopie erhalten, ohne vorher eine andere Kopie erhalten zu haben. Der Algorithmus terminiert, wenn die letzte Ecke ihre erste Kopie erhalten und daraufhin entsprechend Schritt 2 ihre Kopien generiert und versandt hat.

Gesucht ist die Anzahl t der insgesamt generierten und versandten Kopien. Von der Quelle wurden entsprechend Schritt 1 $|N_q|$ Kopien generiert. Von jeder anderen Ecke $e \in E$ wurden entsprechend Schritt 2 $|N_e| - 1$ Kopien generiert.

Die Gesamtzahl der Kopien errechnet sich somit als:

$$
\begin{aligned}
t(q) &= |N_q| + \sum_{e \in E \setminus \{q\}} (|N_e| - 1) \\
&= \sum_{e \in E} (|N_e| - 1) + 1 \\
&= \sum_{e \in E} |N_e| - |E| + 1 \qquad \text{| siehe Formel A.1} \\
&= |K| - |E| + 1.
\end{aligned}
$$

Anhang B.

Versuchsprotokolle

B.1. Validierung der Simulationsumgebung

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	200 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	0 [m/s]
Länge der Pausen	∞
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	100
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	100 [B]
Größe der Nachrichten des Verarbeiters	100 [B]
Größe der Ankündigungen des Produzenten	20 [B]
Größe der Ankündigungen des Verarbeiters	20 [B]
Parameter der Simulation	
Anzahl der Experimente	100
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]

Tabelle B.1.:
Übersicht über die Konfiguration des Testszenarios für den Versuch zur Validierung der Versuchsumgebung.

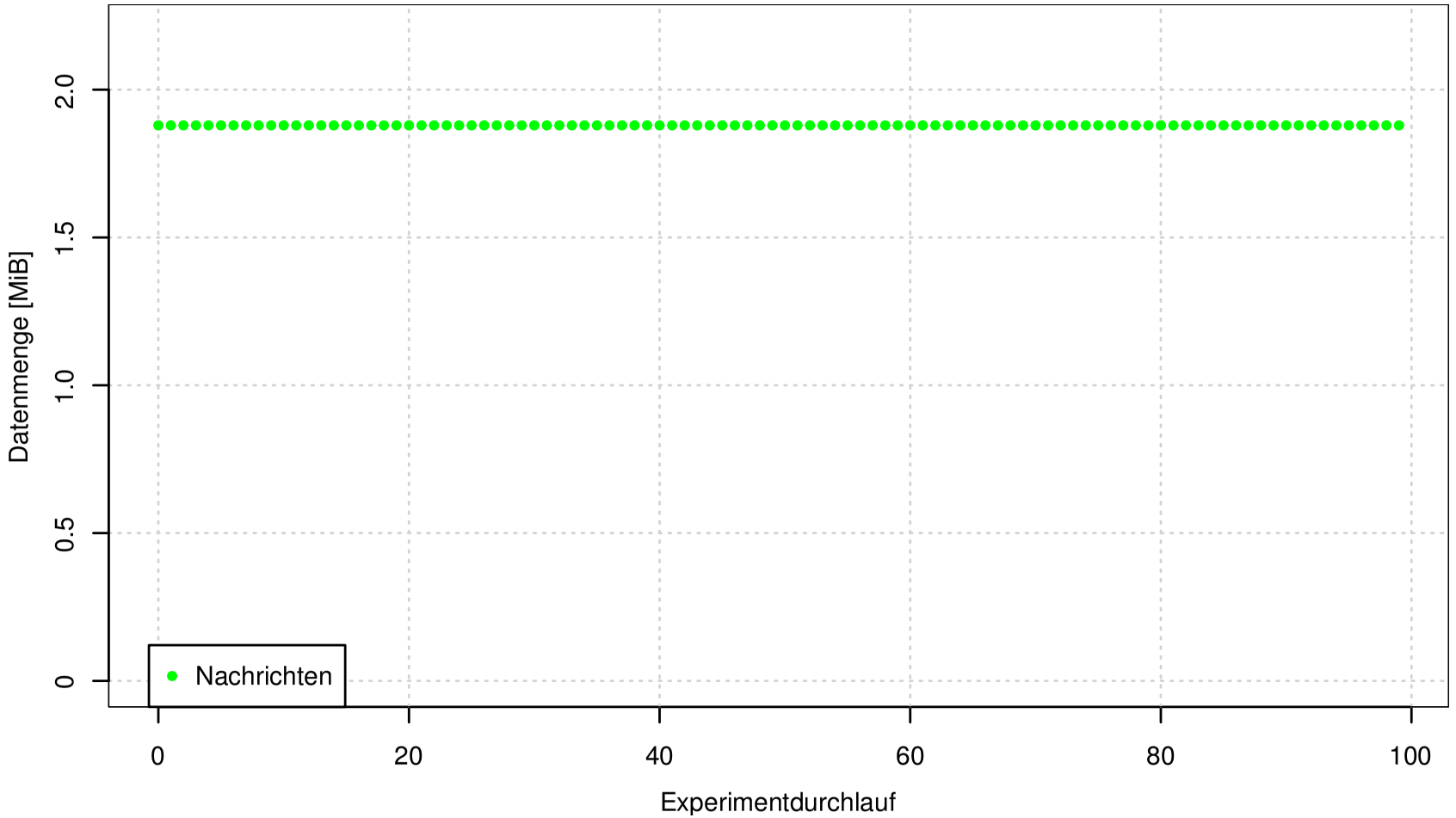


Abbildung B.1.:
Die pro Experimentaldurchlauf mit dem Flooding-Algorithmus erzeugte Netzlast entspricht immer exakt dem berechneten Wert von 1.895.400 [B] bzw. ca. 1,8 [MiB]

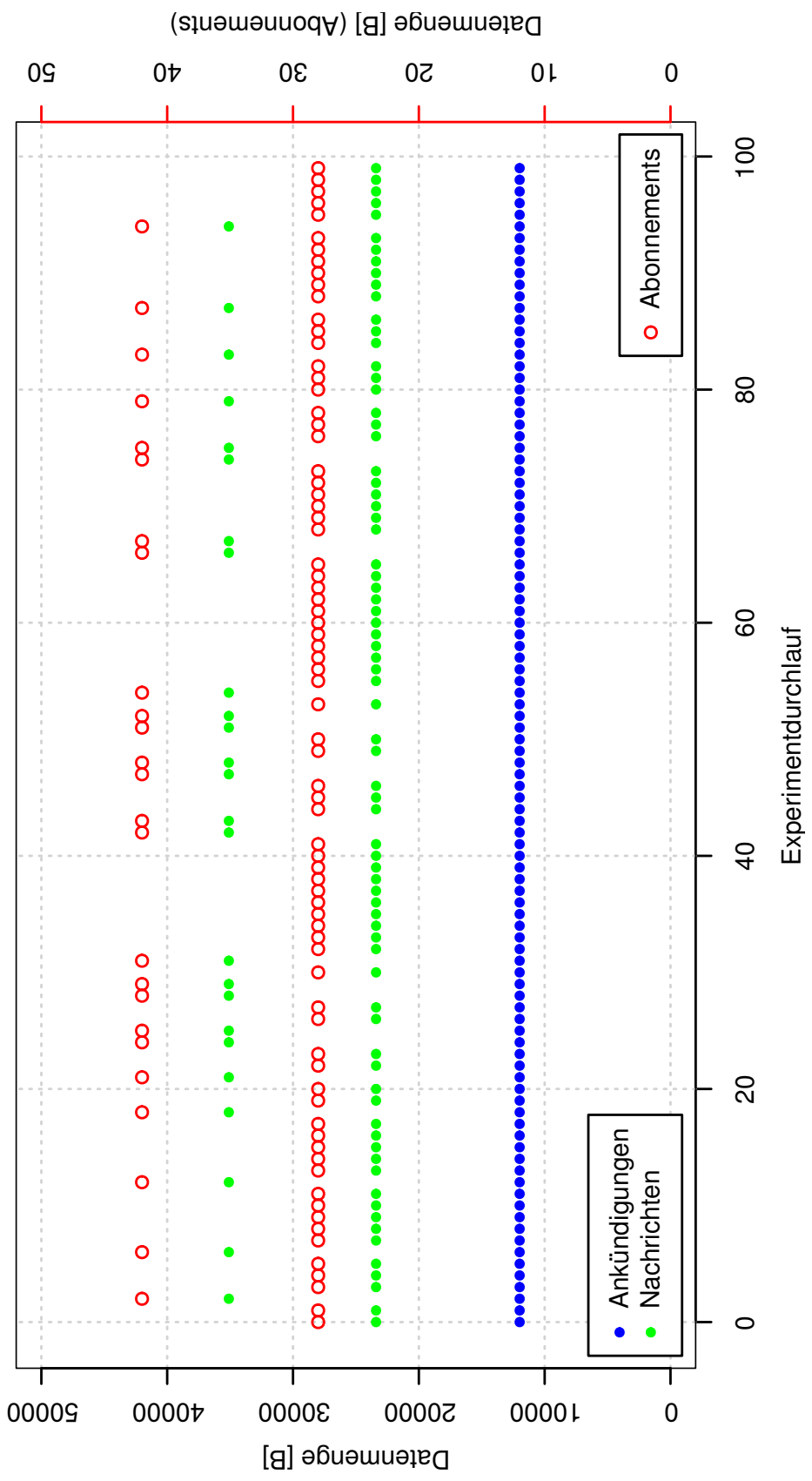


Abbildung B.2.:
Die pro Experimentaldurchlauf mit dem Ein-Wege-Ansatz erzeugte Netzlast für die Verbreitung von Ankündigungen entspricht immer exakt dem berechneten Wert von 11.988 [B] während sie sich für die Übertragung von Abonnements und Nachrichten exakt wie vorhergesagt bei 28 [B] und 23.400 [B] bzw. 42 [B] und 35100 [B] verhält. Die Netzlast für die Übertragung von Ankündigungen ist auf der rechten y-Achse aufgetragen.

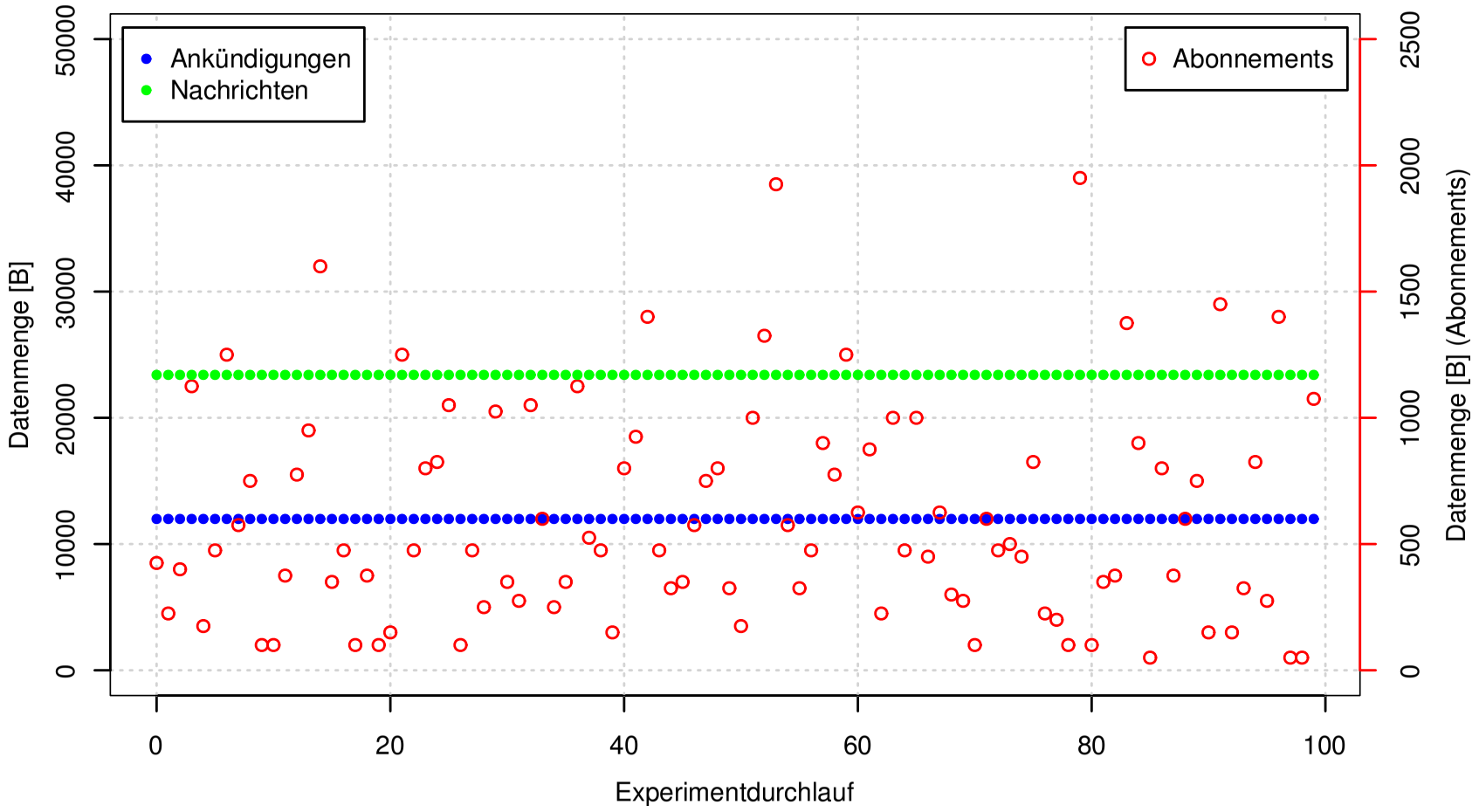


Abbildung B.3.:
Die pro Experimentaldurchlauf mit dem Mehr-Wege-Ansatz erzeugte Netzlast für die Verbreitung von Ankündigungen und Nachrichten entspricht immer exakt dem berechneten Wert während sie für die Übertragung von Abonnements wie vorhergesagt erheblich schwankt. Die Netzlast für die Übertragung von Ankündigungen ist auf der rechten y-Achse aufgetragen.

B.2. Versuch 1 – Gültigkeitsdauer der Ankündigungen

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	[1,2) [m/s]
Länge der Pausen	[10,20) [s]
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	1400 [B]
Größe der Nachrichten des Verarbeiters	1400 [B]
Größe der Ankündigungen des Produzenten	100 [B]
Größe der Ankündigungen des Verarbeiters	100 [B]
Parameter der Simulation	
Anzahl der Experimente	20
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	variabel
Anzahl der zwischengespeicherten Nachrichten	0
Anzahl der in Abonnements referenzierten Nachrichten	0

Tabelle B.2.:
Übersicht über die Konfiguration des Testszenarios für Versuch 1. Hier wird die Gültigkeitsdauer der Ankündigungen zwischen 50 [s] und 1000 [s] variiert.

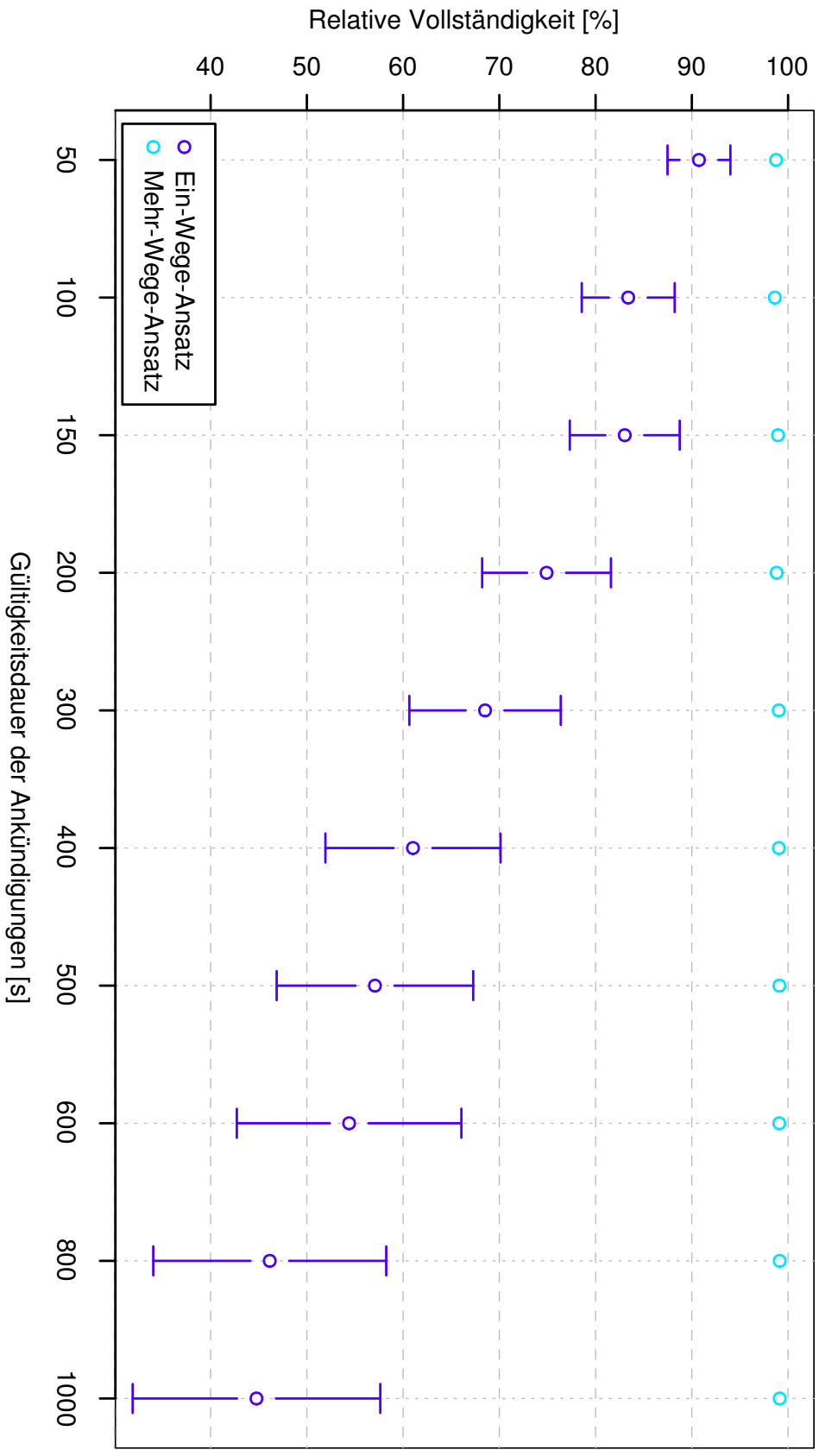


Abbildung B.4.:
Die relative Vollständigkeit als Anteil erfolgreich zugestellter Nachrichten an der Gesamtzahl veröffentlichter Nachrichten in Abhängigkeit von der Gültigkeitsdauer der Ankündigungen mit 95% Konfidenzintervallen.

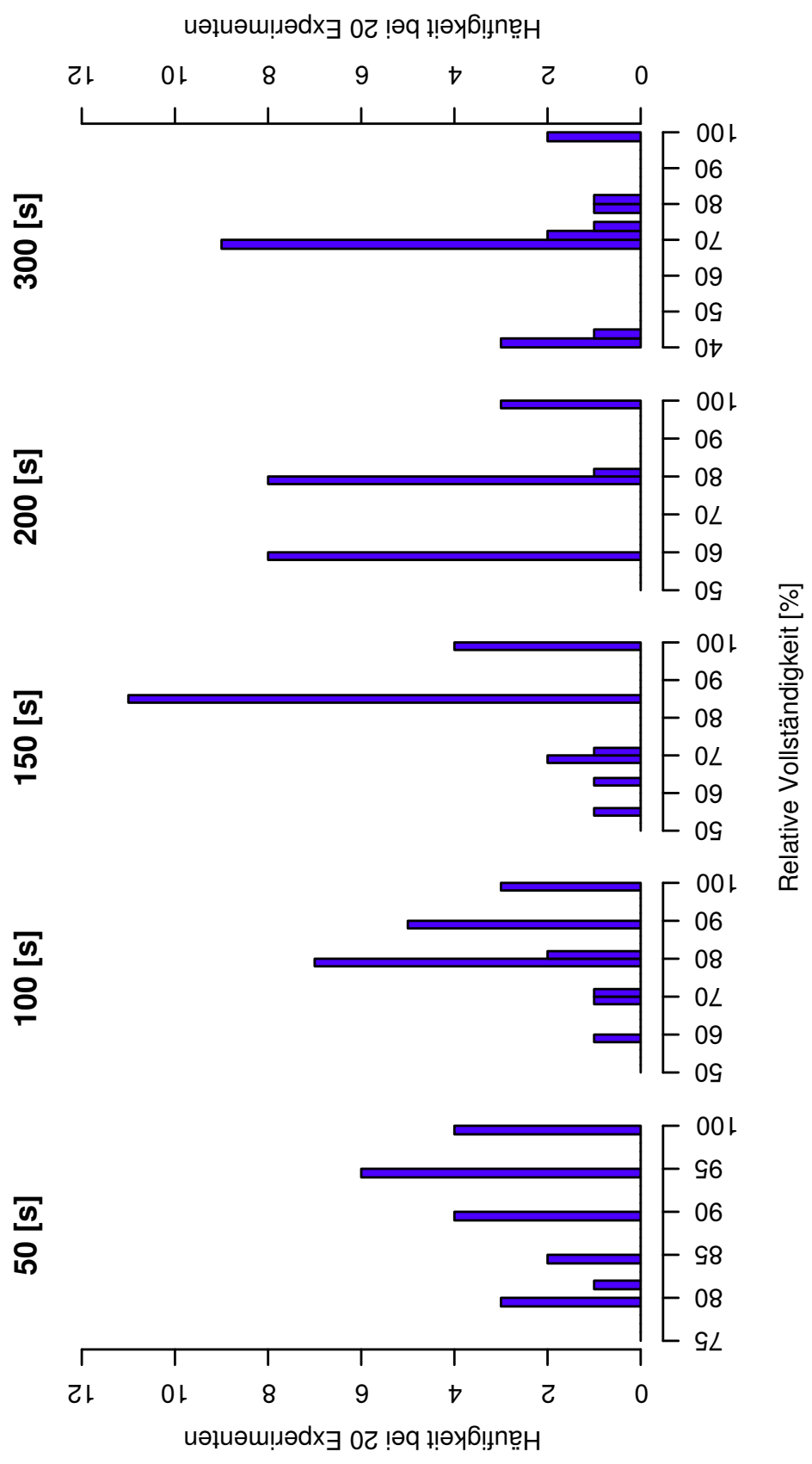


Abbildung B.5.:
Histogramm der relativen Vollständigeit für den Ein-Wege-Ansatz – Die relative Vollständigkeit wird dazu in kleine Abschnitte untergliedert und über jedem Abschnitt wird ein Balken, dessen Höhe mit der Anzahl der Experimente, deren Ergebnis in diesen Abschnitt fällt, korreliert. Die Breite der Balken entspricht der Breite der Abschnitte. Die Darstellung erfolgt für fünf ausgewählte Werte der Gültigkeitsdauer der Ankündigungen, die über dem entsprechenden Diagramm stehen.

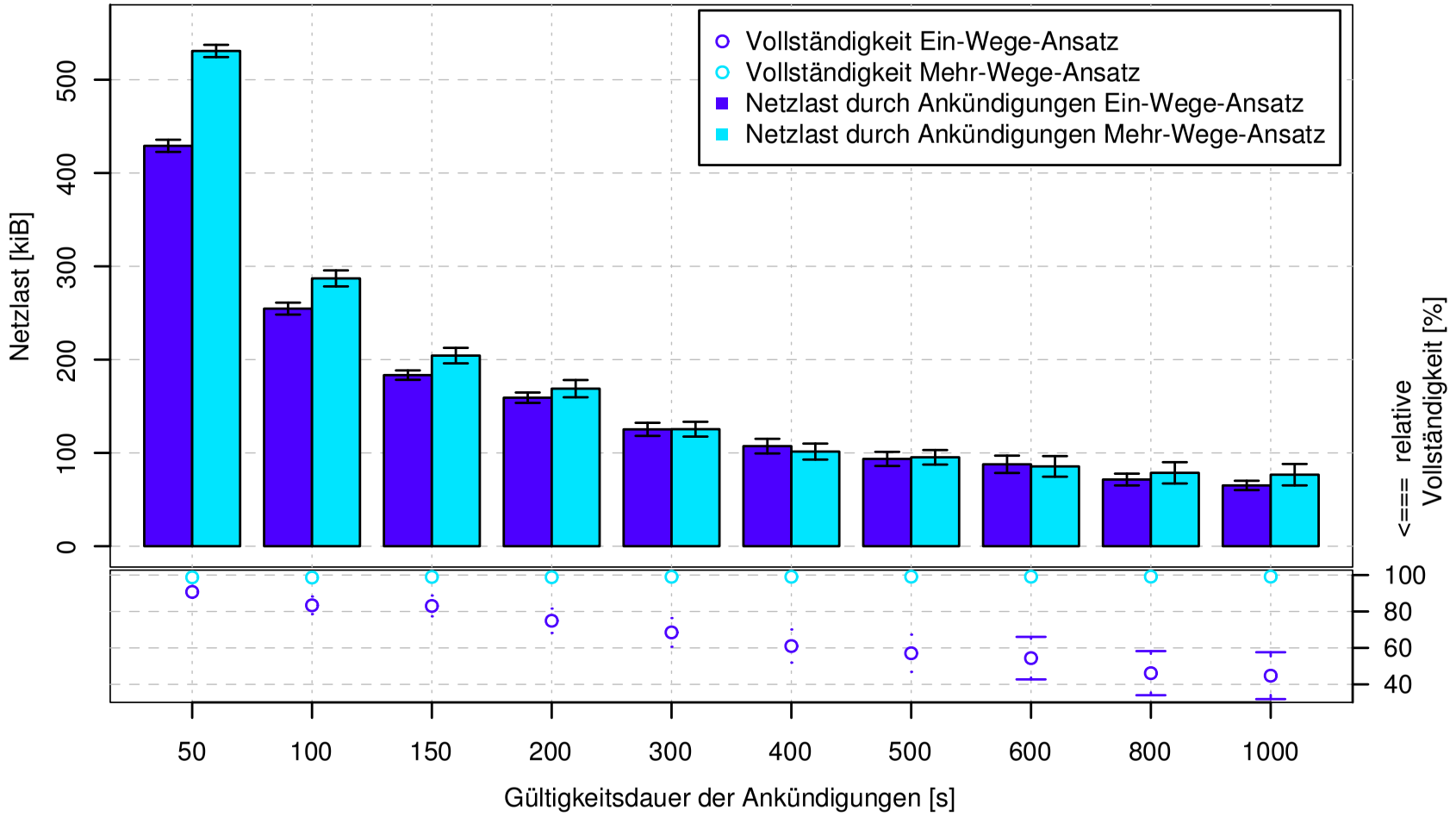


Abbildung B.6.:
Die durch den Versand von Ankündigungen erzeugte Netzlast in Abhängigkeit von der Gültigkeitsdauer der Ankündigungen mit 95% Konfidenzintervall für den Ein-Wege- und Mehr-Wege-Ansatz. Darunter ist zur Orientierung noch einmal die relative Vollständigkeit aufgetragen.

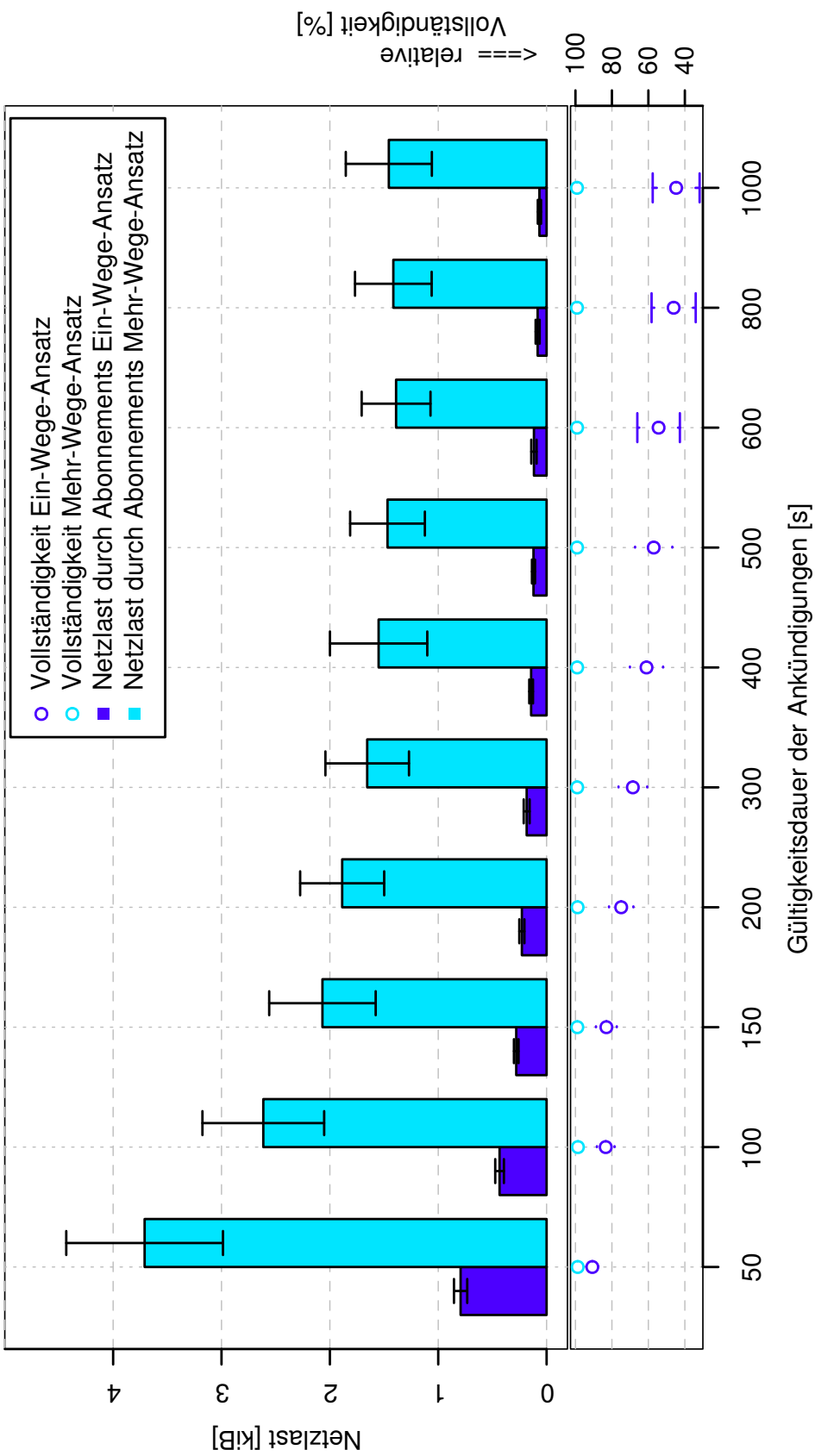


Abbildung B.7.:
Die durch den Versand von Abonnements erzeugte Netzlast in Abhängigkeit von der Gültigkeitsdauer der Ankündigungen mit 95% Konfidenzintervall für den Ein-Wege- und Mehr-Wege-Ansatz. Darunter ist zur Orientierung noch einmal die relative Vollständigkeit aufgetragen.

Abbildung B.8.:
Die durchschnittliche Pfadnutzungsdauer geteilt durch die Gültigkeitsdauer der Ankündigungen in Abhängigkeit von der Gültigkeitsdauer der Ankündigungen mit 95% Konfidenzintervallen. Im Idealfall ist dieser Wert nahe 1, so dass ein Pfad immer die ganze Zeit von Abonnement bis Ende der Gültigkeitsdauer genutzt wird. Darunter ist zur Orientierung noch einmal die durchschnittliche relative Vollständigkeit für das jeweilige Experiment aufgetragen.

B.3. Versuch 2 – Anzahl zwischengespeicherter Nachrichten

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	[1,2) [m/s]
Länge der Pausen	[10,20) [s]
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	1400 [B]
Größe der Nachrichten des Verarbeiters	1400 [B]
Größe der Ankündigungen des Produzenten	100 [B]
Größe der Ankündigungen des Verarbeiters	100 [B]
Parameter der Simulation	
Anzahl der Experimente	20
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	50 [s]
Anzahl der zwischengespeicherten Nachrichten	variabel
Anzahl der in Abonnements referenzierten Nachrichten	0

Tabelle B.3.:
Übersicht über die Konfiguration des Testszenarios für Versuch 2. Die Anzahl zwischengespeicherter Nachrichten wird zwischen 0 und 100 nichtlinear variiert.

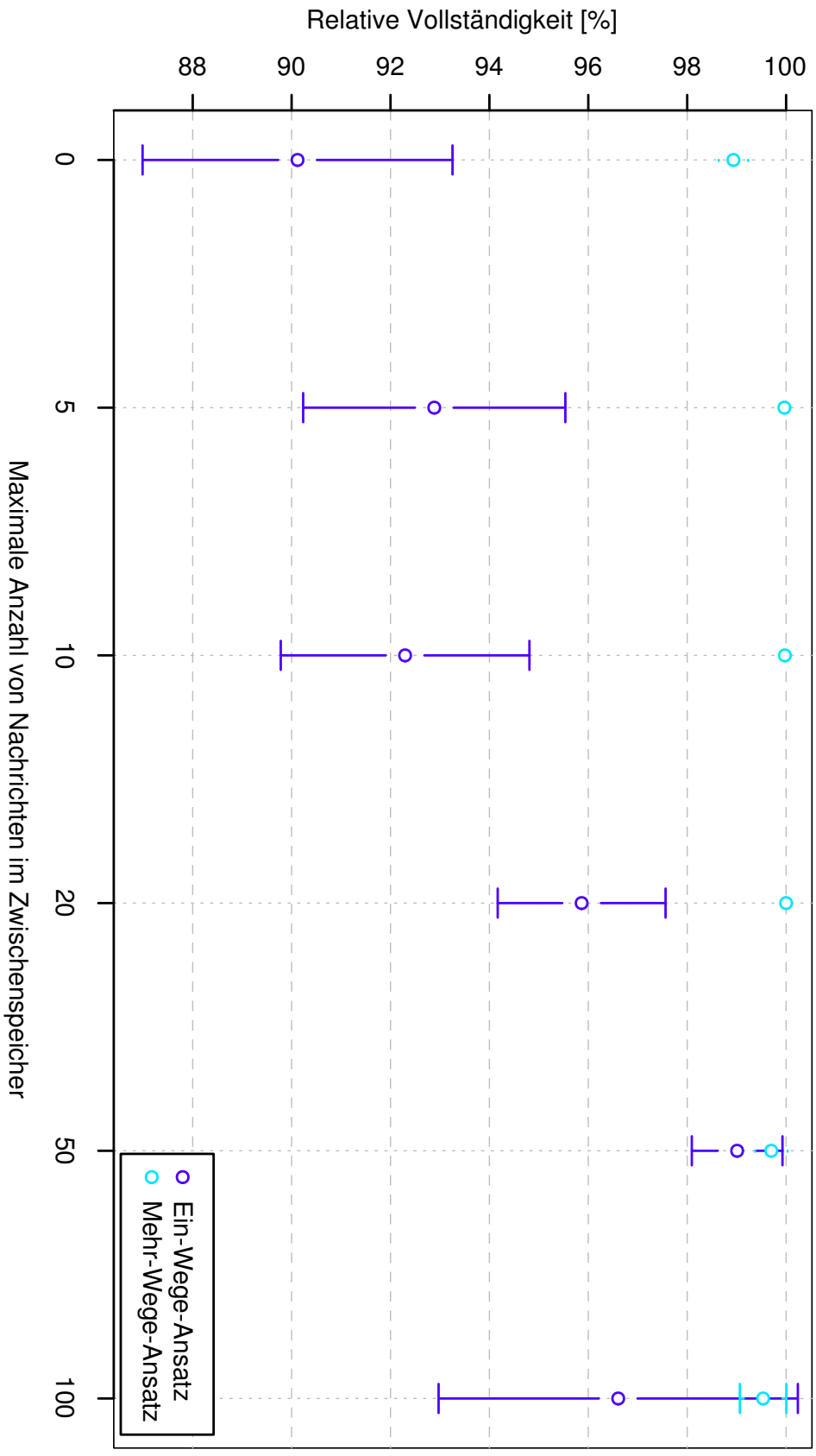


Abbildung B.9.: Durchschnittliche relative Vollständigkeit in Abhängigkeit von der Anzahl der maximal in jedem Broker zwischengespeicherten Nachrichten mit 95% Konfidenzintervallen.

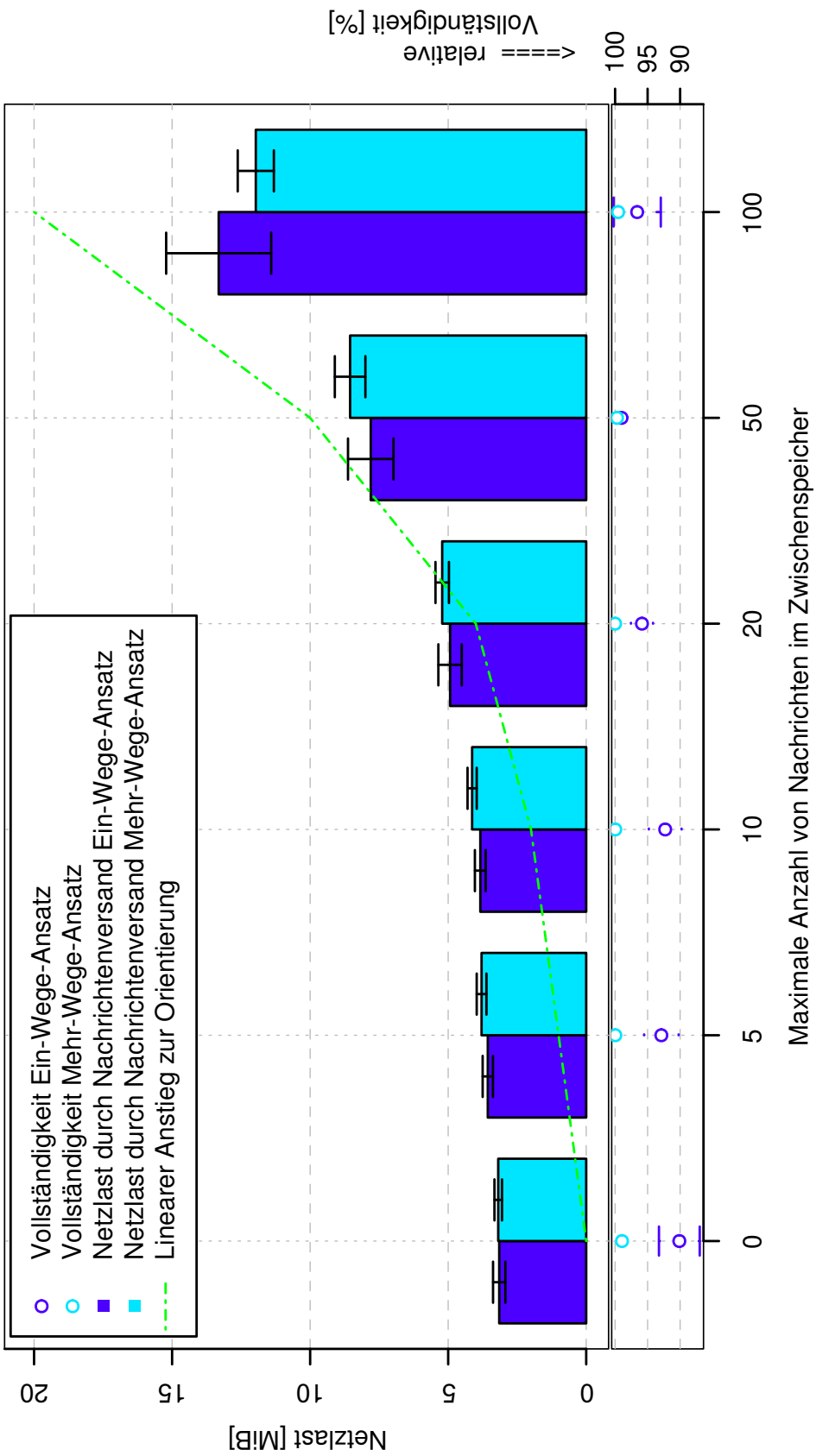


Abbildung B.10.:
Durchschnittliche Netzlast in Abhängigkeit von der Anzahl der maximal in jedem Broker zwischengespeicherten Nachrichten mit 95% Konfidenzintervallen. Da die Konfiguration des Algorithmus auf der x-Achse nicht linear ist, ist zur besseren Orientierung ein linearer Verlauf als gestrichpunktete Linie eingezeichnet. Ebenfalls zur Orientierung ist unten noch einmal die relative Vollständigkeit aufgetragen.

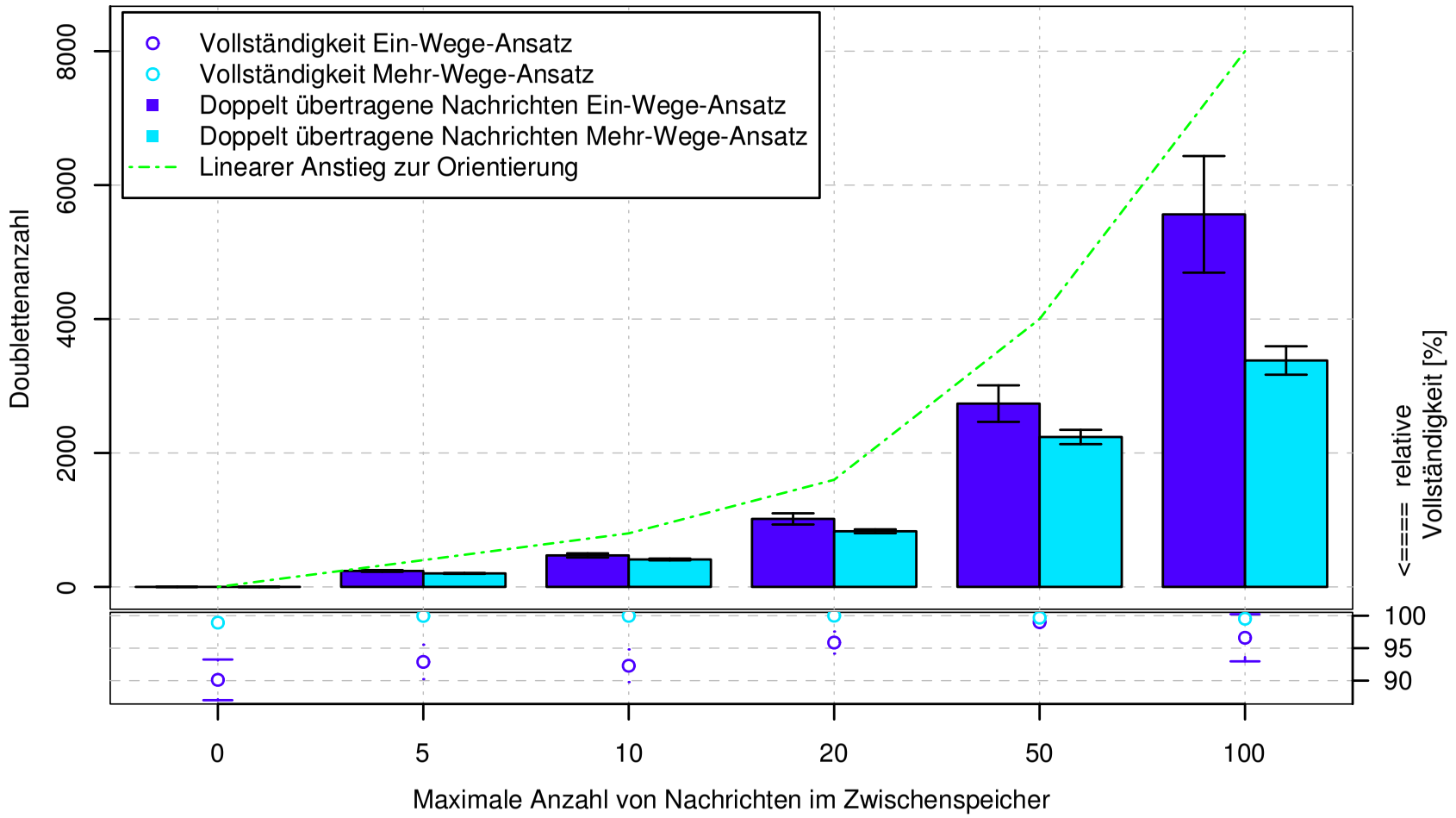


Abbildung B.11.:
Durchschnittliche Anzahl pro Experiment doppelt übertragener Nachrichten in Abhängigkeit von der Anzahl der maximal in jedem Broker zwischengespeicherten Nachrichten mit 95% Konfidenzintervallen. Da die Konfiguration des Algorithmus auf der x-Achse nicht linear ist, ist zur besseren Orientierung ein linearer Verlauf als gestrichpunktete Linie eingezeichnet. Ebenfalls zur Orientierung ist unten noch einmal die relative Vollständigkeit aufgetragen.

B.4. Versuch 3 – Anzahl in Abonnements referenzierter Nachrichten

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	[1,2) [m/s]
Länge der Pausen	[10,20) [s]
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	1400 [B]
Größe der Nachrichten des Verarbeiters	1400 [B]
Größe der Ankündigungen des Produzenten	100 [B]
Größe der Ankündigungen des Verarbeiters	100 [B]
Parameter der Simulation	
Anzahl der Experimente	20
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	50 [s]
Anzahl der zwischengespeicherten Nachrichten	20
Anzahl der in Abonnements referenzierten Nachrichten	variabel

Tabelle B.4.:
Übersicht über die Konfiguration des Testszenarios für Versuch 3. Die Anzahl in Abonnements referenzierter Nachrichten wird zwischen 0 und 100 nichtlinear variiert.

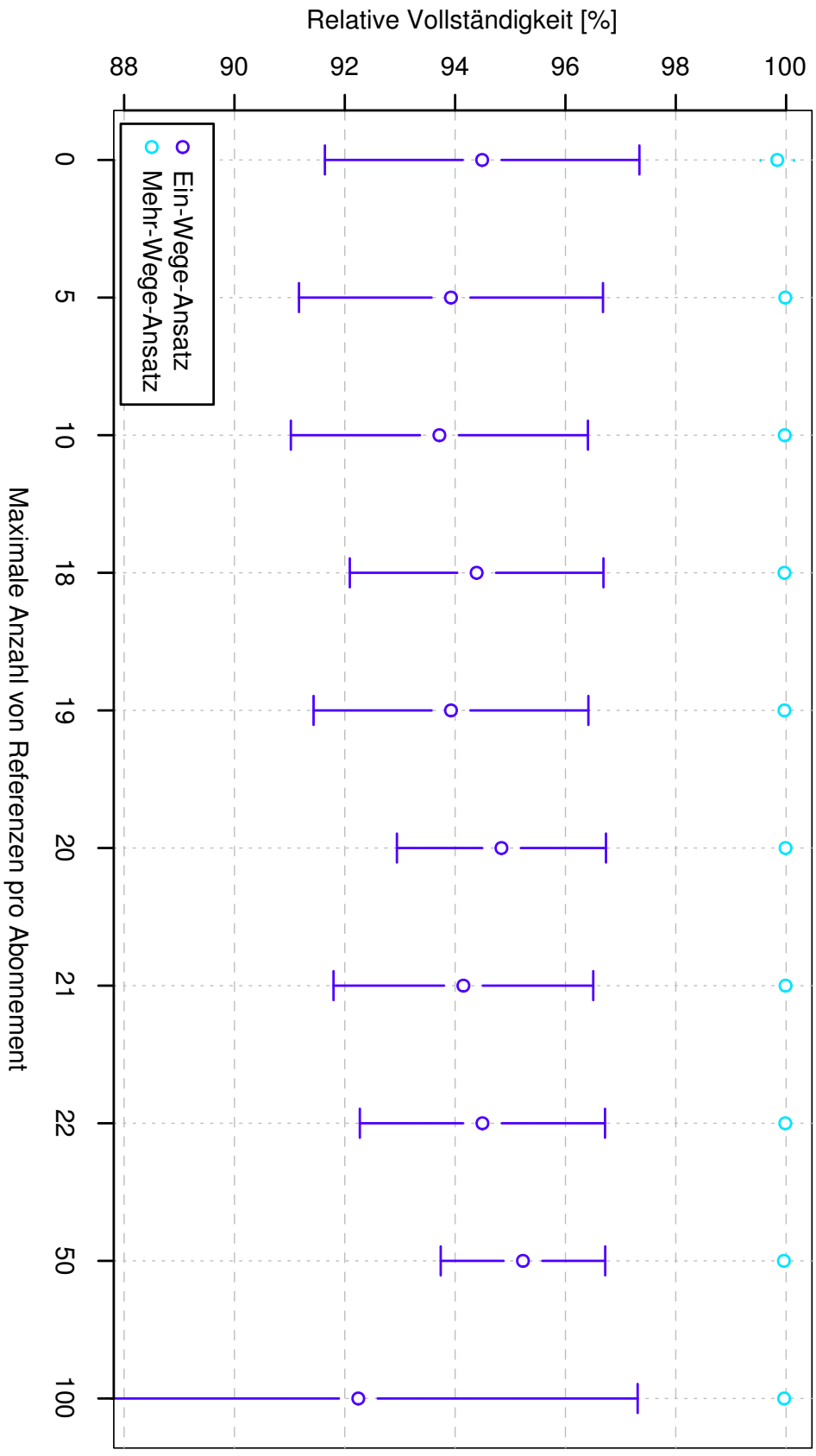


Abbildung B.12.:
Durchschnittliche relative Vollständigkeit in Abhängigkeit von der Anzahl der maximal in jedem Abonnement referenzierten Nachrichten mit 95% Konfidenzintervallen.

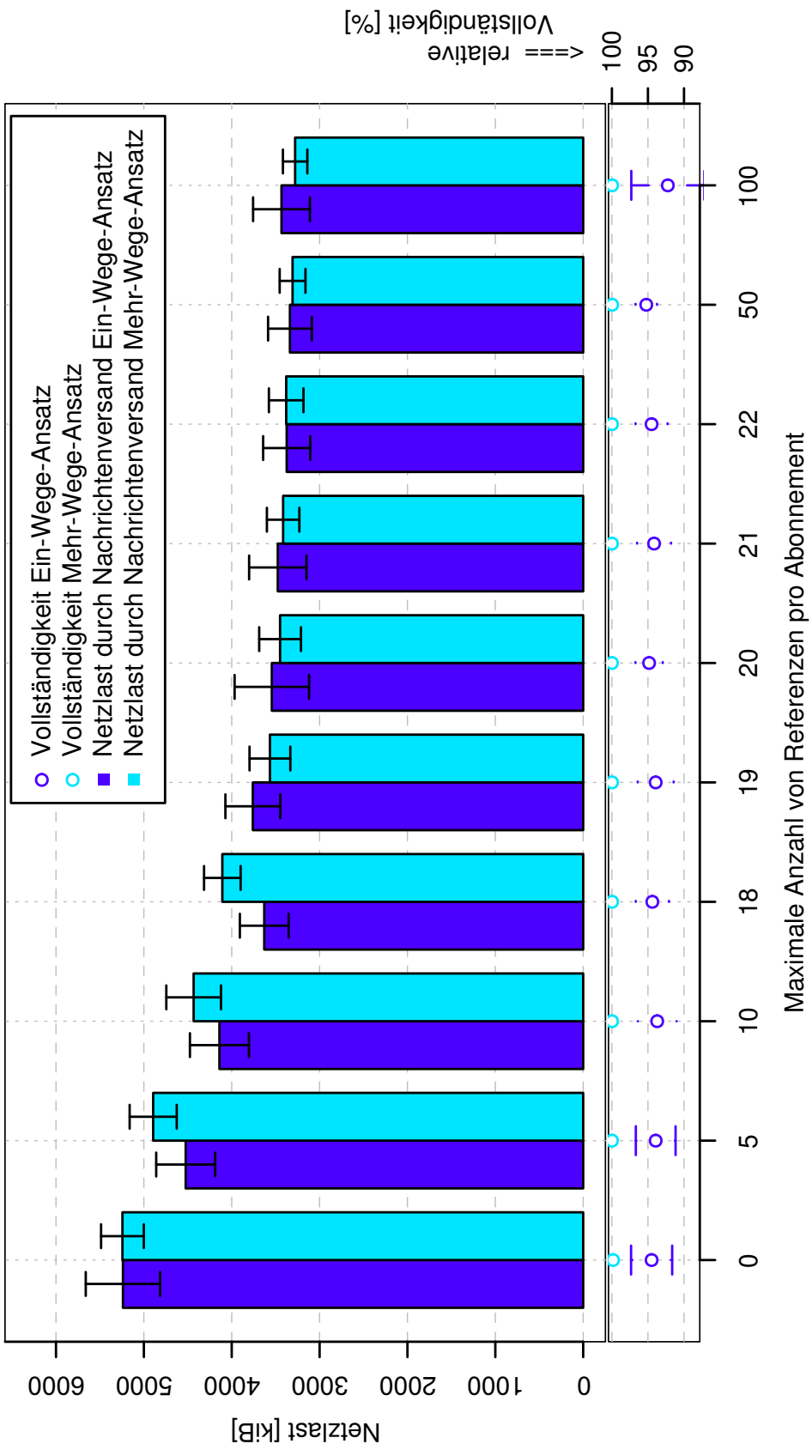


Abbildung B.13.:
Durchschnittliche gesamte Netzlast in Abhängigkeit von der Anzahl der maximal in jedem Abonnement referenzierten Nachrichten mit 95% Konfidenzintervallen. Zur Orientierung ist unten noch einmal die relative Vollständigkeit aufgetragen.

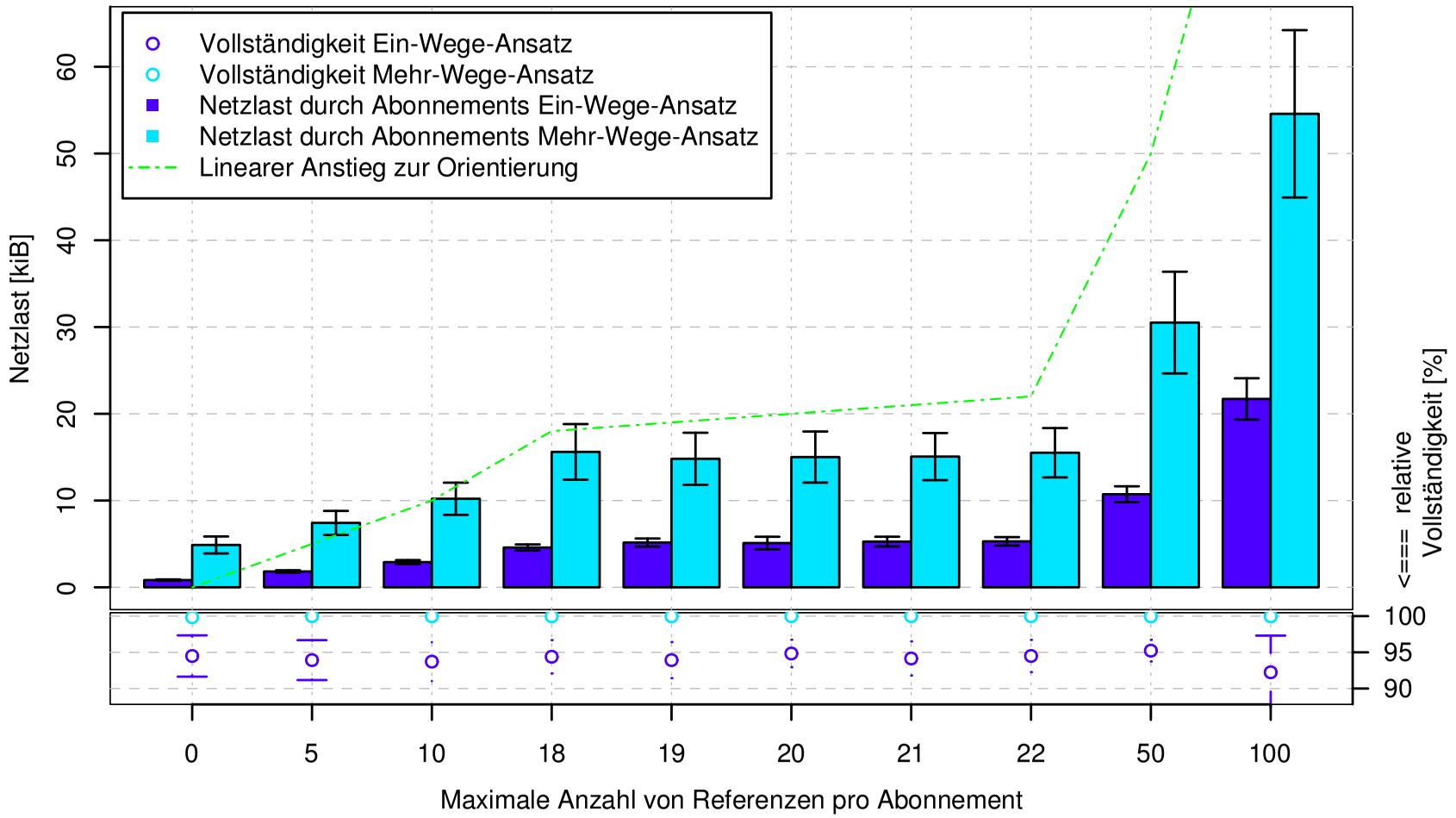


Abbildung B.14.:
Durchschnittliche Netzlast durch die Übertragung von Abonnements in Abhängigkeit von der Anzahl der maximal in jedem Abonnement referenzierten Nachrichten mit 95% Konfidenzintervallen. Da die Konfiguration des Algorithmus auf der x-Achse nicht linear ist, ist zur besseren Orientierung ein linearer Verlauf als gestrichpunktete Linie eingezeichnet. Ebenfalls zur Orientierung ist unten noch einmal die relative Vollständigkeit aufgetragen.

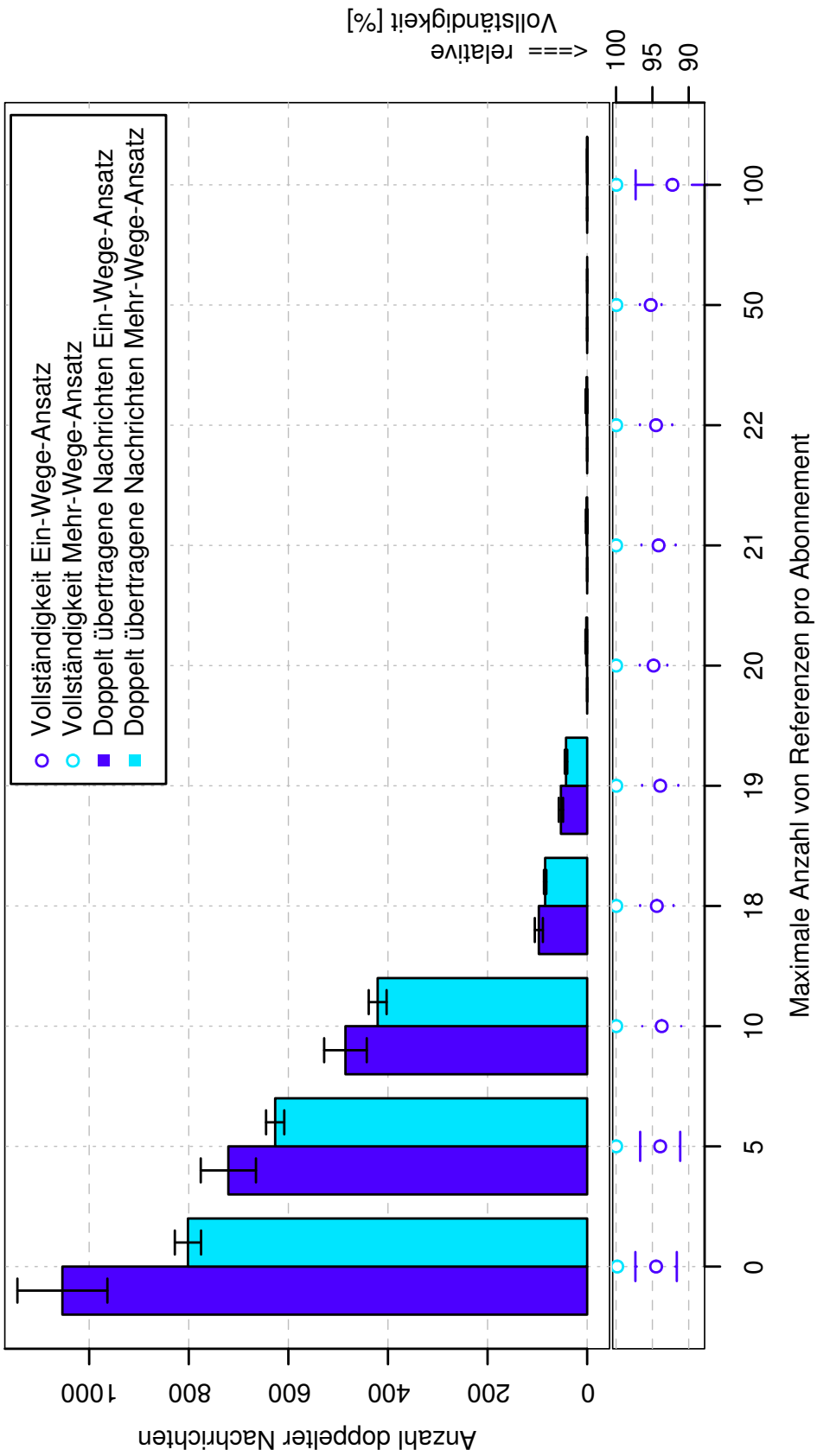


Abbildung B.15.:
Durchschnittliche Anzahl pro Experiment doppelt übertragener Nachrichten in Abhängigkeit von der Anzahl der maximal in jedem Abonnement referenzierten Nachrichten mit 95% Konfidenzintervallen. Zur Orientierung ist unten noch einmal die relative Vollständigkeit aufgetragen.

B.5. Versuch 4 – Auftreten von Pfadfehlern

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	[1,2) [m/s]
Länge der Pausen	[10,20) [s]
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	1400 [B]
Größe der Nachrichten des Verarbeiters	1400 [B]
Größe der Ankündigungen des Produzenten	100 [B]
Größe der Ankündigungen des Verarbeiters	100 [B]
Parameter der Simulation	
Anzahl der Experimente	100
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	10000 [s]
Anzahl der zwischengespeicherten Nachrichten	0
Anzahl der in Abonnements referenzierten Nachrichten	0

Tabelle B.5.: Übersicht über die Konfiguration des Testszenarios für Versuch 4.

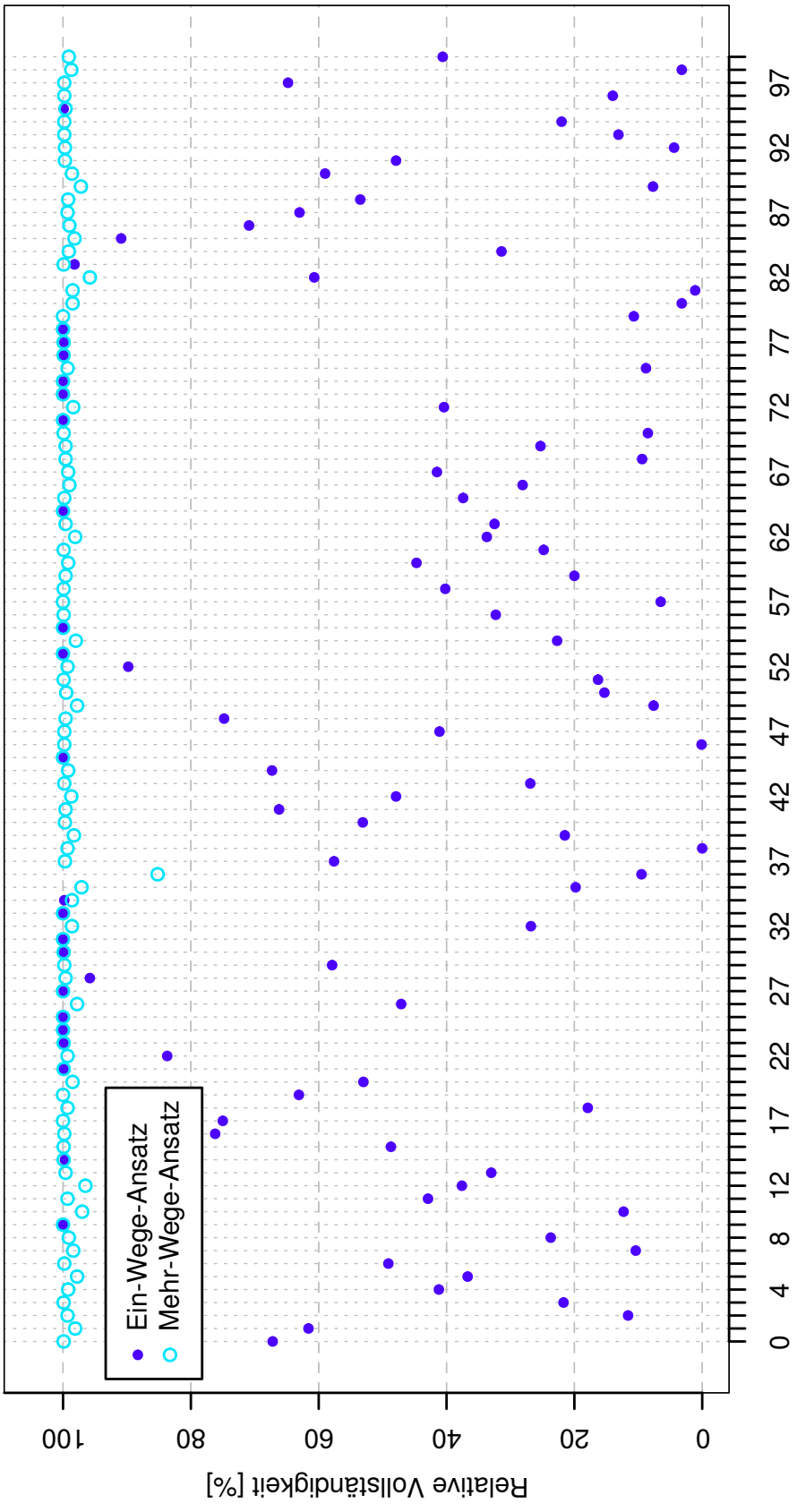


Abbildung B.16.: Relative Vollständigkeit für alle Versuchsdurchläufe.

Abbildung B.17.:
Die Anzahl registrierter Pfadbrüche [unten] sowie die durchschnittliche Pfadnutzungsdauer [oben] für jeden der 100 Versuchsdurchläufe. Die eingezeichneten Linien parallel zur X-Achse stellen die jeweiligen arithmetischen Mittel [durchgehende Linie] mit 95% Konfidenzintervallen [gestrichelte Linien] dar.

B.6. Versuch 5 – Einfluss von Mobilität

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	variabel
Länge der Pausen	[5,10) [s]
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	1400 [B]
Größe der Nachrichten des Verarbeiters	1400 [B]
Größe der Ankündigungen des Produzenten	100 [B]
Größe der Ankündigungen des Verarbeiters	100 [B]
Parameter der Simulation	
Anzahl der Experimente	20
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	50 [s]
Anzahl der zwischengespeicherten Nachrichten	20
Anzahl der in Abonnements referenzierten Nachrichten	20

Tabelle B.6.:
Übersicht über die Konfiguration des Testszenarios für Versuch 5. Die Geschwindigkeit der Geräte wird zwischen 1 [m/s] und 10 [m/s] linear variiert.

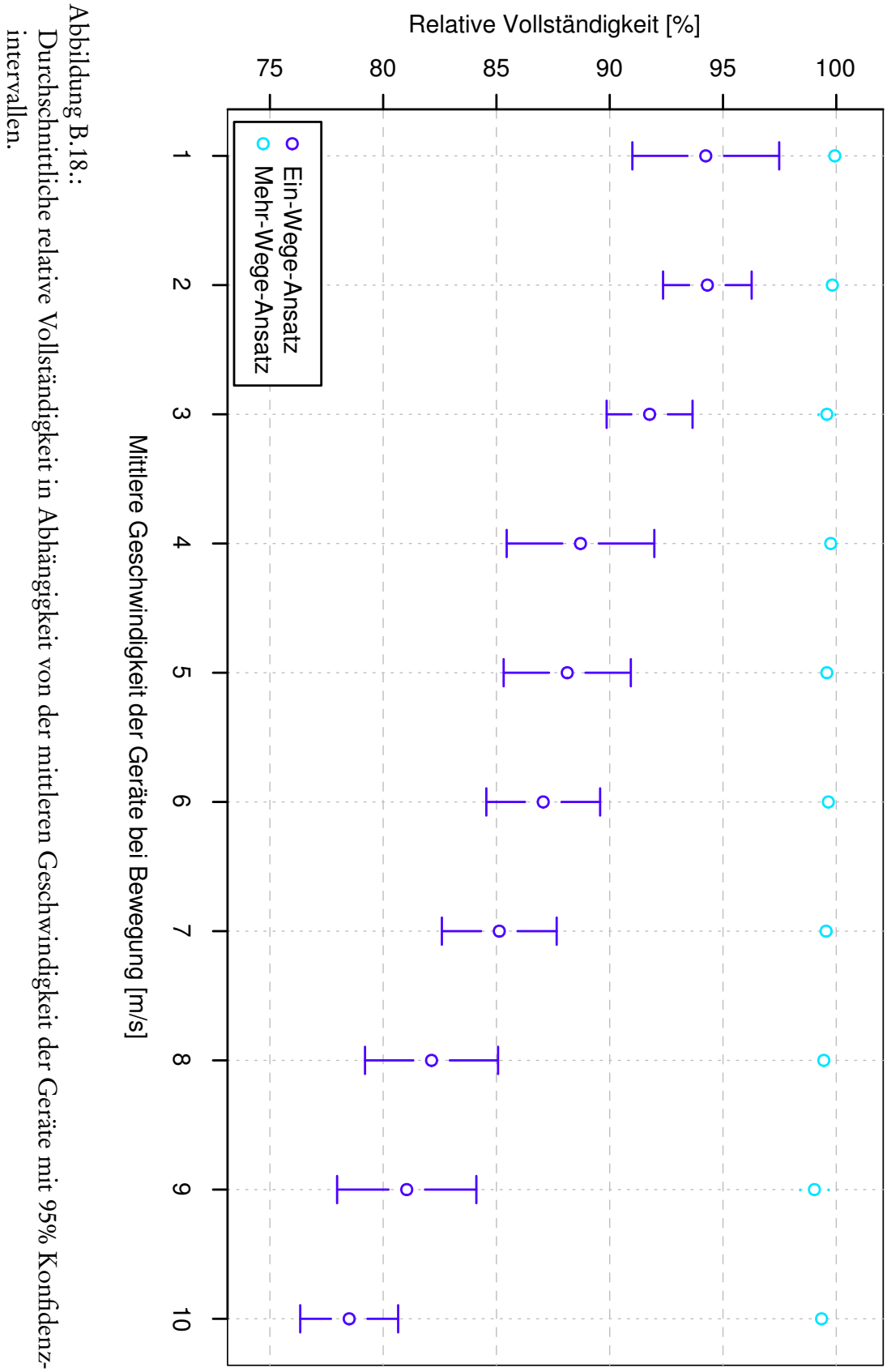


Abbildung B.18.:
Durchschnittliche relative Vollständigkeit in Abhängigkeit von der mittleren Geschwindigkeit der Geräte mit 95% Konfidenzintervallen.

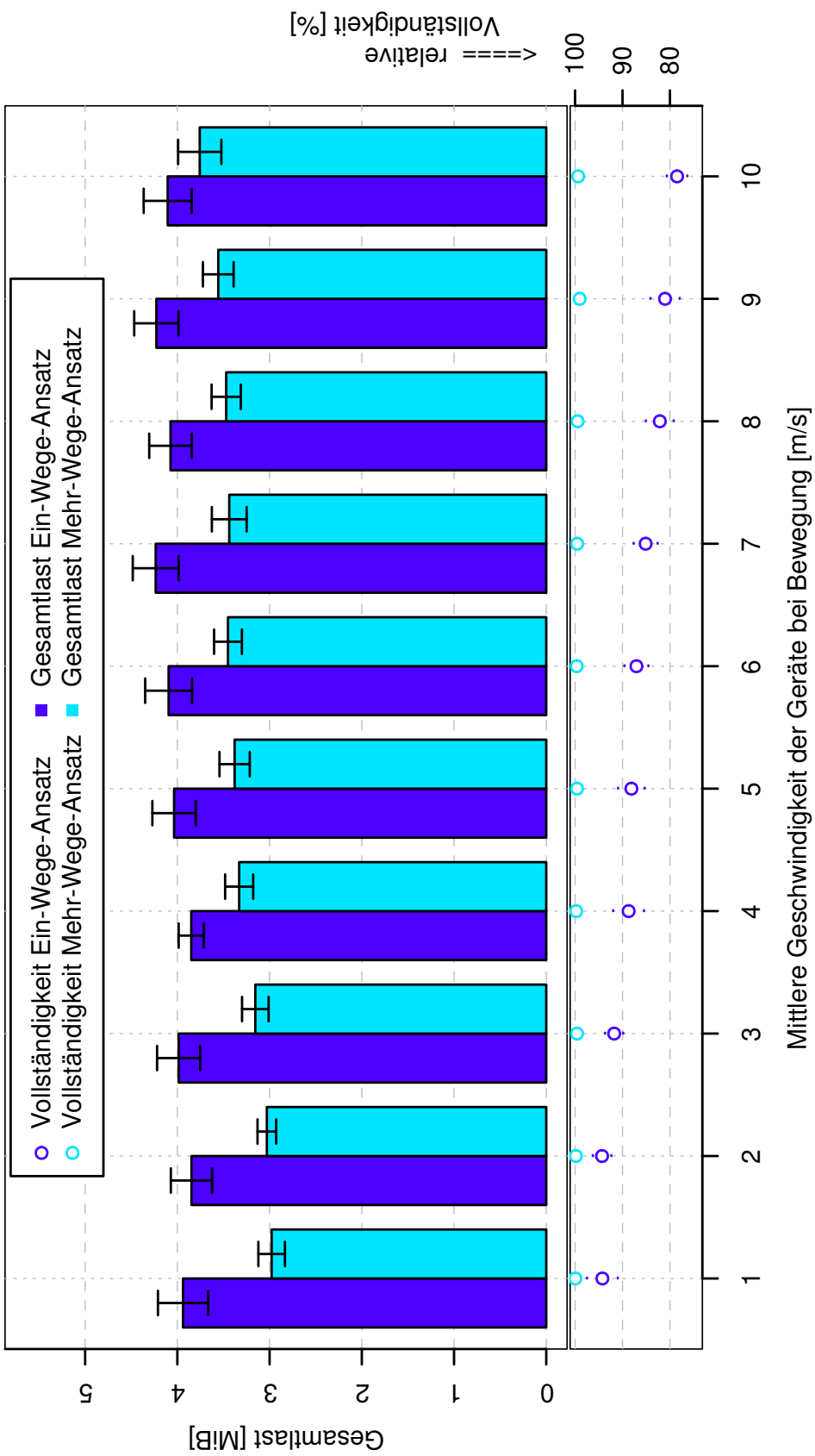


Abbildung B.19.:
Durchschnittlich insgesamt erzeugte Netzlast pro Versuchsdurchlauf für Versuch 5 in Abhängigkeit von der mittleren Geschwindigkeit der Geräte mit 95% Konfidenzintervallen.

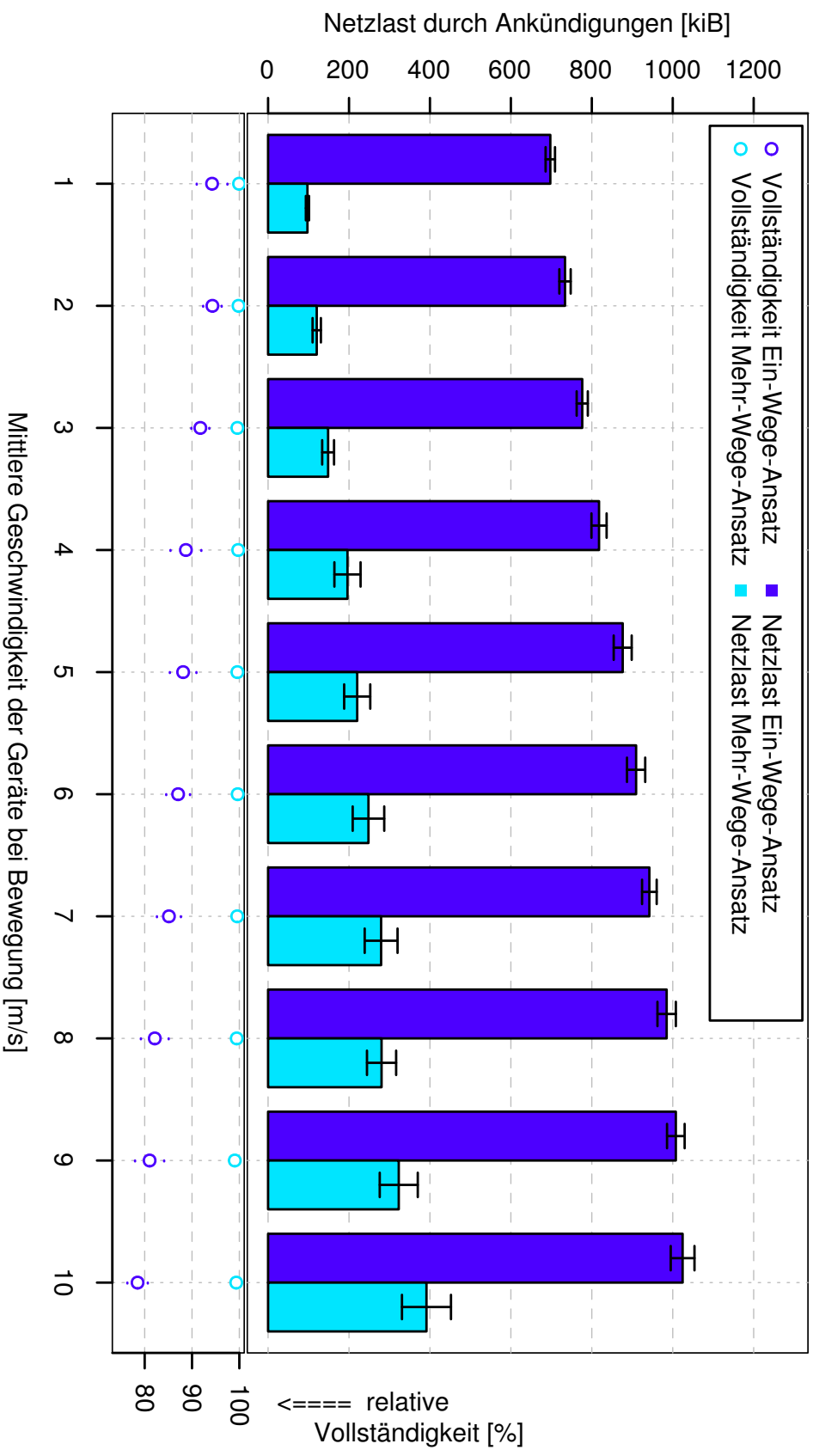


Abbildung B.20.:
Durchschnittlich durch Ankündigungen erzeugte Netzlast pro Versuchsdurchlauf für Versuch 5 in Abhängigkeit von der mittleren Geschwindigkeit der Geräte mit 95% Konfidenzintervallen.

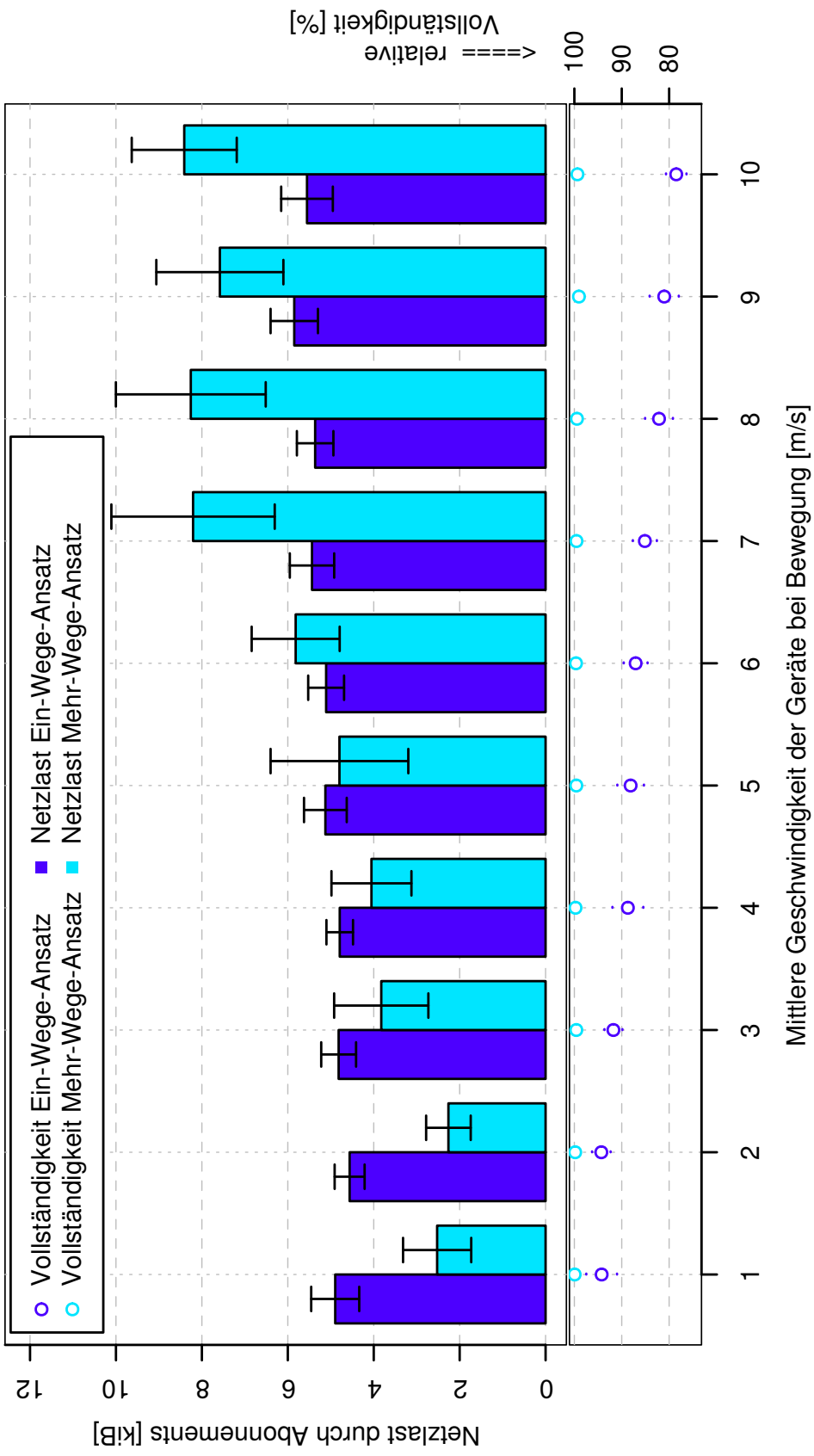


Abbildung B.21.:
Durchschnittlich durch Abonnements erzeugte Netzlast pro Versuchsdurchlauf für Versuch 5 in Abhängigkeit von der mittleren Geschwindigkeit der Geräte mit 95% Konfidenzintervallen.

B.7. Versuch 6 – Vergleich mit Flooding

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	[1,2) [m/s]
Länge der Pausen	[10,20) [s]
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	500 [B]
Größe der Nachrichten des Verarbeiters	500 [B]
Größe der Ankündigungen des Produzenten	100 [B]
Größe der Ankündigungen des Verarbeiters	100 [B]
Parameter der Simulation	
Anzahl der Experimente	20
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	50 [s]
Anzahl der zwischengespeicherten Nachrichten	20
Anzahl der in Abonnements referenzierten Nachrichten	20

Tabelle B.7.:
Übersicht über die Konfiguration des Testszenarios für Versuch 6. Der Flooding-Algorithmus benötigt keine spezielle Konfiguration. Identifizierer der Nachrichten werden für 10 [s] gespeichert, um einer erneuten Übertragung einer bereits versandten Nachricht vorzubeugen.

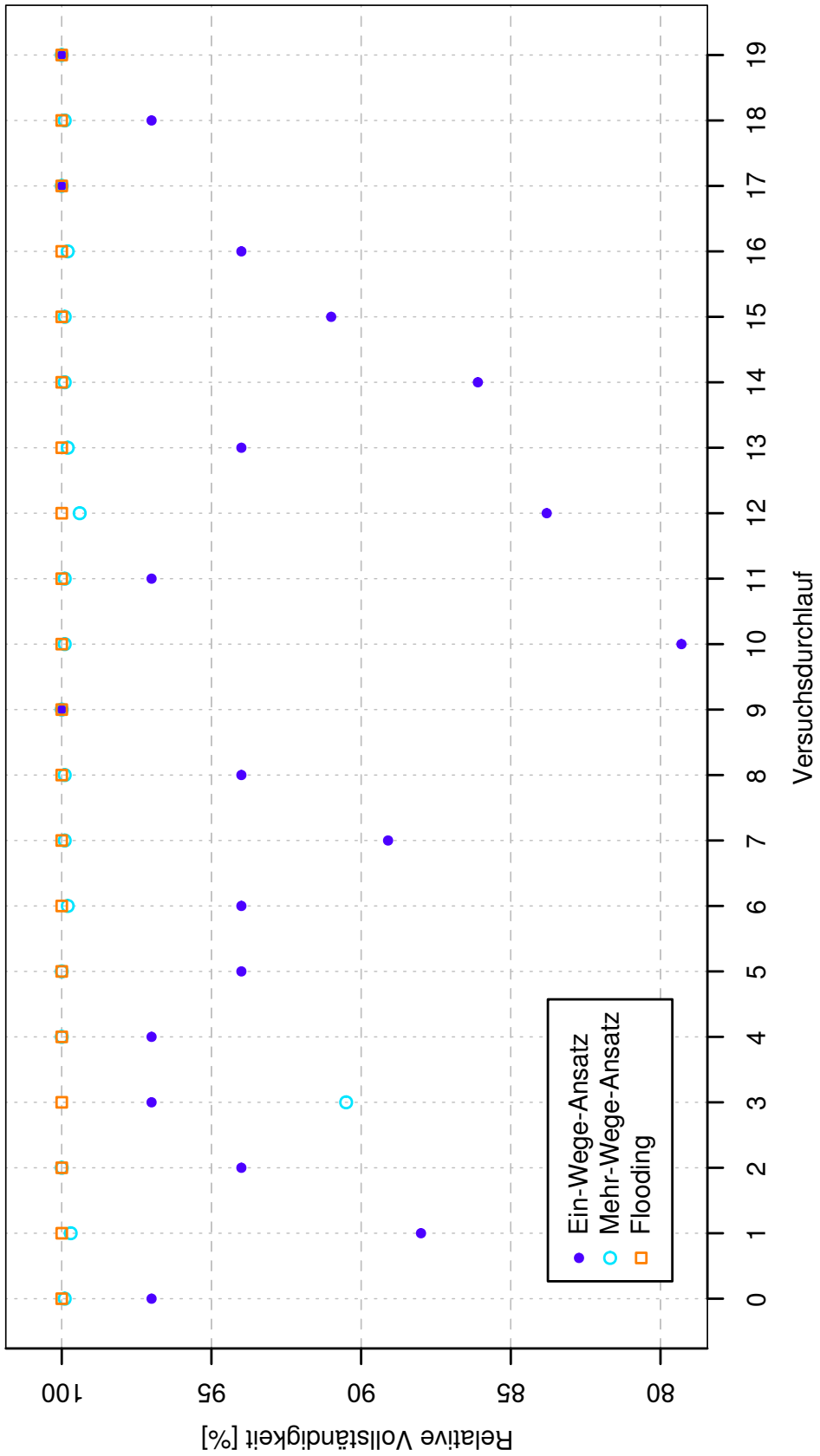


Abbildung B.22.: Relative Vollständigkeit für Versuch 6 für jeden Versuchsdurchlauf für die drei verglichenen Ansätze.

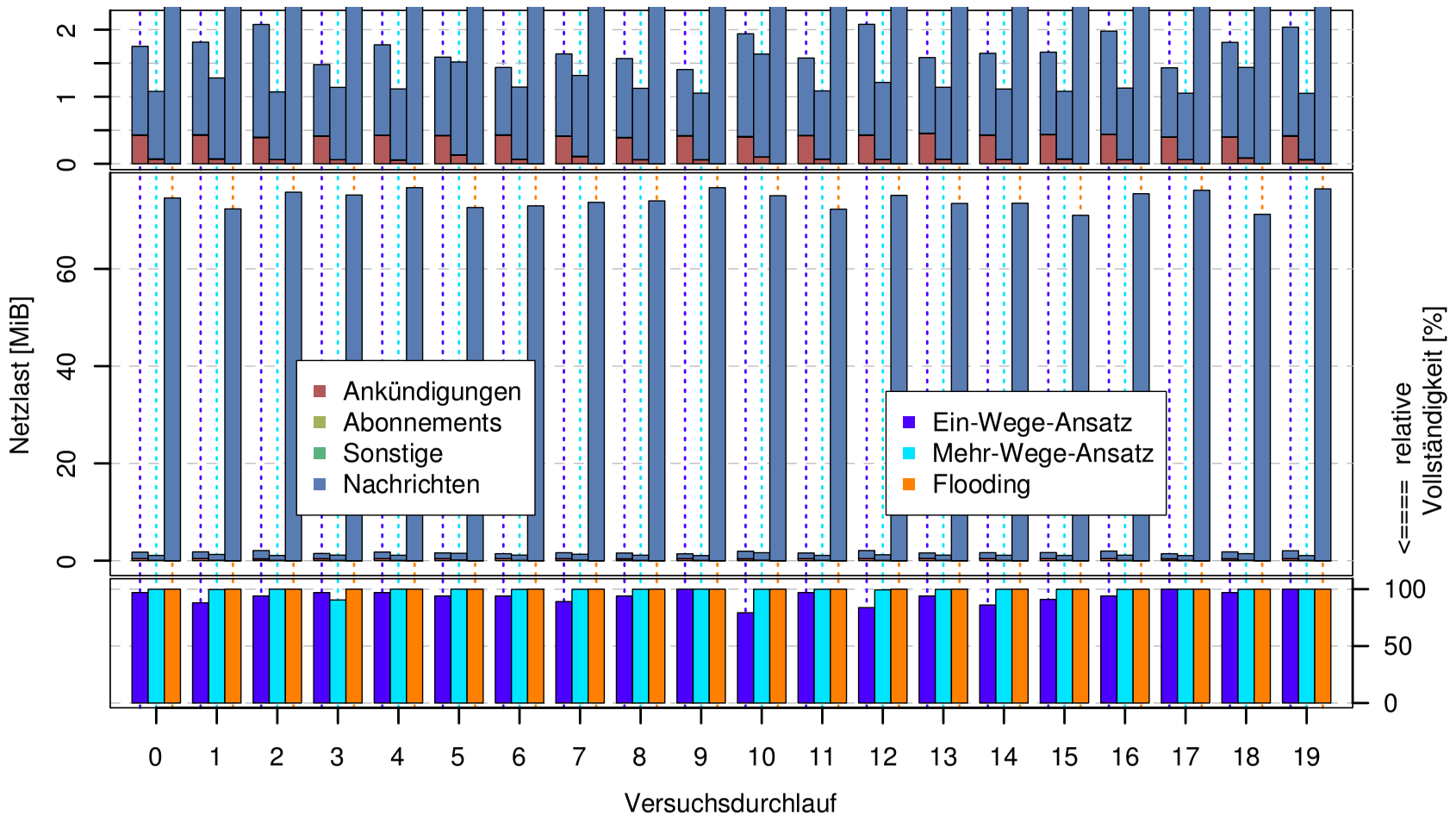


Abbildung B.23.:
Netzlast für Versuch 6 für jeden Versuchsdurchlauf jeweils für die drei verglichenen Ansätze. Zur Orientierung ist unten die Vollständigkeit für den entsprechenden Versuch aufgetragen. Der obere Abschnitt des Diagramms stellt einen vergrößerten Ausschnitt am nahe 0 gelegenen Ende der Y-Achse dar, in dem man den Anteil der nicht Nutzlast tragenden Pakete an der Netzlast abschätzen kann. Balken für denselben Algorithmus liegen in den Diagrammteilen übereinander und sind durch eine gestrichelte Linie in der Farbe des Algorithmus hinterlegt.

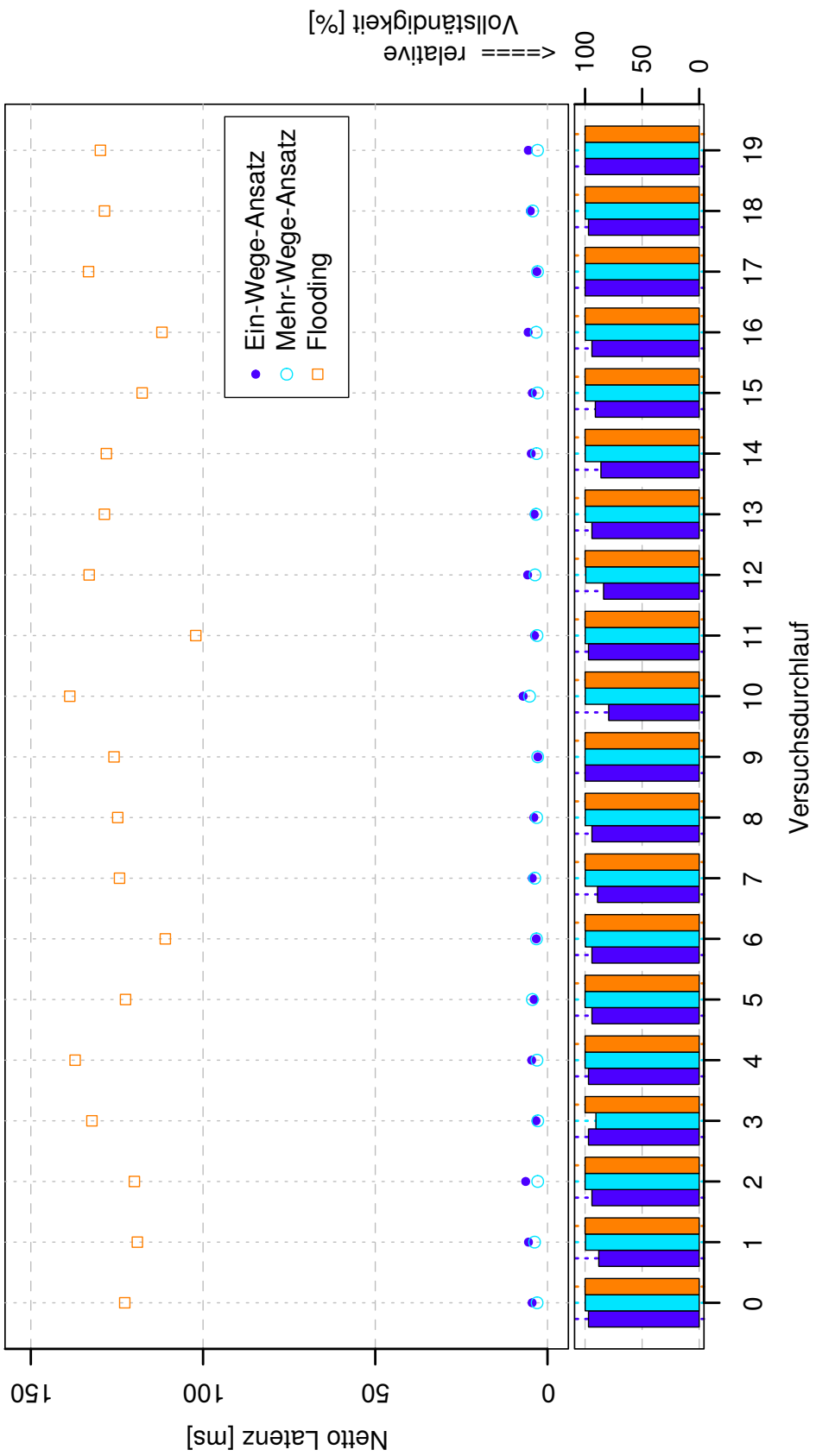


Abbildung B.24.:
Netto-Latenz für Versuch 6 für jeden Versuchsdurchlauf jeweils für die drei verglichenen Ansätze. Zur Orientierung ist unten die Vollständigkeit für den entsprechenden Versuch aufgetragen.

B.8. Versuch 7 – Verhalten bei Anwendungen der Klasse 2

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	0 [m/s]
Länge der Pausen	∞
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	1400 [B]
Größe der Nachrichten des Verarbeiters	1400 [B]
Größe der Ankündigungen des Produzenten	100 [B]
Größe der Ankündigungen des Verarbeiters	100 [B]
Parameter der Simulation	
Anzahl der Experimente	20
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	1000 [s]
Anzahl der zwischengespeicherten Nachrichten	20
Anzahl der in Abonnements referenzierten Nachrichten	20

Tabelle B.8.: Übersicht über die Konfiguration des Testszenarios für Versuch 7.

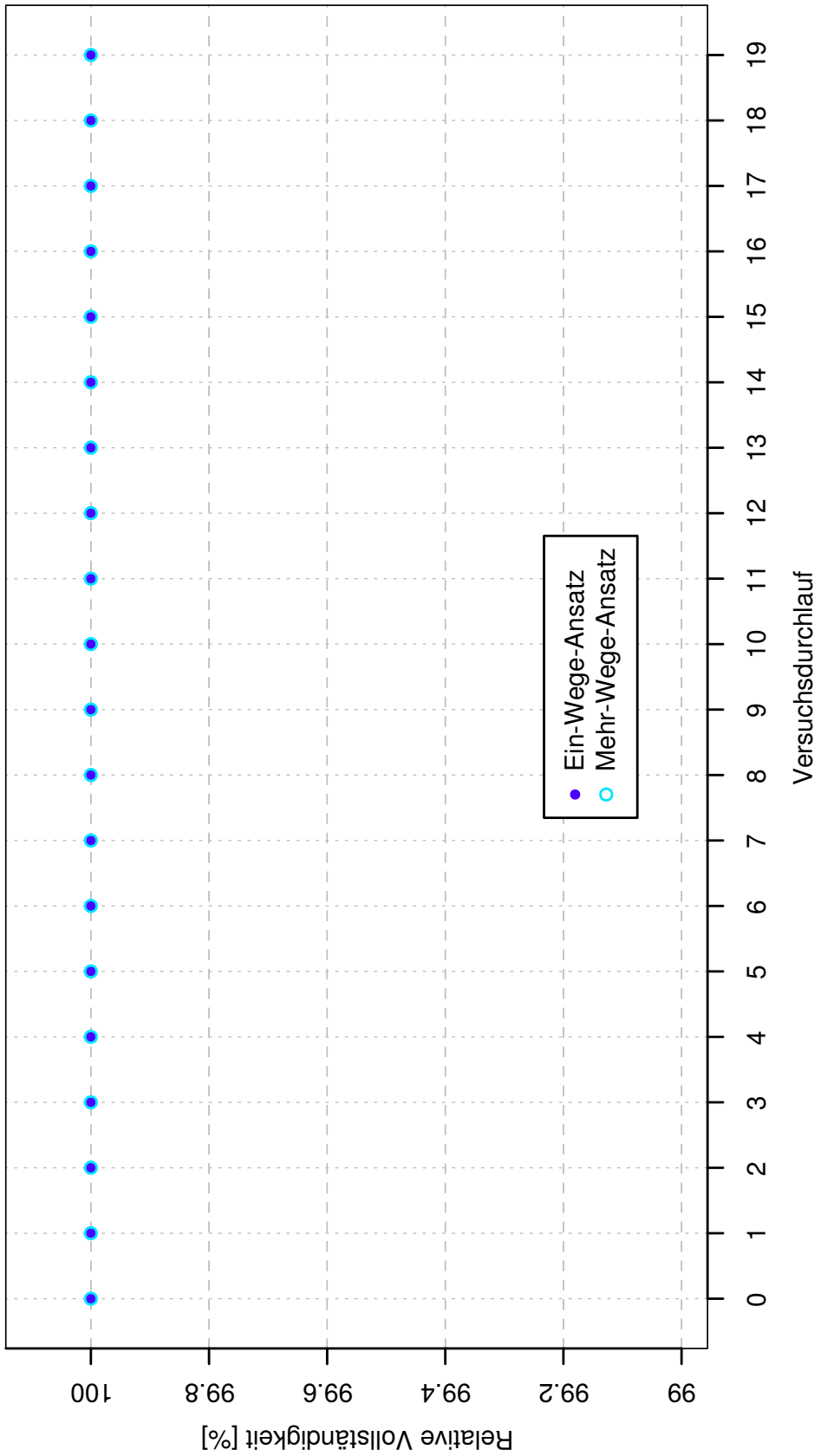


Abbildung B.25.: Relative Vollständigkeit für Versuch 7 für jeden Versuchsdurchlauf für die beiden verglichenen Ansätze.

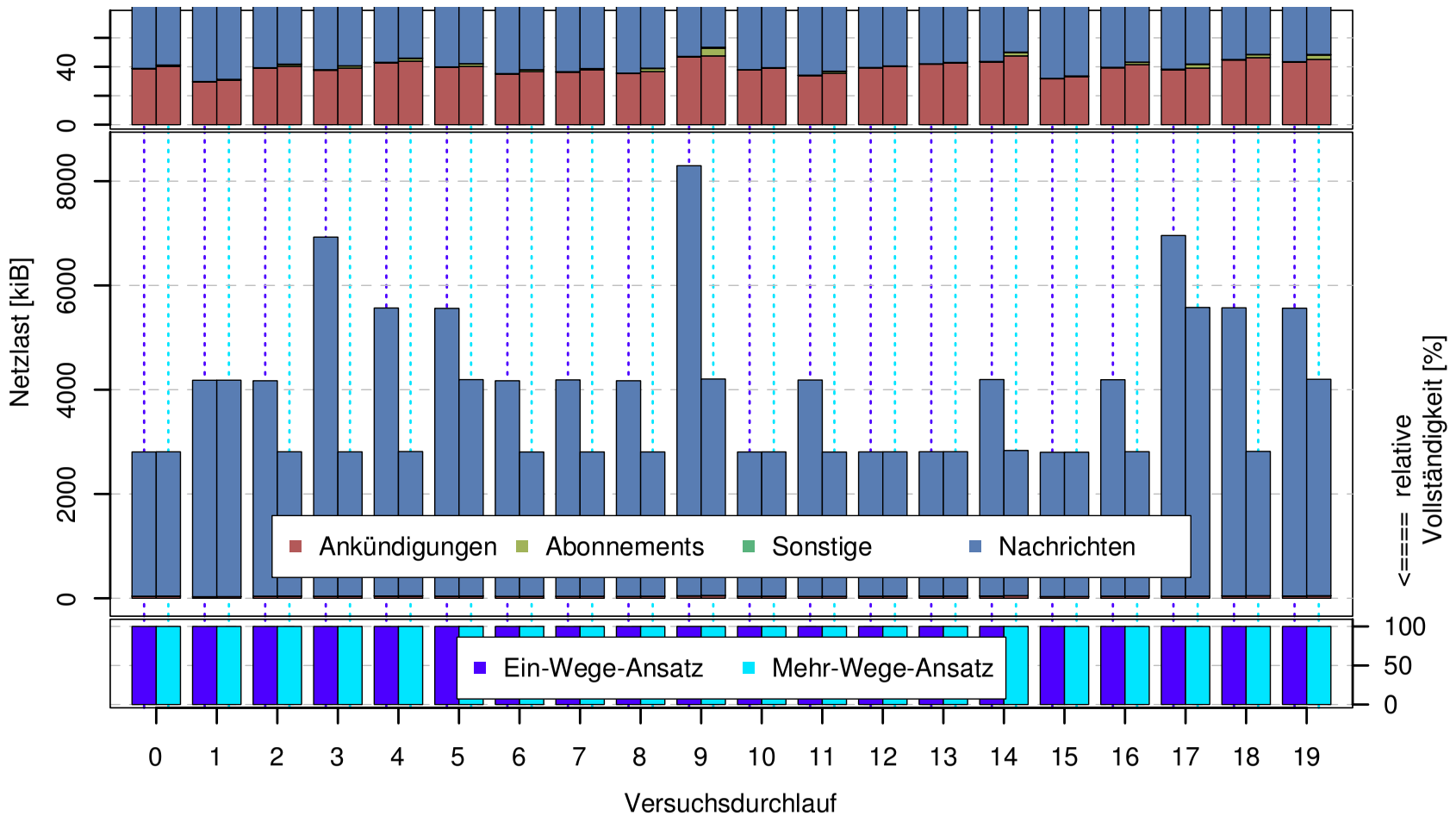


Abbildung B.26.:
Netzlast für Versuch 7 für jeden Versuchsdurchlauf jeweils für die beiden verglichenen Ansätze. Zur Orientierung ist unten die Vollständigkeit für den entsprechenden Versuch aufgetragen. Der obere Abschnitt des Diagramms stellt einen vergrößerten Ausschnitt am nahe 0 gelegenen Ende der Y-Achse dar, in dem man den Anteil der nicht Nutzlast tragenden Pakete an der Netzlast abschätzen kann. Balken für denselben Algorithmus liegen in den Diagrammteilen übereinander und sind durch eine gestrichelte Linie in der Farbe des Algorithmus hinterlegt.

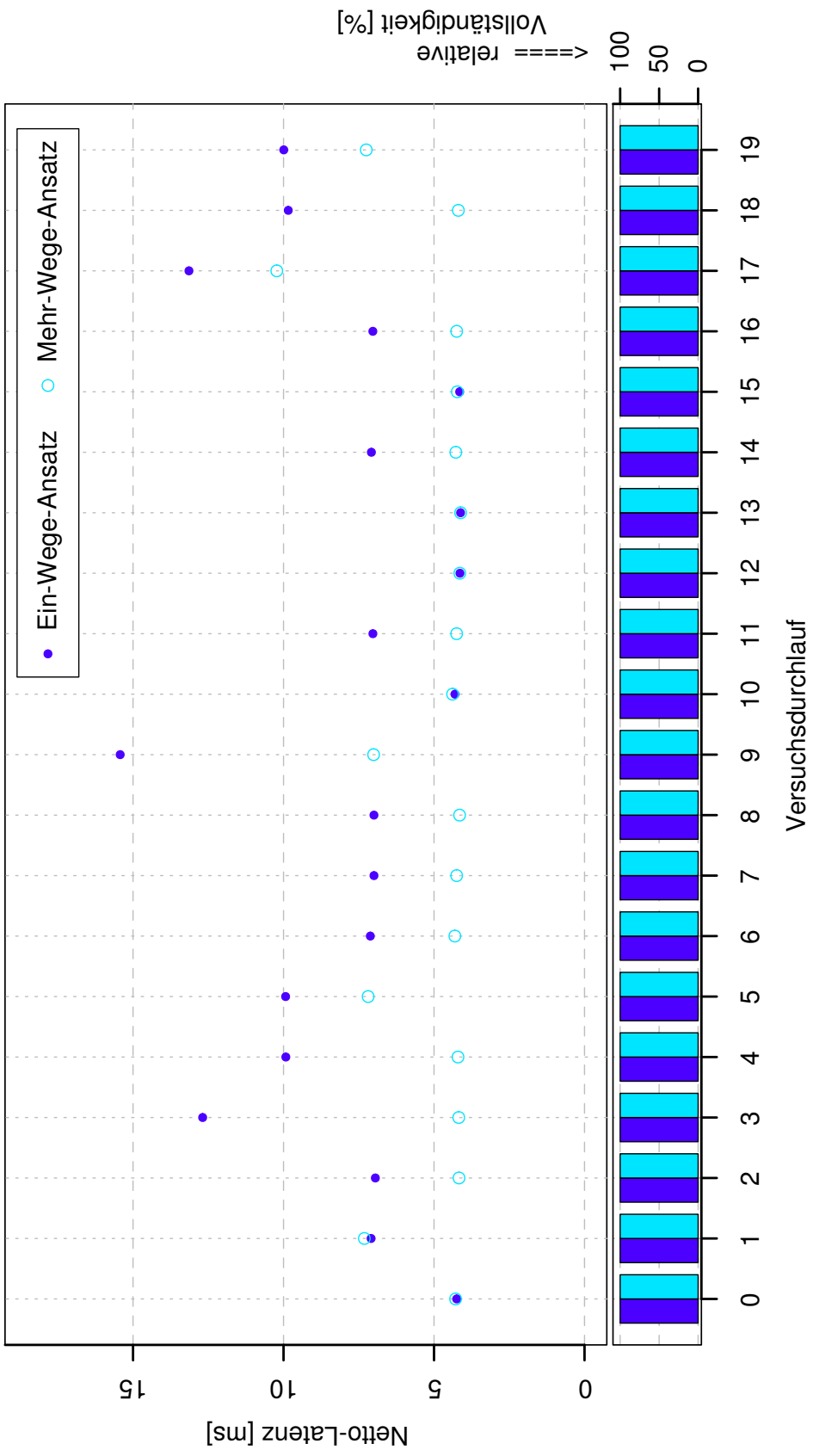


Abbildung B.27.:
Netto-Latenz für Versuch 7 für jeden Versuchsdurchlauf jeweils für die drei verglichenen Ansätze. Zur Orientierung ist unten die Vollständigkeit für den entsprechenden Versuch aufgetragen.

B.9. Versuch 8 – Verhalten bei Anwendungen der Klasse 3

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	[1,2) [m/s]
Länge der Pausen	[10,20) [s]
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	50 [kiB]
Größe der Nachrichten des Verarbeiters	35 [kiB]
Größe der Ankündigungen des Produzenten	200 [B]
Größe der Ankündigungen des Verarbeiters	200 [B]
Parameter der Simulation	
Anzahl der Experimente	20
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	50 [s]
Anzahl der zwischengespeicherten Nachrichten	20
Anzahl der in Abonnements referenzierten Nachrichten	20

Tabelle B.9.: Übersicht über die Konfiguration des Testszenarios für Versuch 8.

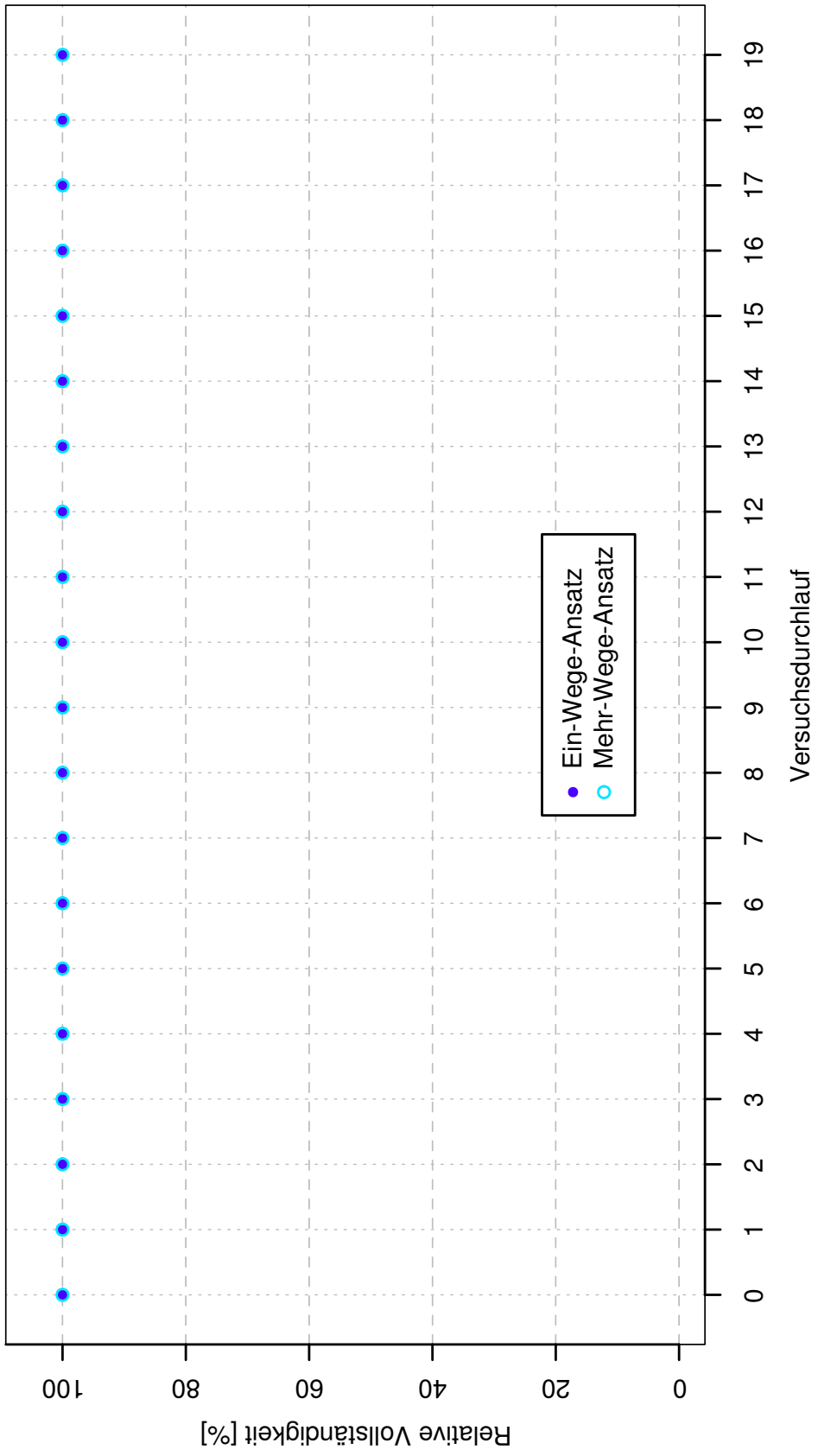


Abbildung B.28.: Relative Vollständigkeit für Versuch 8 für jeden Versuchsdurchlauf für die beiden verglichenen Ansätze.

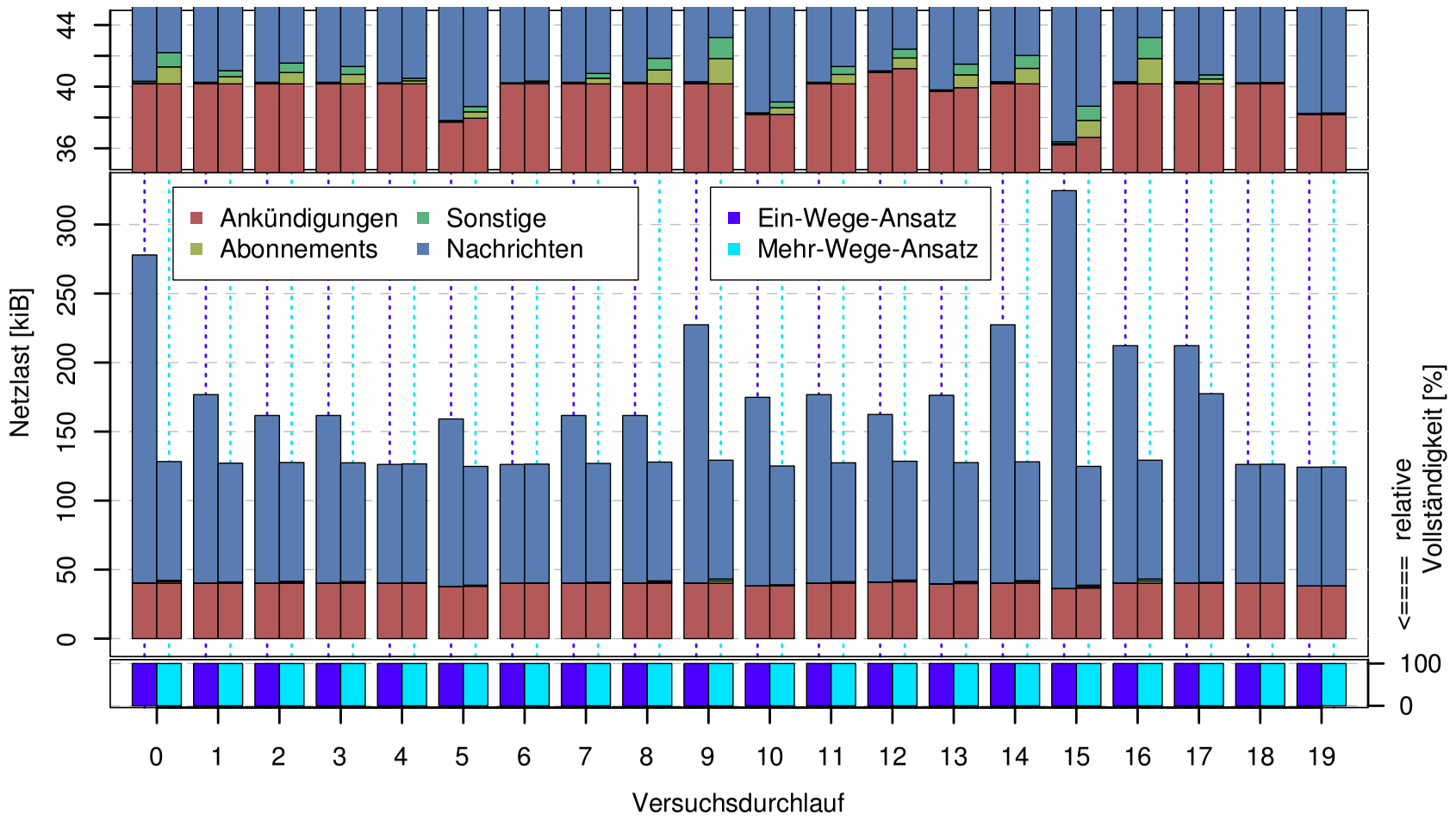


Abbildung B.29.:
Netzlast für Versuch 8 für jeden Versuchsdurchlauf jeweils für die beiden verglichenen Ansätze. Zur Orientierung ist unten die Vollständigkeit für den entsprechenden Versuch aufgetragen. Der obere Abschnitt des Diagramms stellt einen vergrößerten Ausschnitt der Y-Achse dar, in dem man den Anteil der Abonnements und im Wesentlichen Kündigungen an der Netzlast abschätzen kann. Balken für denselben Algorithmus liegen in den Diagrammteilen übereinander und sind durch eine gestrichelte Linie in der Farbe des Algorithmus hinterlegt.

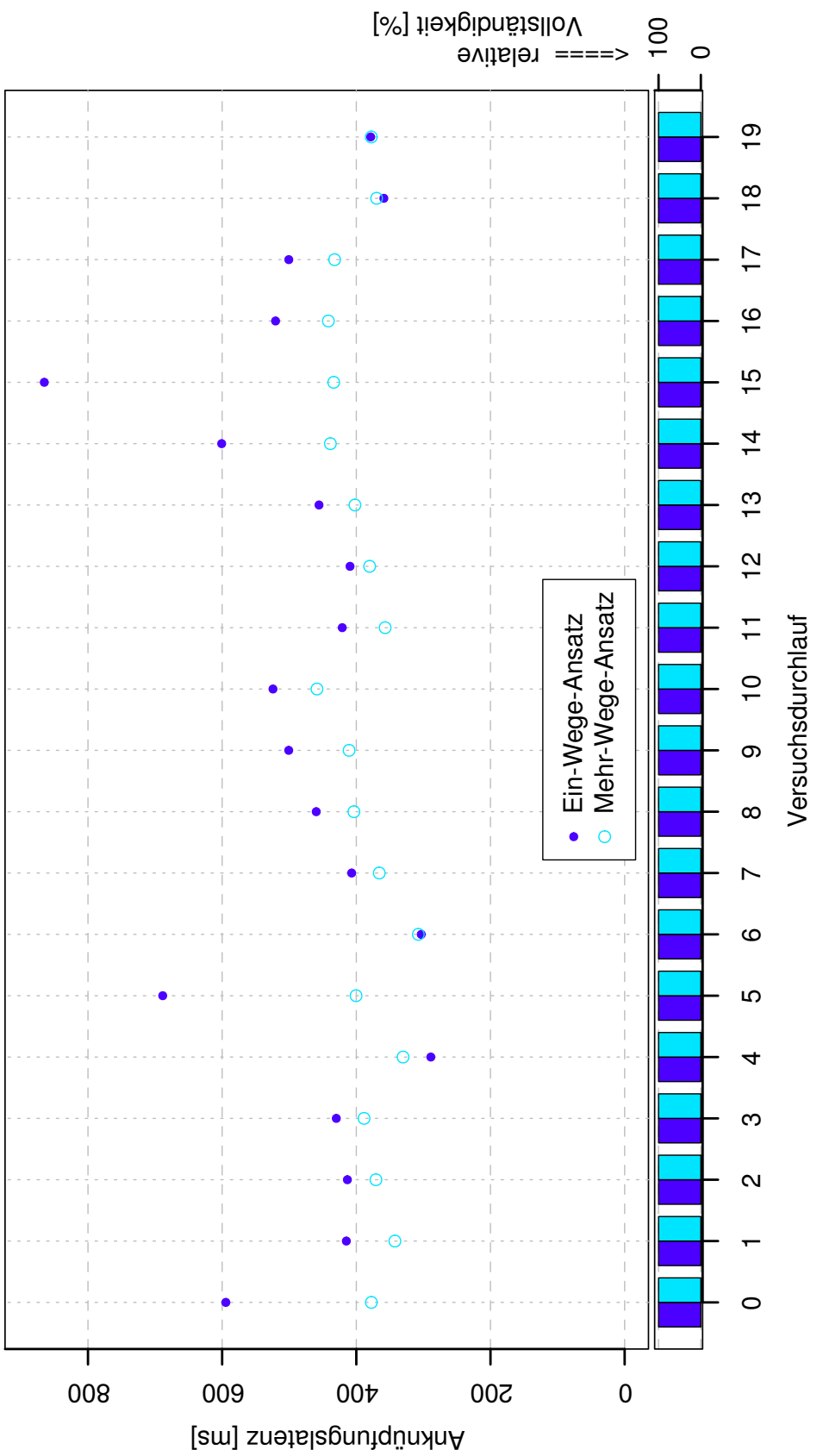


Abbildung B.30.:
Anknüpfungslatenz für Versuch 8 für jeden Versuchsdurchlauf jeweils für die drei verglichenen Ansätze. Zur Orientierung ist unten die Vollständigkeit für den entsprechenden Versuch aufgetragen.

B.10. Versuch 9 – Verhalten bei Anwendungen der Klasse 4

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	0 [m/s]
Länge der Pausen	∞
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	20 [kiB]
Größe der Nachrichten des Verarbeiters	20 [kiB]
Größe der Ankündigungen des Produzenten	200 [B]
Größe der Ankündigungen des Verarbeiters	200 [B]
Parameter der Simulation	
Anzahl der Experimente	20
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	1000 [s]
Anzahl der zwischengespeicherten Nachrichten	20
Anzahl der in Abonnements referenzierten Nachrichten	20

Tabelle B.10.: Übersicht über die Konfiguration des Testszenarios für Versuch 9.

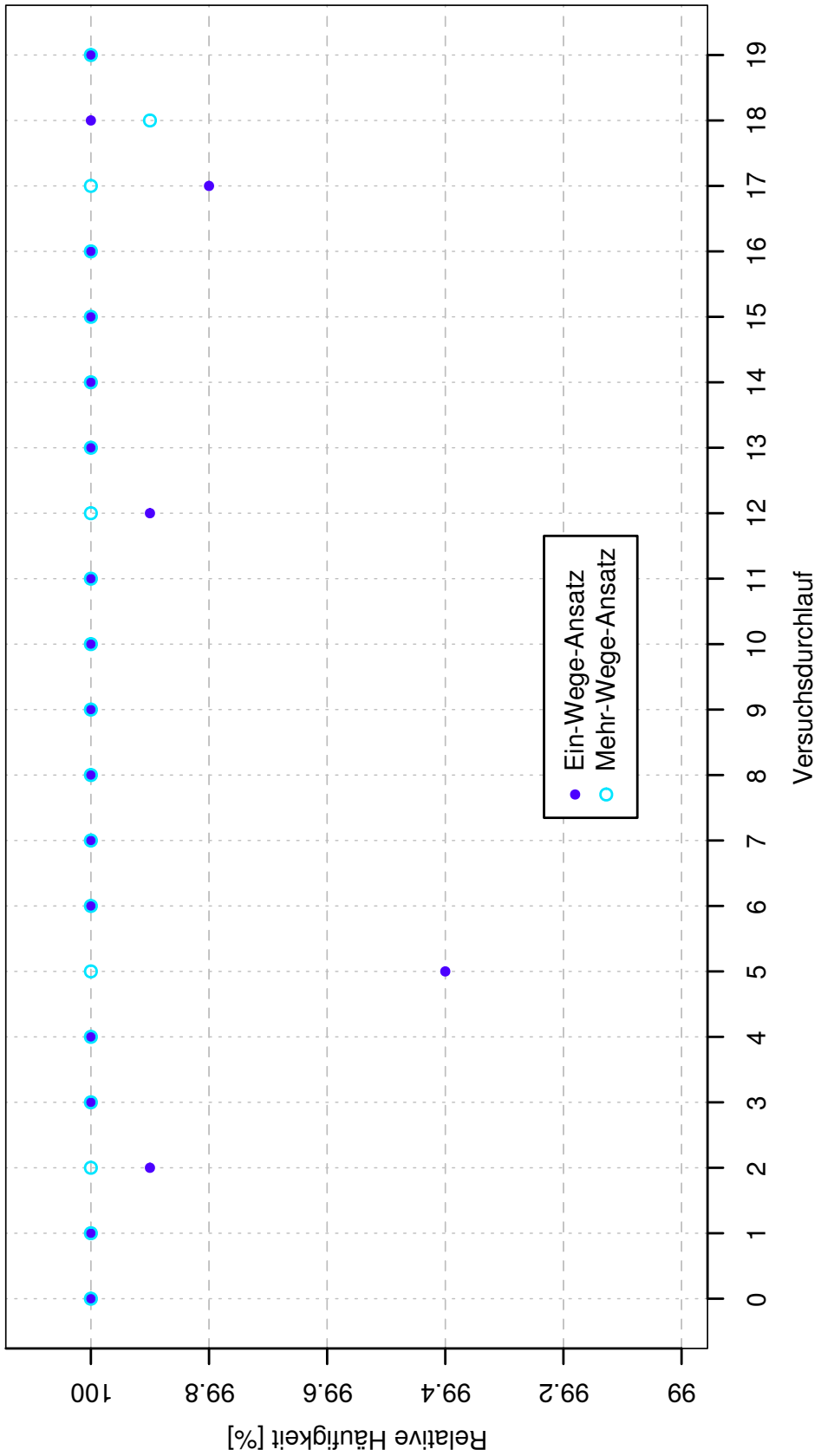


Abbildung B.31.: Relative Vollständigkeit für Versuch 9 für jeden Versuchsdurchlauf für die beiden verglichenen Ansätze.

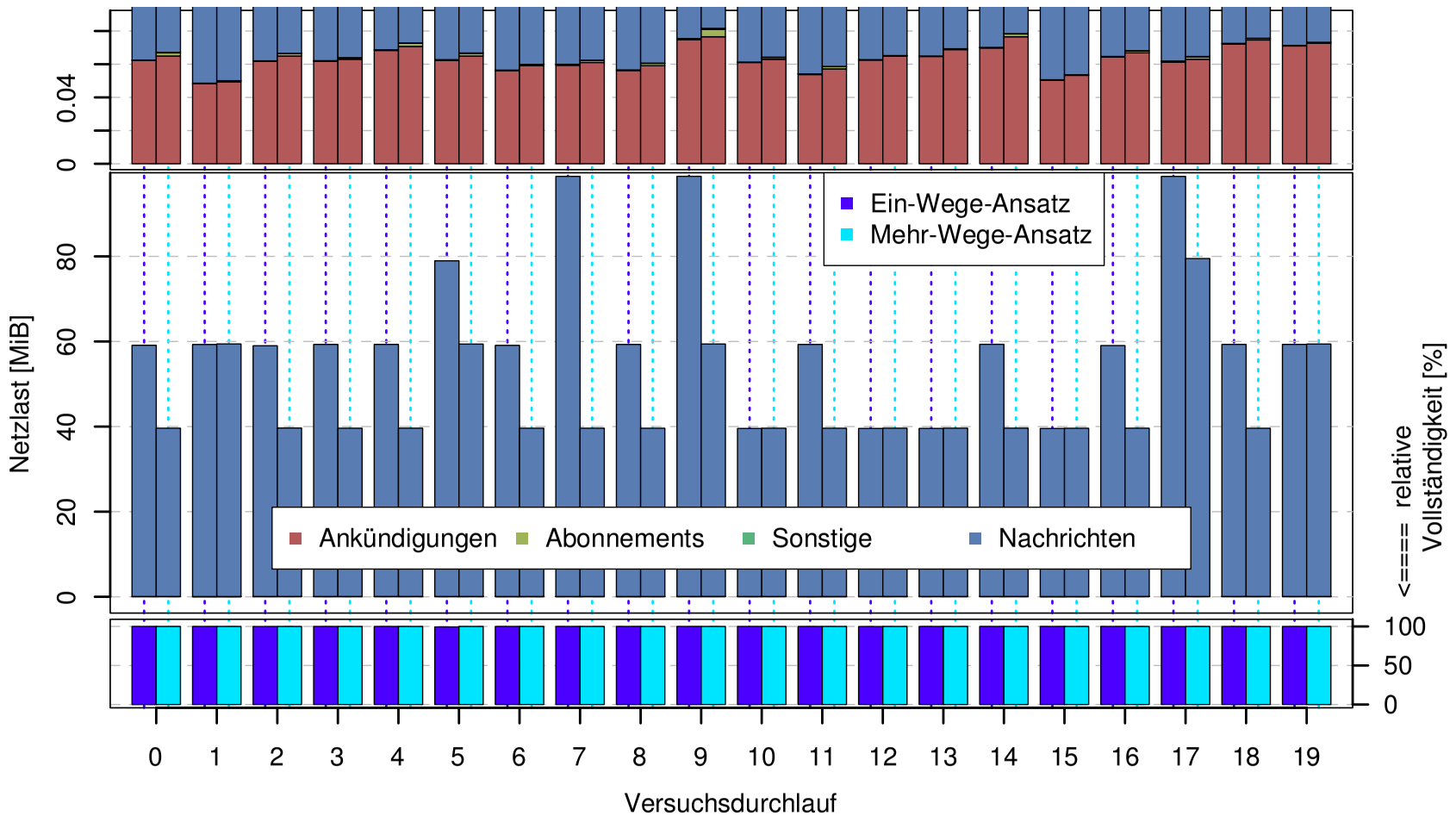


Abbildung B.32.:
Netzlast für Versuch 9 für jeden Versuchsdurchlauf jeweils für die beiden verglichenen Ansätze. Zur Orientierung ist unten die Vollständigkeit für den entsprechenden Versuch aufgetragen. Der obere Abschnitt des Diagramms stellt einen vergrößerten Ausschnitt am nahe 0 gelegenen Ende der Y-Achse dar, in dem man den Anteil der nicht Nutzlast tragenden Pakete an der Netzlast abschätzen kann. Balken für denselben Algorithmus liegen in den Diagrammteilen übereinander und sind durch eine gestrichelte Linie in der Farbe des Algorithmus hinterlegt.

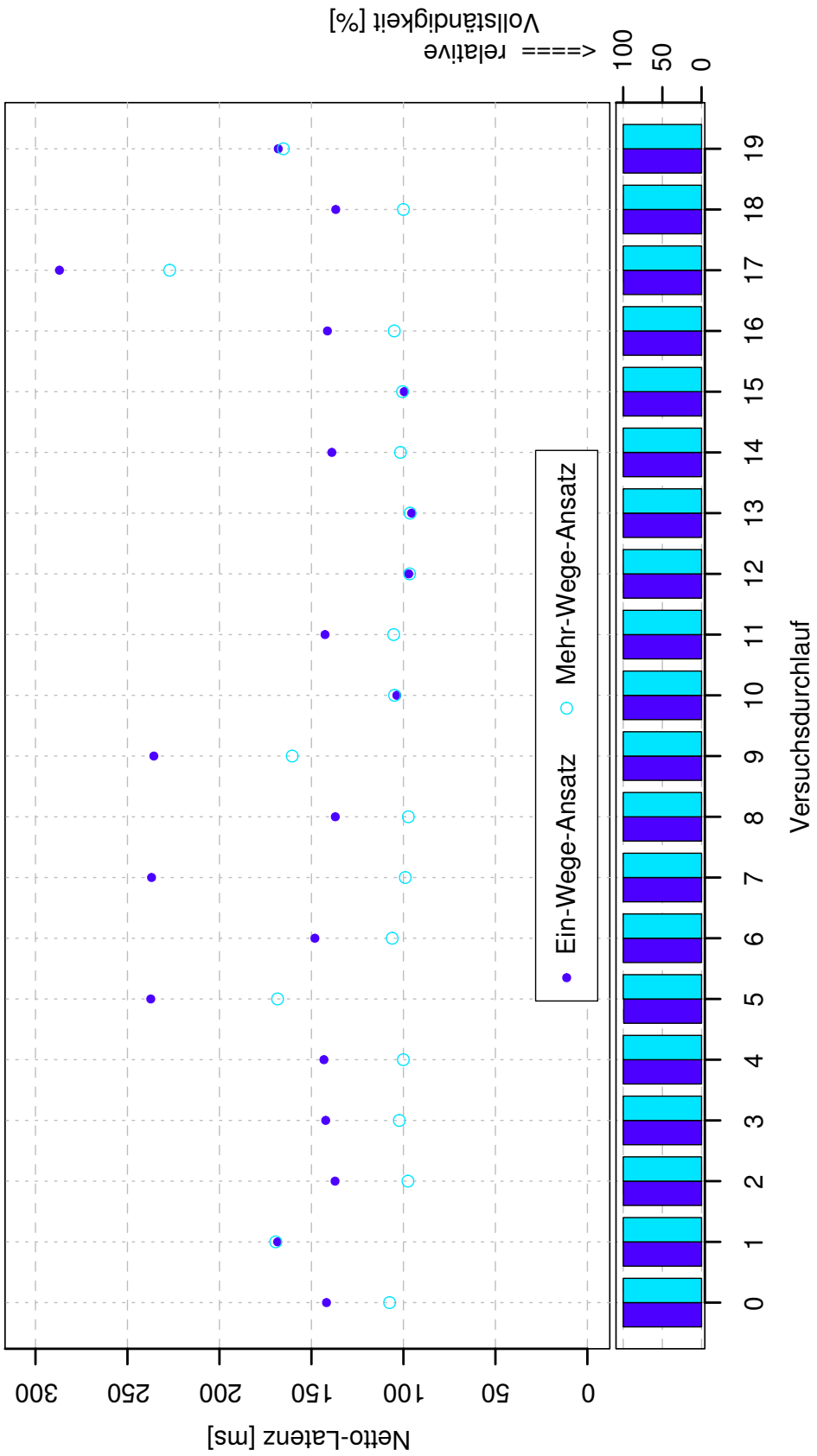


Abbildung B.33.:
Netto-Latenz für Versuch 9 für jeden Versuchsdurchlauf jeweils für die drei verglichenen Ansätze. Zur Orientierung ist unten die Vollständigkeit für den entsprechenden Versuch aufgetragen.

B.11. Versuch 10 – Horizontale Skalierbarkeit

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	variabel
Anzahl der Produzenten	1
Anzahl der Verarbeiter	1
Anzahl der Konsumenten	1
Anzahl zusätzlicher Verarbeiter	0
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	0 [m/s]
Länge der Pausen	∞
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	1400 [B]
Größe der Nachrichten des Verarbeiters	1400 [B]
Größe der Ankündigungen des Produzenten	100 [B]
Größe der Ankündigungen des Verarbeiters	100 [B]
Parameter der Simulation	
Anzahl der Experimente	100
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	50 [s] / 1000 [s]
Anzahl der zwischengespeicherten Nachrichten	20
Anzahl der in Abonnements referenzierten Nachrichten	20

Tabelle B.11.:
Übersicht über die Konfiguration des Testszenarios für Versuch 10. Die Gültigkeitsdauer der Ankündigungen beträgt für den Ein-Wege-Ansatz 50 [s] und für den Mehr-Wege-Ansatz 1000 [s] um vergleichbare Werte für die Vollständigkeit im Ausgangsbereich von 10 Geräten zu erreichen. Die Anzahl der Geräte wird zwischen 10 und 37 linear variiert.

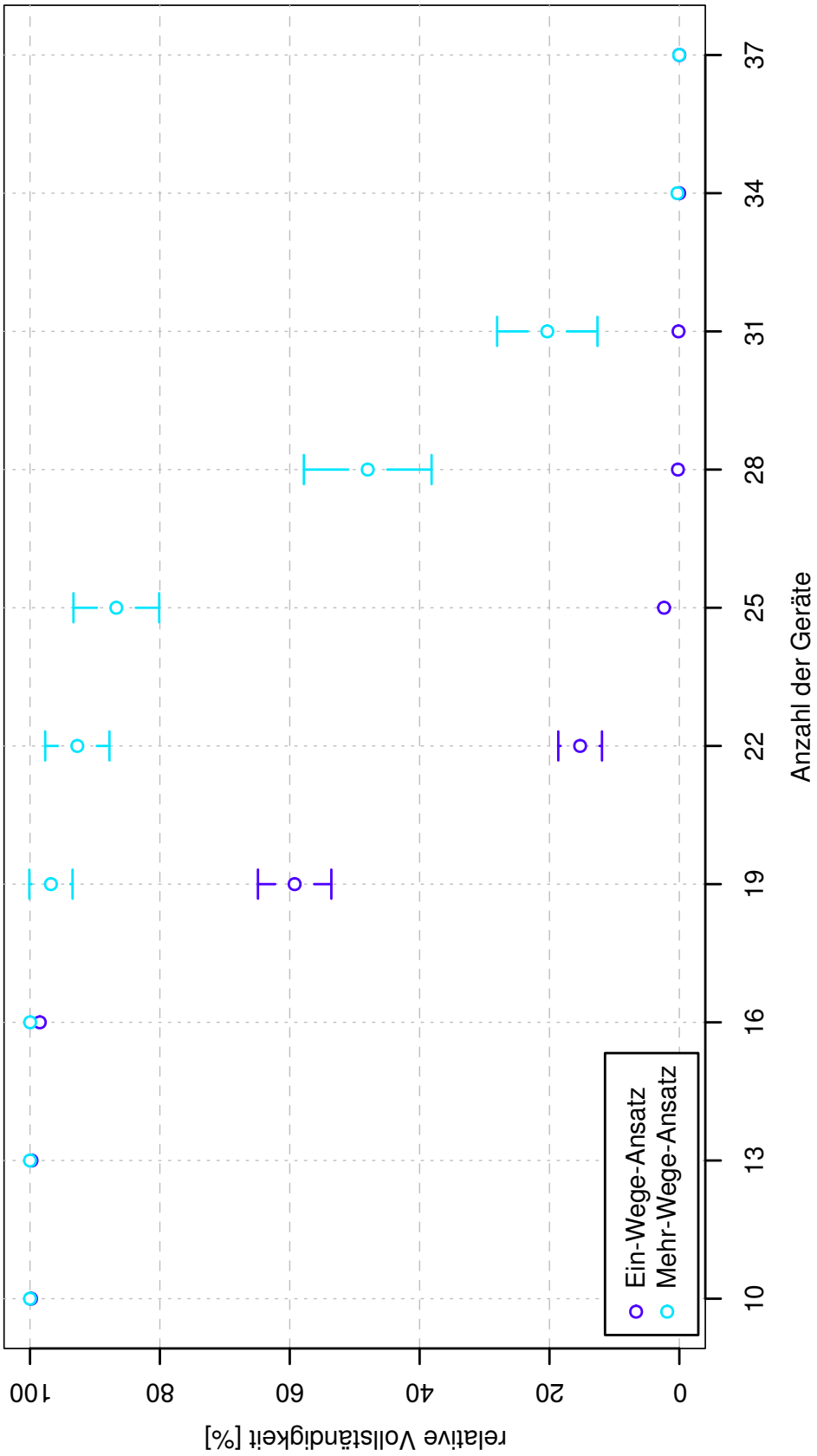


Abbildung B.34.:
Die erzielte Vollständigkeit in Abhängigkeit von der Anzahl der Geräte in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 10.

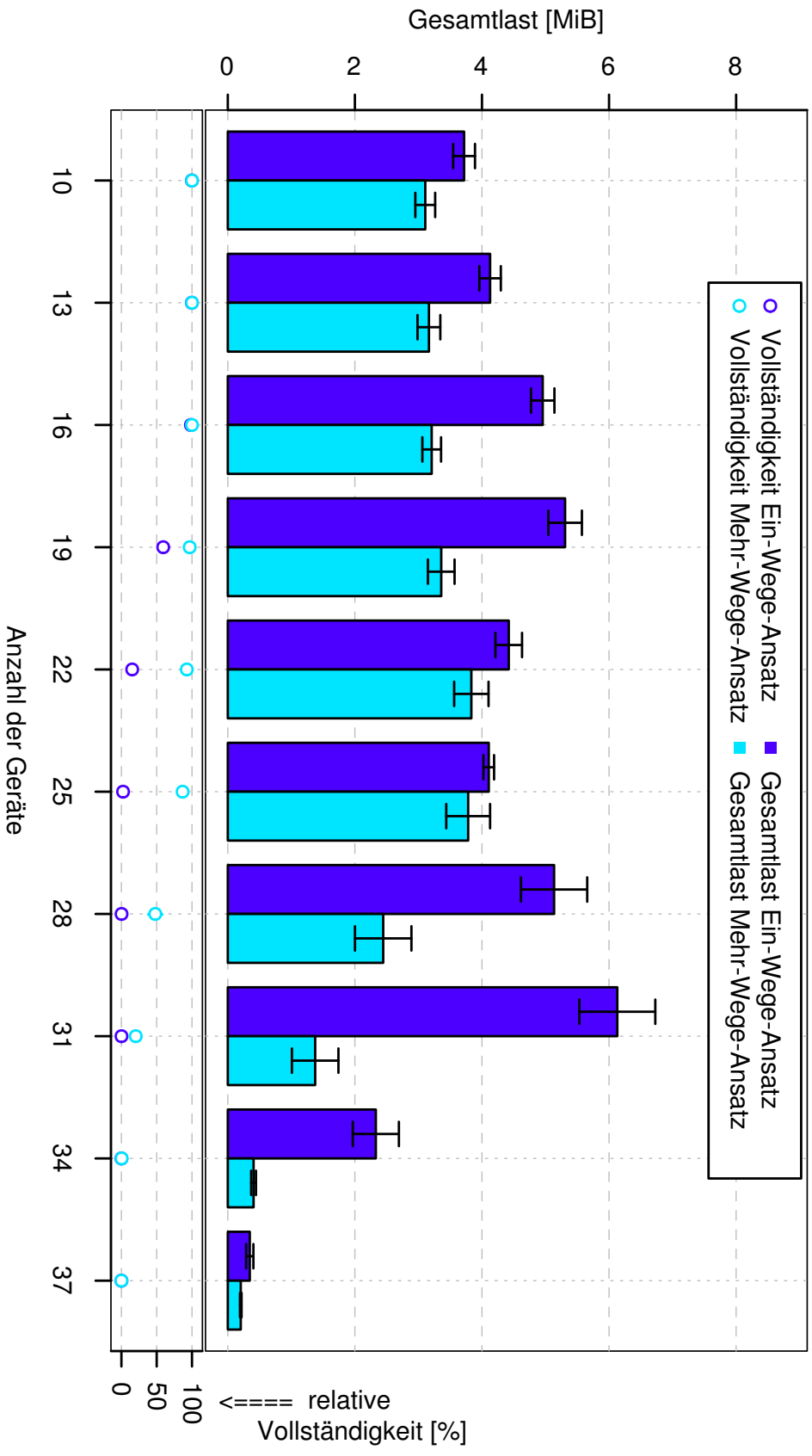


Abbildung B.35.:
Die im Versuch erzeugte Netzlast in Abhängigkeit von der Anzahl der Geräte in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 10. Zur Orientierung sind die Werte für die Vollständigkeit im unteren schmalen Diagramm nochmal aufgetragen.

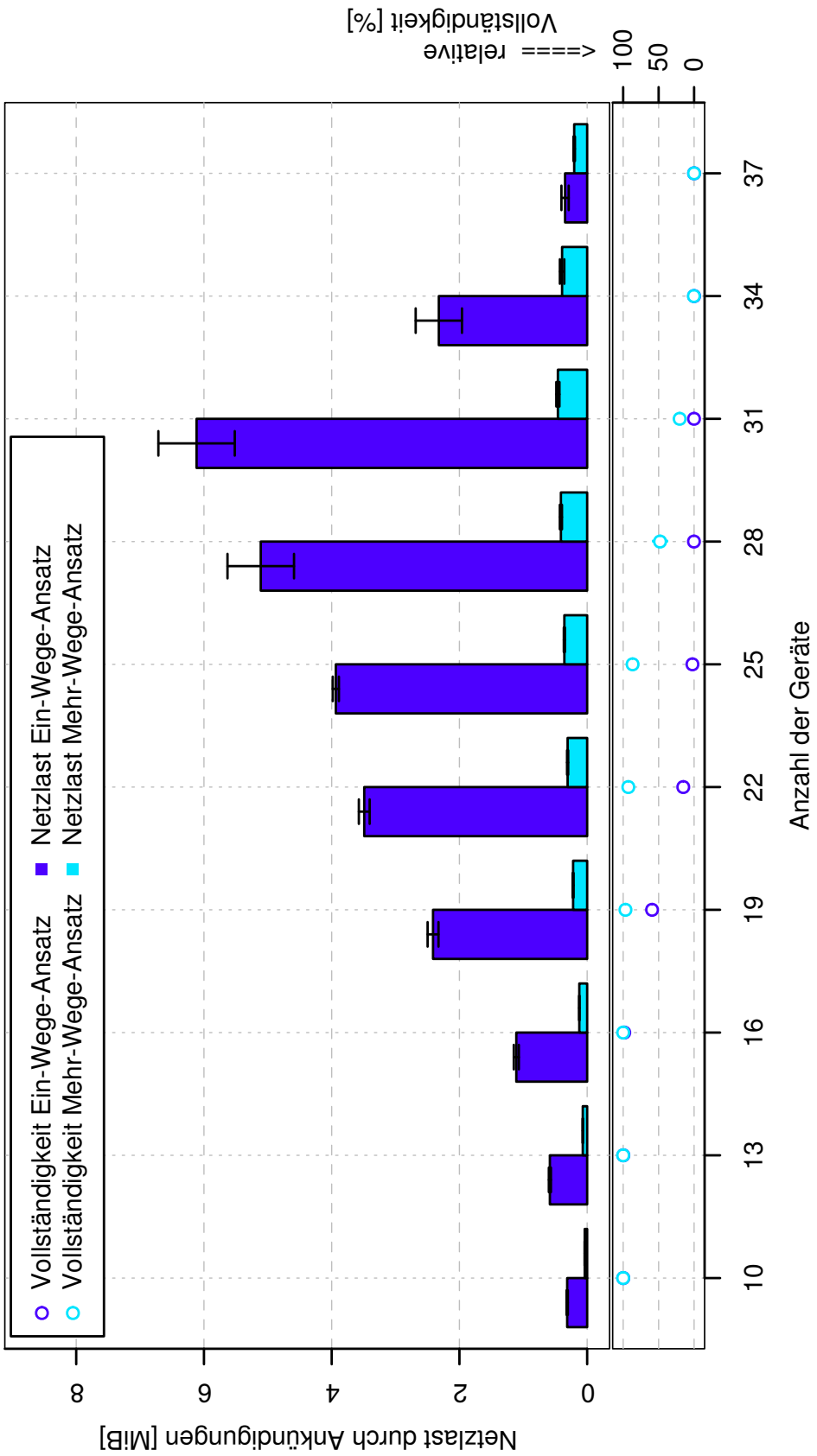


Abbildung B.36.:
Die durch Ankündigungen erzeugte Netzlast in Abhängigkeit von der Anzahl der Geräte in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 10. Zur Orientierung sind die Werte für die Vollständigkeit im unteren schmalen Diagramm nochmal aufgetragen.

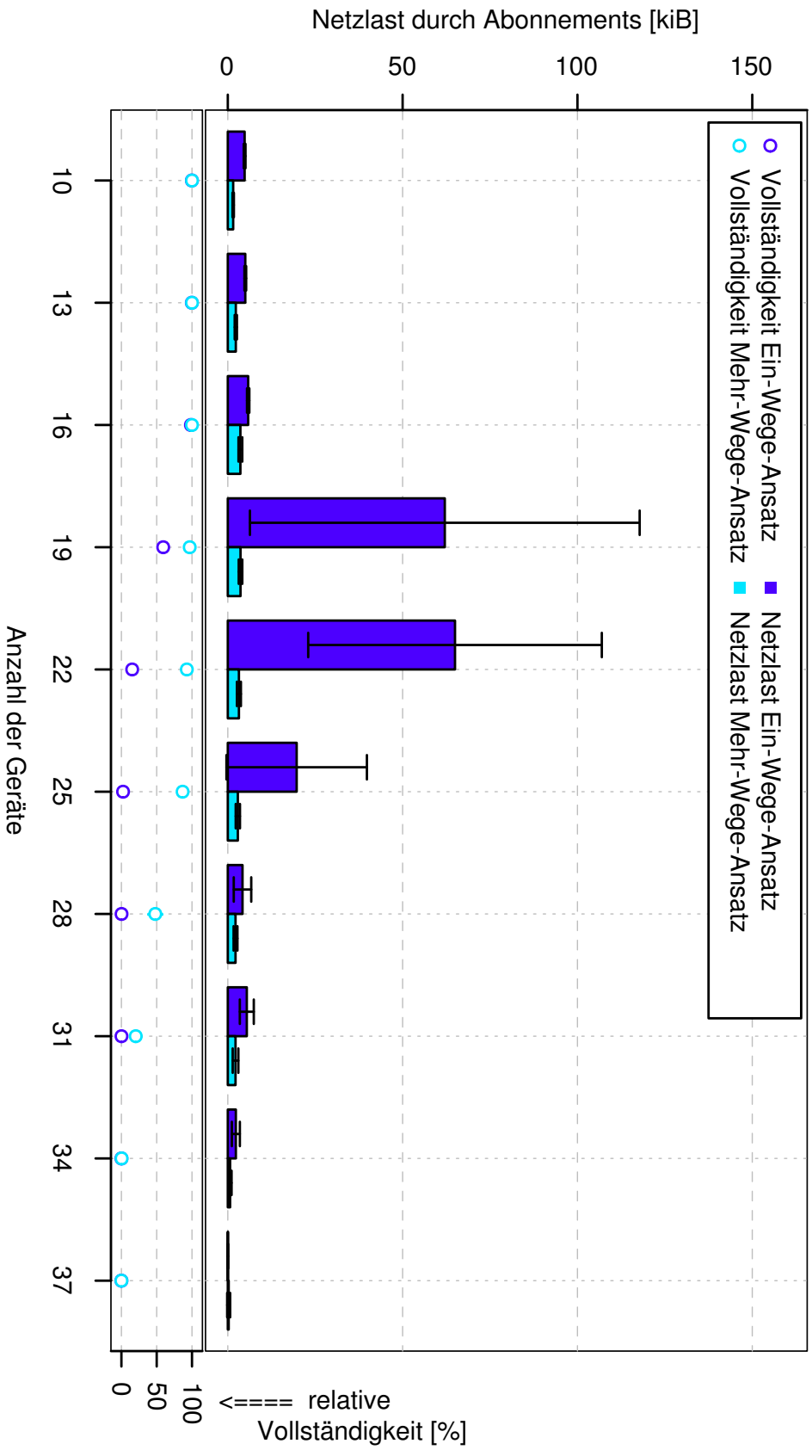


Abbildung B.37.:
Die durch Abonnements erzeugte Netzlast in Abhängigkeit von der Anzahl der Geräte in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 10. Zur Orientierung sind die Werte für die Vollständigkeit im unteren schmalen Diagramm nochmal aufgetragen.

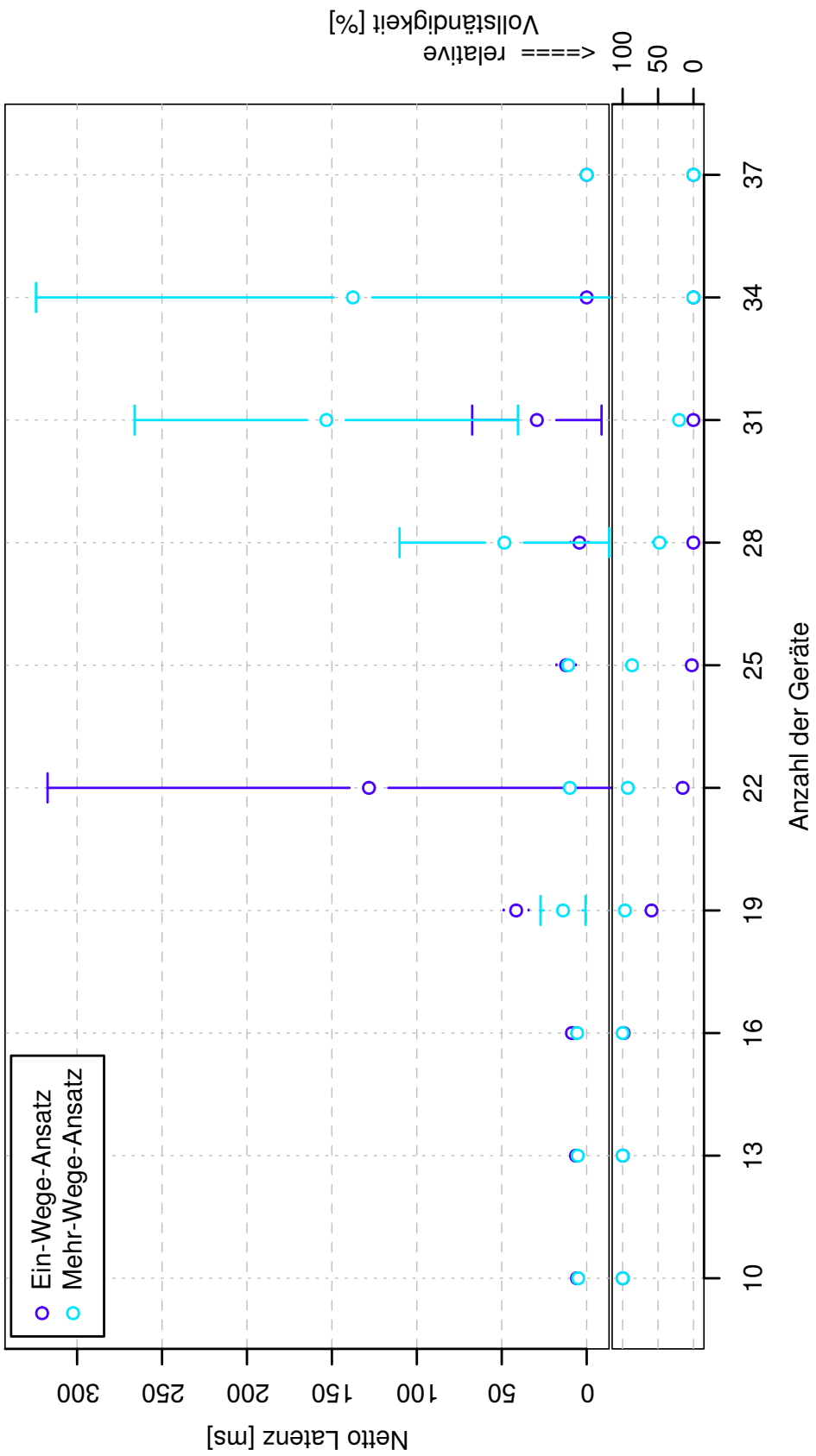


Abbildung B.38.:
Die Netto-Latenz in Abhängigkeit von der Anzahl der Geräte in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 10. Zur Orientierung sind die Werte für die Vollständigkeit im unteren schmalen Diagramm nochmal aufgetragen.

B.12. Versuch 11 – Vertikale Skalierbarkeit

Parameter	Wert
Größe der Topologie	
Anzahl der Geräte	10
Anzahl der Produzenten	variabel (p)
Anzahl der Verarbeiter	p
Anzahl der Konsumenten	p
Anzahl zusätzlicher Verarbeiter	p
Seitenlänge der Experimentalfläche	400 [m]
Grad der Veränderlichkeit der Topologie	
Geschwindigkeit der Geräte	0 [m/s]
Länge der Pausen	∞
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der Nachrichten pro Produzent	1000
Frequenz der Veröffentlichung pro Produzent	1 [Hz]
Größe der Nachrichten des Produzenten	1400 [B]
Größe der Nachrichten des Verarbeiters	1400 [B]
Größe der Ankündigungen des Produzenten	100 [B]
Größe der Ankündigungen des Verarbeiters	100 [B]
Parameter der Simulation	
Anzahl der Experimente	100
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	50 [s] / 1000 [s]
Anzahl der zwischengespeicherten Nachrichten	20
Anzahl der in Abonnements referenzierten Nachrichten	20

Tabelle B.12.:
Übersicht über die Konfiguration des Testszenarios für Versuch 11. Die Gültigkeitsdauer der Ankündigungen beträgt für den Ein-Wege-Ansatz 50 [s] und für den Mehr-Wege-Ansatz 1000 [s] um vergleichbare Werte für die Vollständigkeit im Ausgangsbereich von einer Quelle/Senke/Verarbeiter zu erreichen. Die Anzahl von Quellen/Senken/Verarbeitern wird linear zwischen 1 und 7 variiert.

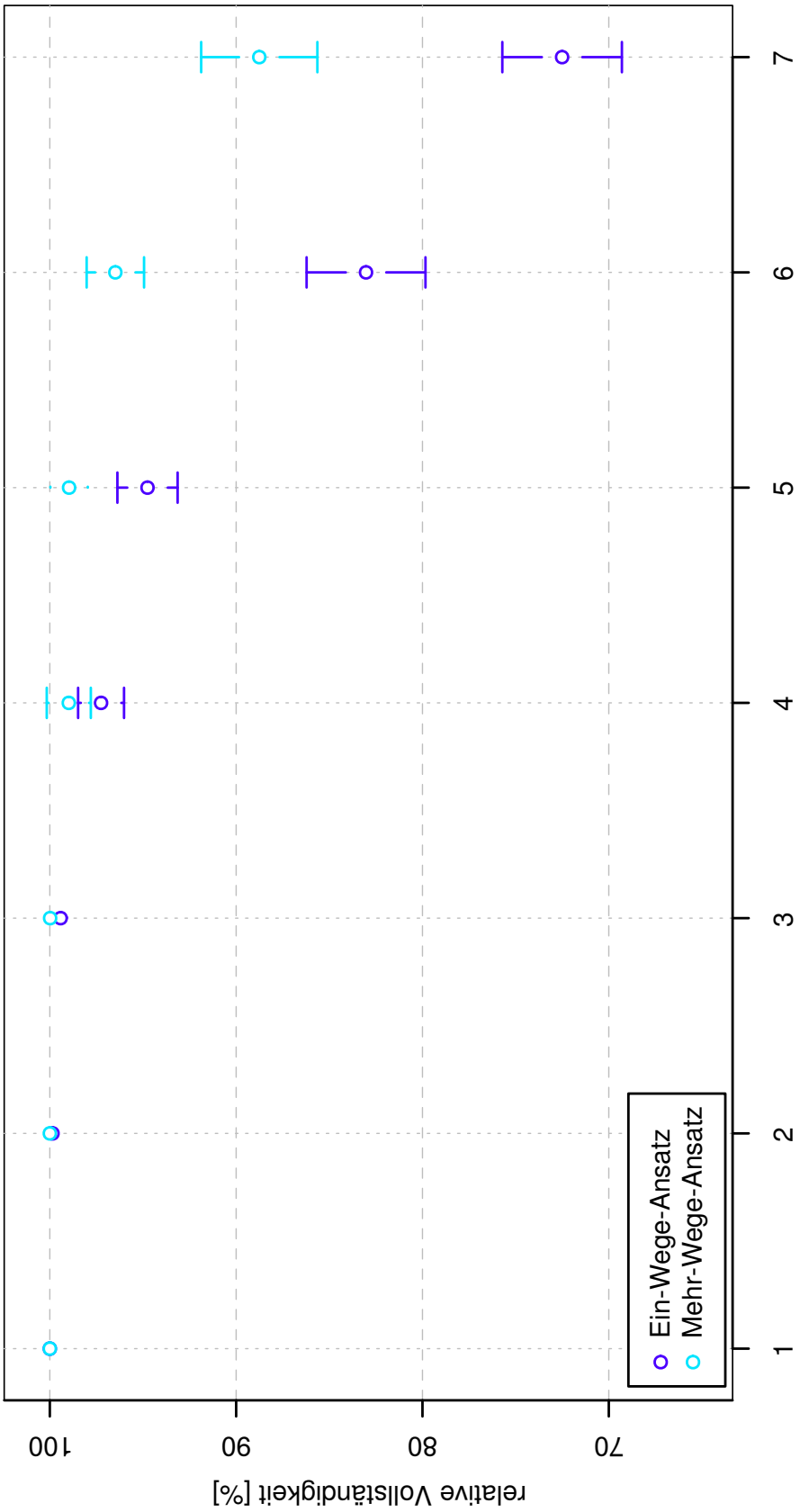


Abbildung B.39.:
Die erzielte Vollständigkeit in Abhängigkeit von der Anzahl der Produzenten in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 11.

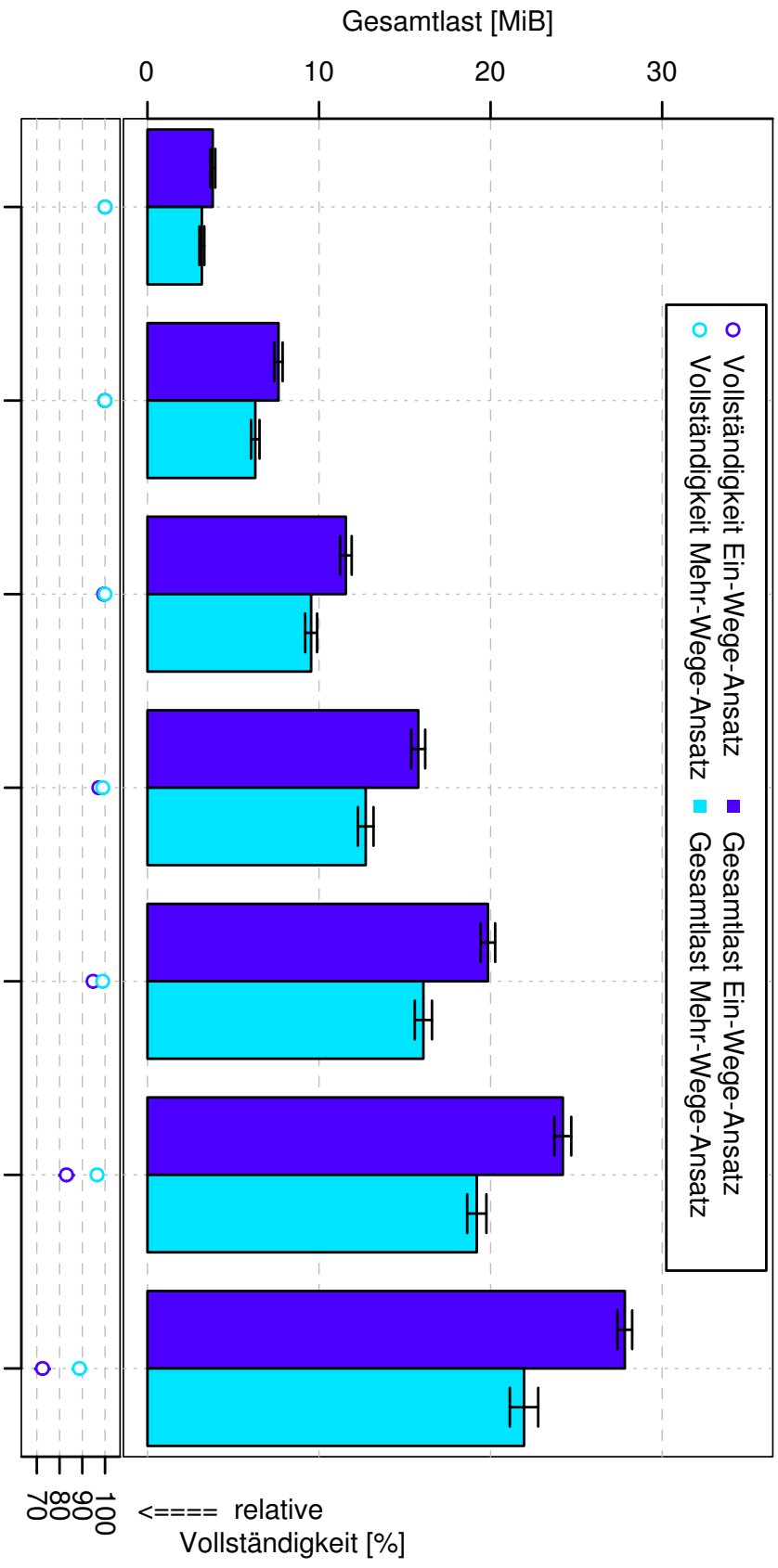


Abbildung B.40.:
Die im Versuch erzeugte Netzlast in Abhängigkeit von der Anzahl der Produzenten in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 11. Zur Orientierung sind die Werte für die Vollständigkeit im unteren schmalen Diagramm nochmal aufgetragen.

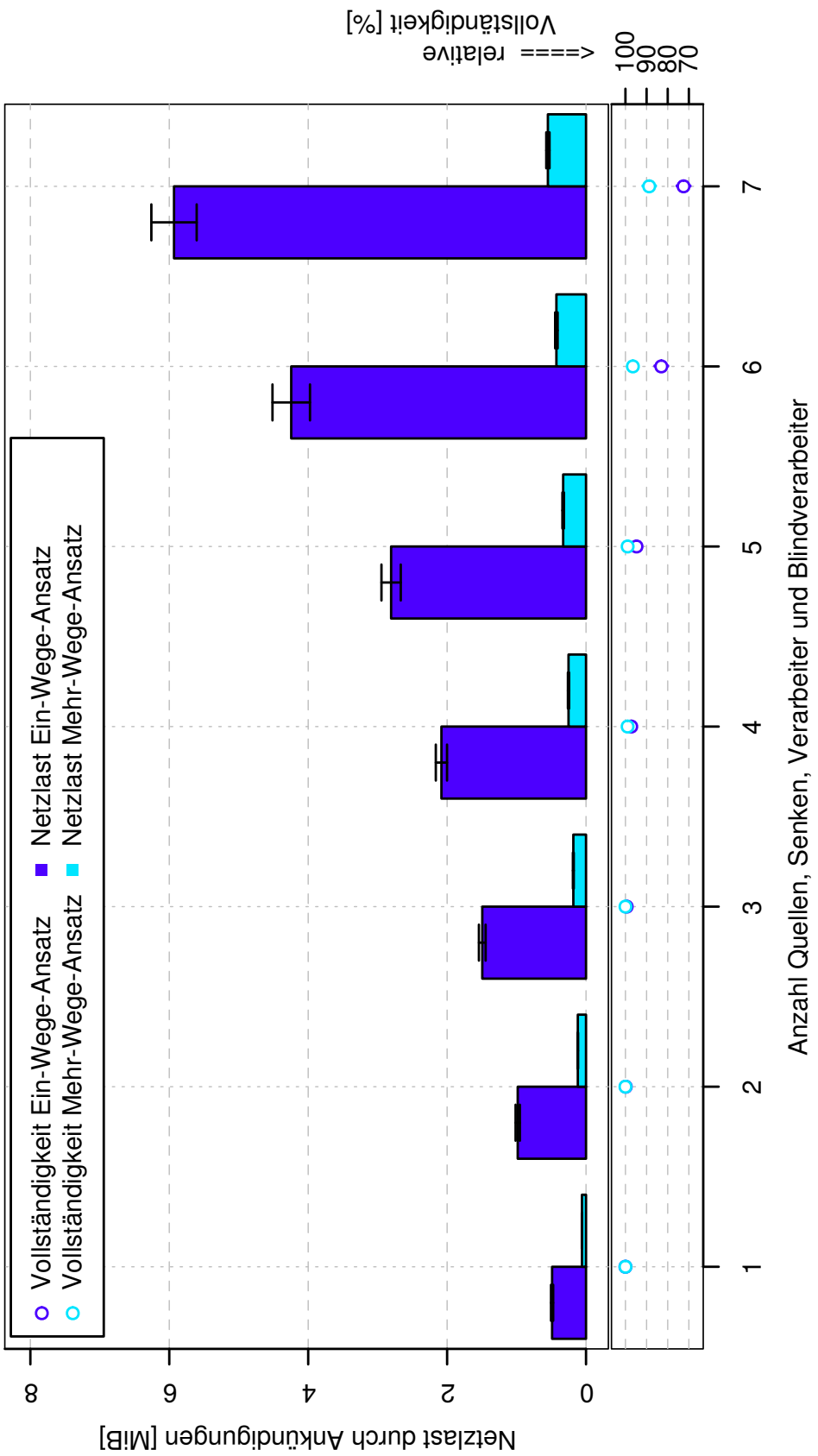


Abbildung B.41.:
Die durch Ankündigungen erzeugte Netzlast in Abhängigkeit von der Anzahl der Produzenten in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 11. Zur Orientierung sind die Werte für die Vollständigkeit im unteren schmalen Diagramm nochmal aufgetragen.

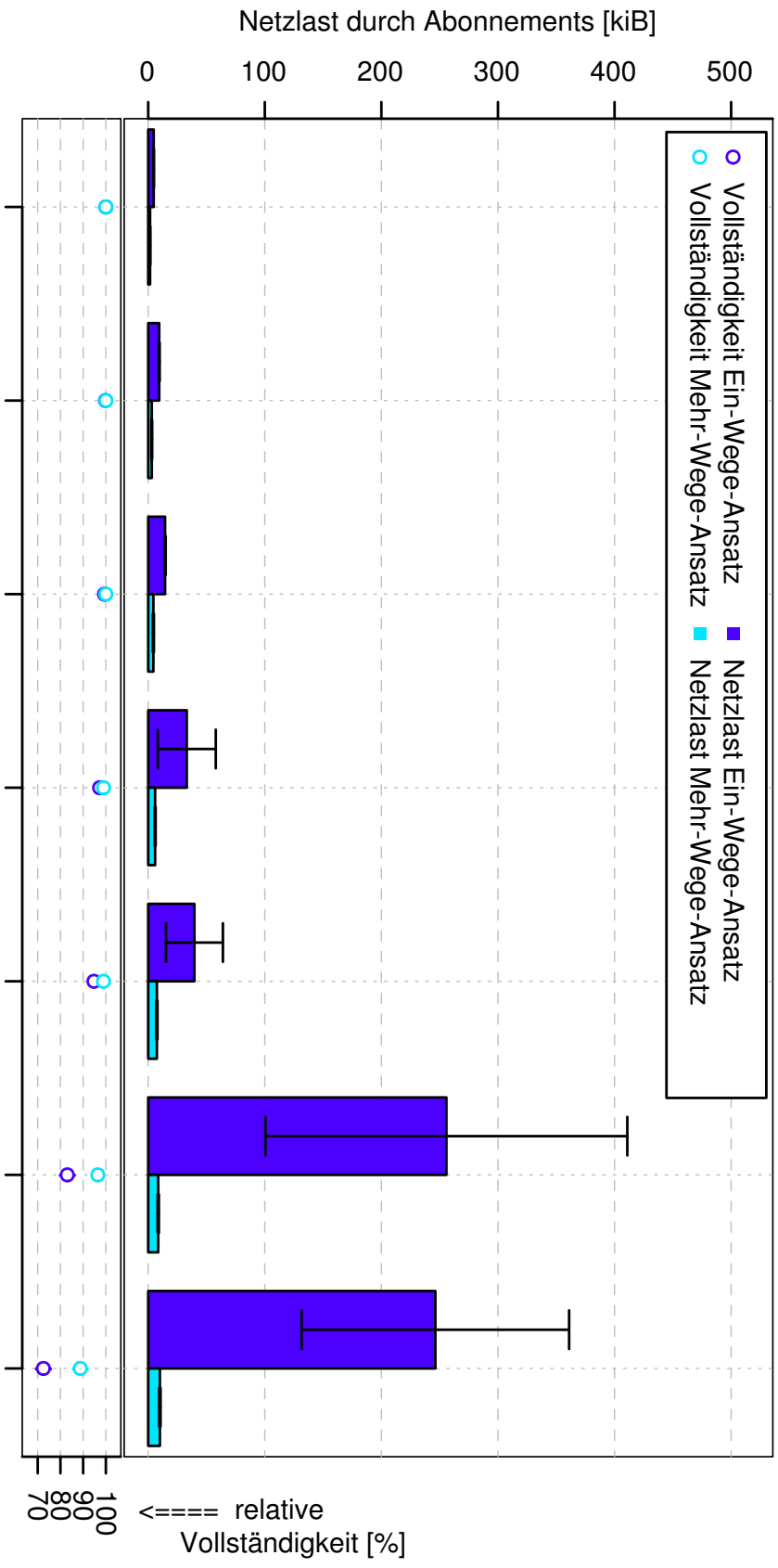


Abbildung B.42.:
Die durch Abonnements erzeugte Netzlast in Abhängigkeit von der Anzahl der Produzenten in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 11. Zur Orientierung sind die Werte für die Vollständigkeit im unteren schmalen Diagramm nochmal aufgetragen.

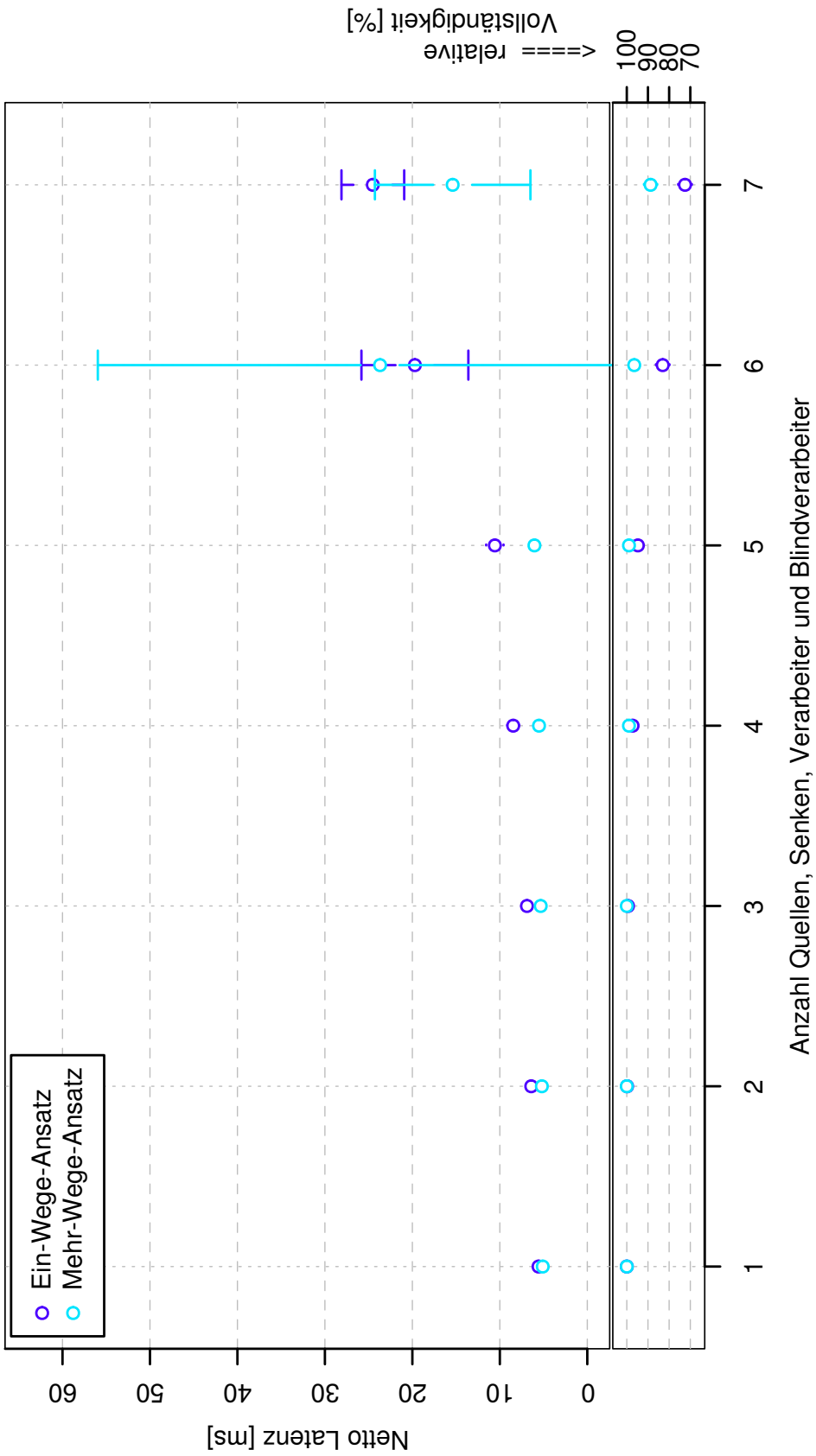


Abbildung B.43.:
Die Netto-Latenz in Abhängigkeit von der Anzahl der Produzenten in der Netzwerktopologie mit 95 % Konfidenzintervallen für Versuch 11. Zur Orientierung sind die Werte für die Vollständigkeit im unteren schmalen Diagramm nochmal aufgetragen.

B.13. Versuch 12 – Untersuchung in einem realitätsnahen Szenario

Eigenschaft	Wert
Größe der Topologie	
Anzahl der Produzenten von Fahrplaninformationen	1
Anzahl der Verarbeiter von Fahrplaninformationen	2
Anzahl der stationären Konsumenten von Fahrplaninformationen	2
Anzahl der mobilen Konsumenten von Fahrplaninformationen	2
Anzahl der Produzenten von Bilddaten	5
Anzahl der Verarbeiter von Bilddaten	2
Anzahl der stationären Konsumenten von Bilddaten	1
Anzahl der mobilen Konsumenten von Bilddaten	2
Größe und Anzahl der veröffentlichten Nachrichten	
Anzahl der veröffentlichten Fahrplaninformationen	60
Zeit zwischen der Veröffentlichung zweier Fahrplaninformationen	60 [s]
Größe der veröffentlichten Fahrplaninformationen	[10, 20) [kiB]
Verkleinerungsfaktor durch Verarbeitung der Fahrplaninformationen	0,1
Größe der Ankündigungen für Fahrplaninformationen	500 [B]
Anzahl der veröffentlichten Bilddaten pro Produzent	7000
Veröffentlichungsfrequenz der Bilddaten pro Produzent	2 [Hz]
Größe der veröffentlichten Bilddaten	[96, 144) [kiB]
Verkleinerungsfaktor durch Verarbeitung der Bilddaten	0,1
Größe der Ankündigungen für Bilddaten	600 [B]
Parameter der Simulation	
Anzahl der Experimente	100
Dauer eines Experiments	1200 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[100,105) [s]
Parameter der Simulation	
Anzahl der Experimente	100
Dauer eines Experiments	4400 [s]
Zeitpunkt der Veröffentlichung der ersten Nachricht	[399,400) [s]
Konfiguration der Algorithmen	
Gültigkeitsdauer der Ankündigungen	variabel
Anzahl der zwischengespeicherten Nachrichten	20
Anzahl der in Abonnements referenzierten Nachrichten	100

Tabelle B.13.:
Übersicht über die Eigenschaften des Bahnhofszenarios. Der Wert für die Frequenz der Bilddaten-Veröffentlichung ist der Durchschnittswert. Die tatsächlich gewählten Zeitpunkte schwanken normalverteilt mit einer Standardabweichung von 10 [ms] um ungewollte Nebeneffekte zu vermeiden.

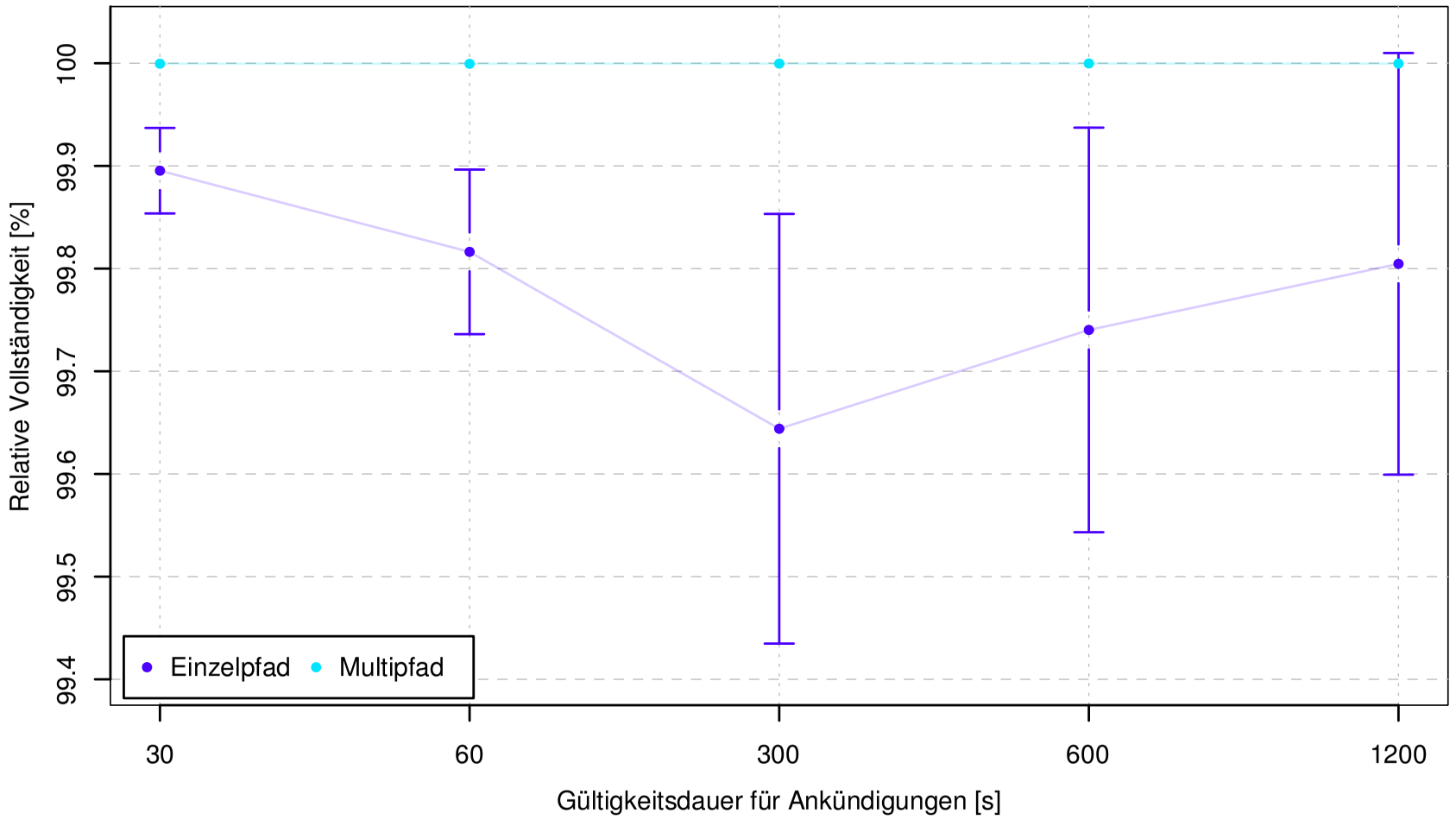


Abbildung B.44.:
Die erzielte Vollständigkeit beim Empfang von Bilddaten durch die stationäre Senke in Abhängigkeit von der Konfiguration der Algorithmen mit 95 % Konfidenzintervallen für Versuch 12. Die halbtransparenten Verbindungslinien dienen der besseren Erkennbarkeit der Zusammengehörigkeit der einzelnen Messwerte und tragen selbst keine Informationen.

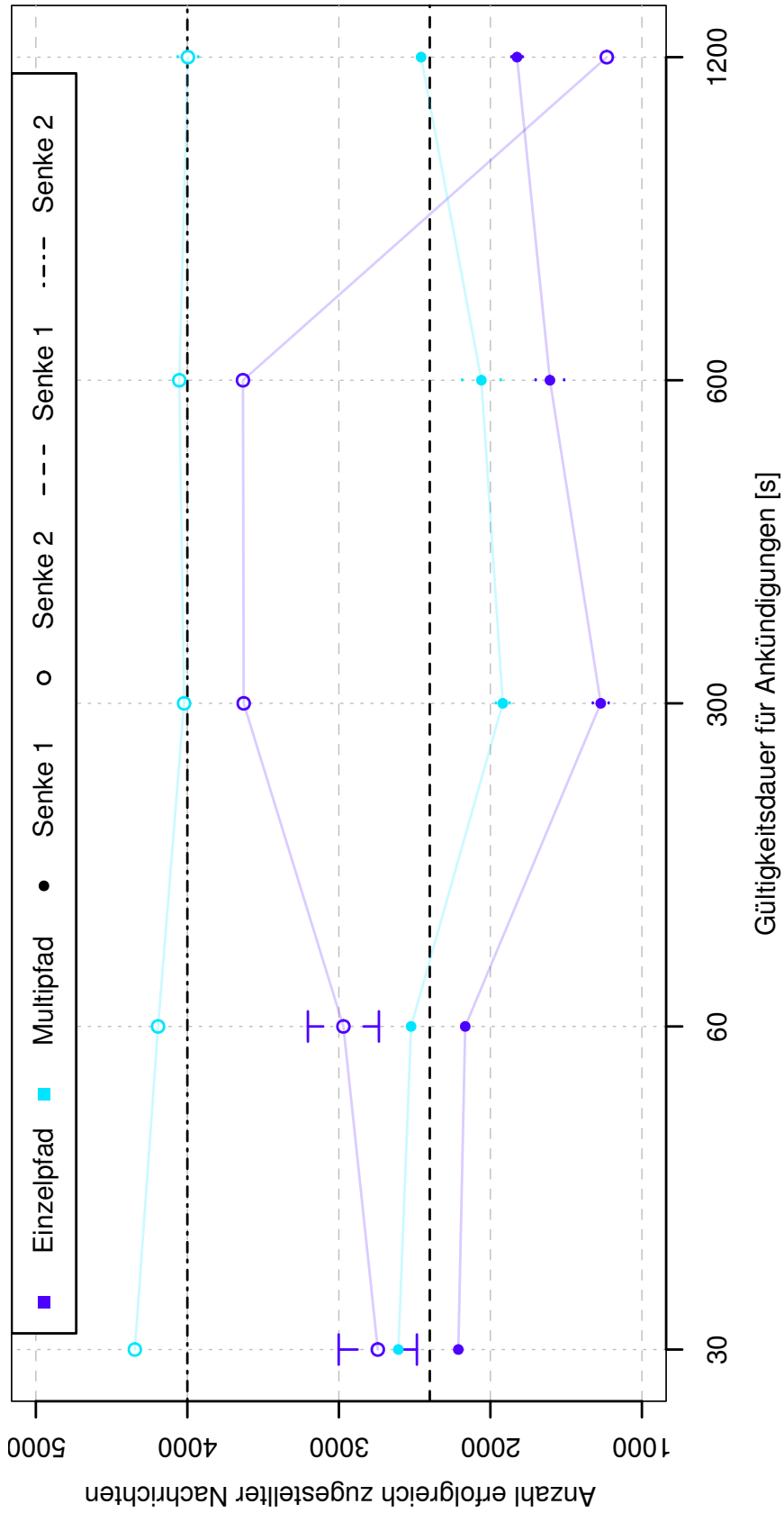


Abbildung B.45.:
Die erzielte Vollständigkeit beim Empfang von Bilddaten durch die beiden mobilen Senken in Abhängigkeit von der Konfiguration der Algorithmen mit 95 % Konfidenzintervallen für Versuch 12. Die halbtransparenten Verbindungslinien dienen der besseren Erkennbarkeit der Zusammengehörigkeit der einzelnen Messwerte und tragen selbst keine Informationen. Die schwarzen horizontalen Linien kennzeichnen die Anzahl der in der betrachteten Zeitspanne tatsächlich an die entsprechende Senke versandten Nachrichten. Durch die Zwischenspeicherung von Nachrichten auf den einzelnen Geräten können zusätzlich Nachrichten empfangen werden, die bereits vor dem betrachteten Zeitraum an die entsprechende Senke versandt wurden.

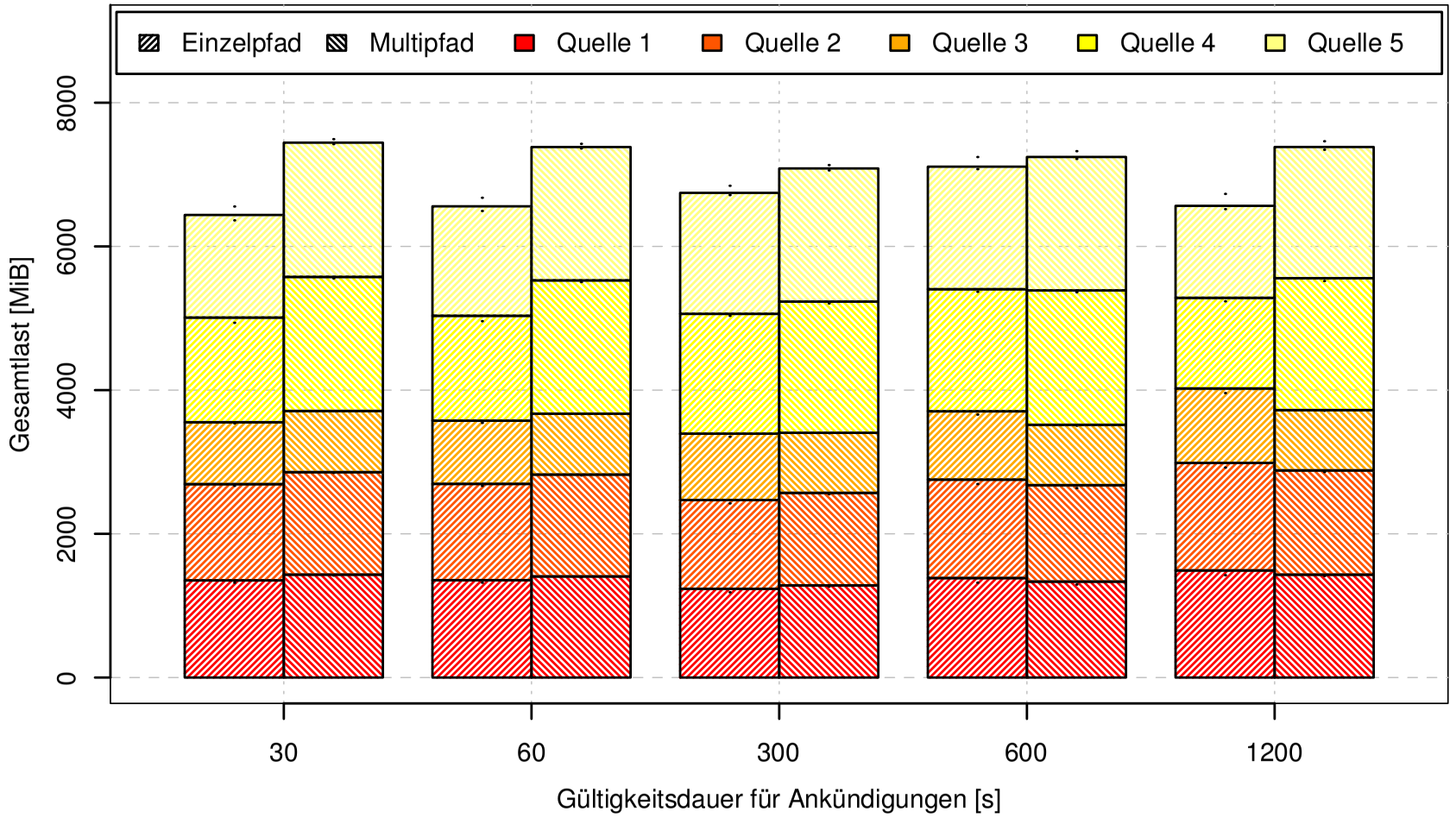


Abbildung B.46.:

Die insgesamt erzeugte Netzlast für die Übertragung der Bilddaten der fünf Quellen an die entsprechenden Senken für Versuch 12 in Abhängigkeit von der Versuchskonfiguration für beide Ansätze. Die 95 % Konfidenzintervalle werden durch die kleinen Punkte am oberen Ende des jeweiligen Blocks gekennzeichnet. Die Bilddaten von Quelle 3 werden von keiner mobilen Senke empfangen, daher ist die Netzlast hier geringer.

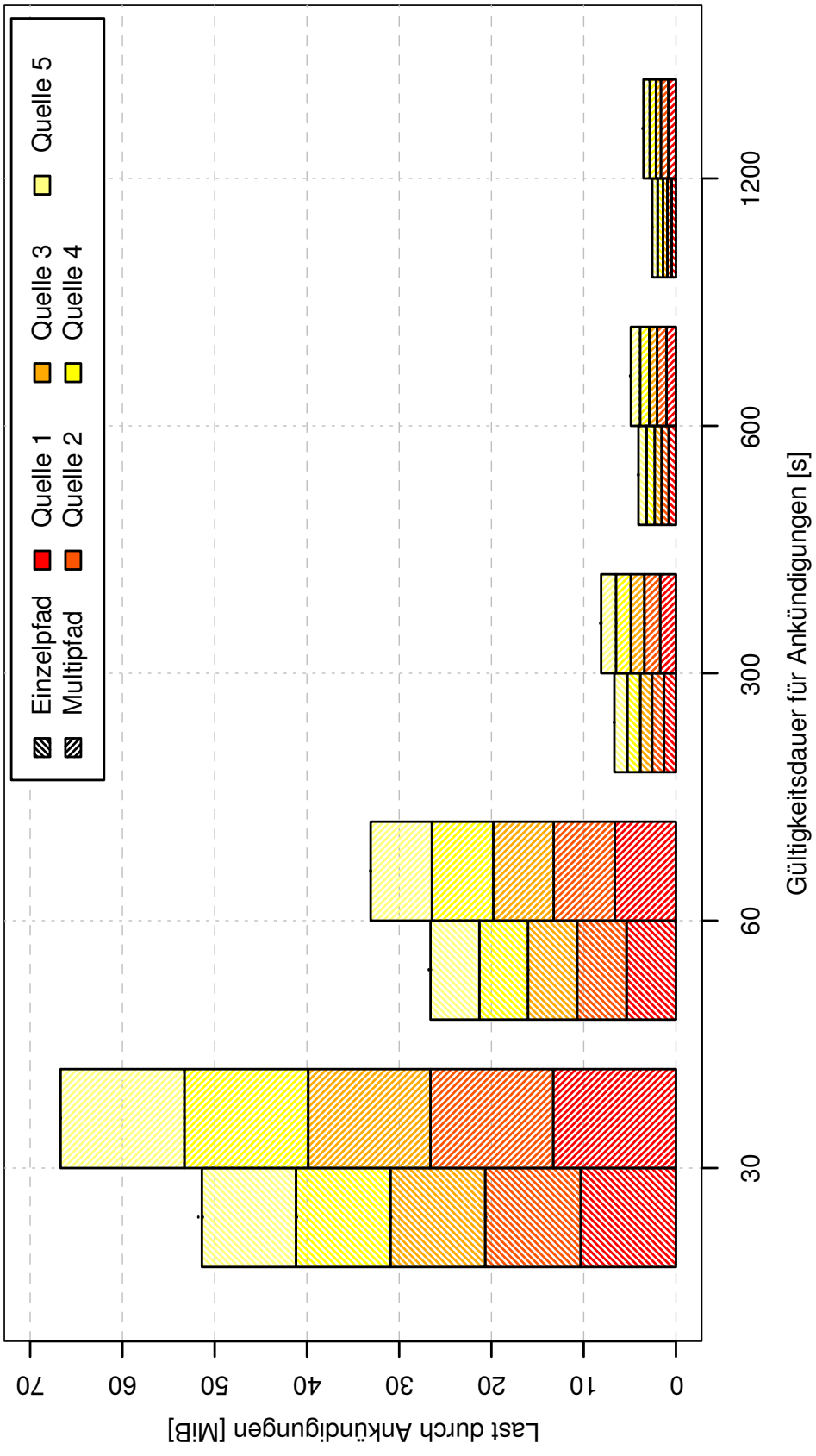


Abbildung B.47.:
Die durch versandte Ankündigungen erzeugte Netzlast für die Übertragung der Bilddaten der fünf Quellen an die entsprechenden Senken für Versuch 12 in Abhängigkeit von der Versuchskonfiguration für beide Ansätze. Die 95 % Konfidenzintervalle werden durch die kleinen Punkte am oberen Ende des jeweiligen Blocks gekennzeichnet.

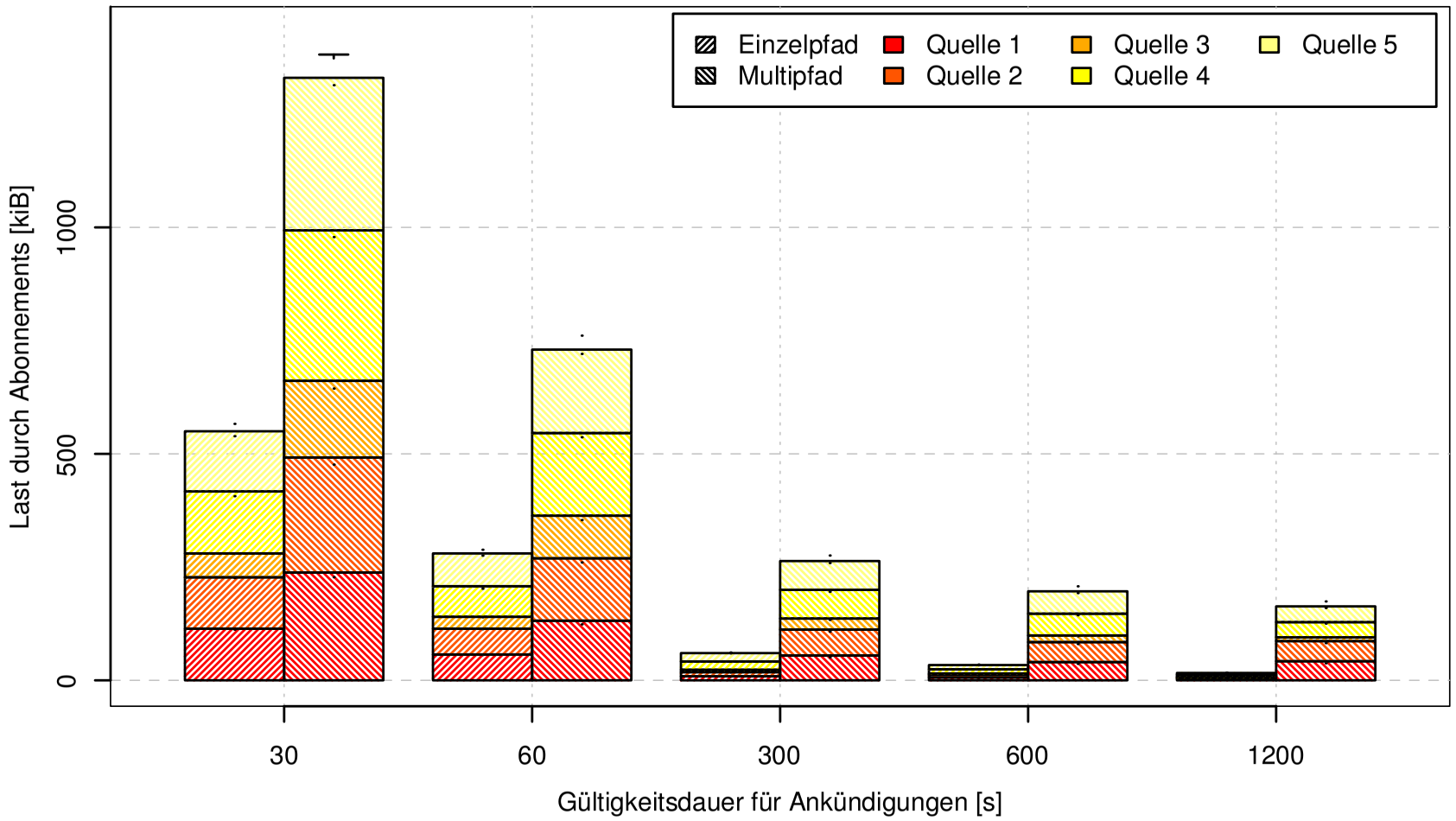


Abbildung B.48.:

Die durch versandte Abonnements erzeugte Netzlast für die Übertragung der Bilddaten der fünf Quellen an die entsprechenden Senken für Versuch 12 in Abhängigkeit von der Versuchskonfiguration für beide Ansätze. Die 95 % Konfidenzintervalle werden durch die kleinen Punkte und gegebenenfalls Linien am oberen Ende des jeweiligen Blocks gekennzeichnet. Die Bilddaten von Quelle 3 werden von keiner mobilen Senke empfangen, daher ist die Netzlast hier geringer.

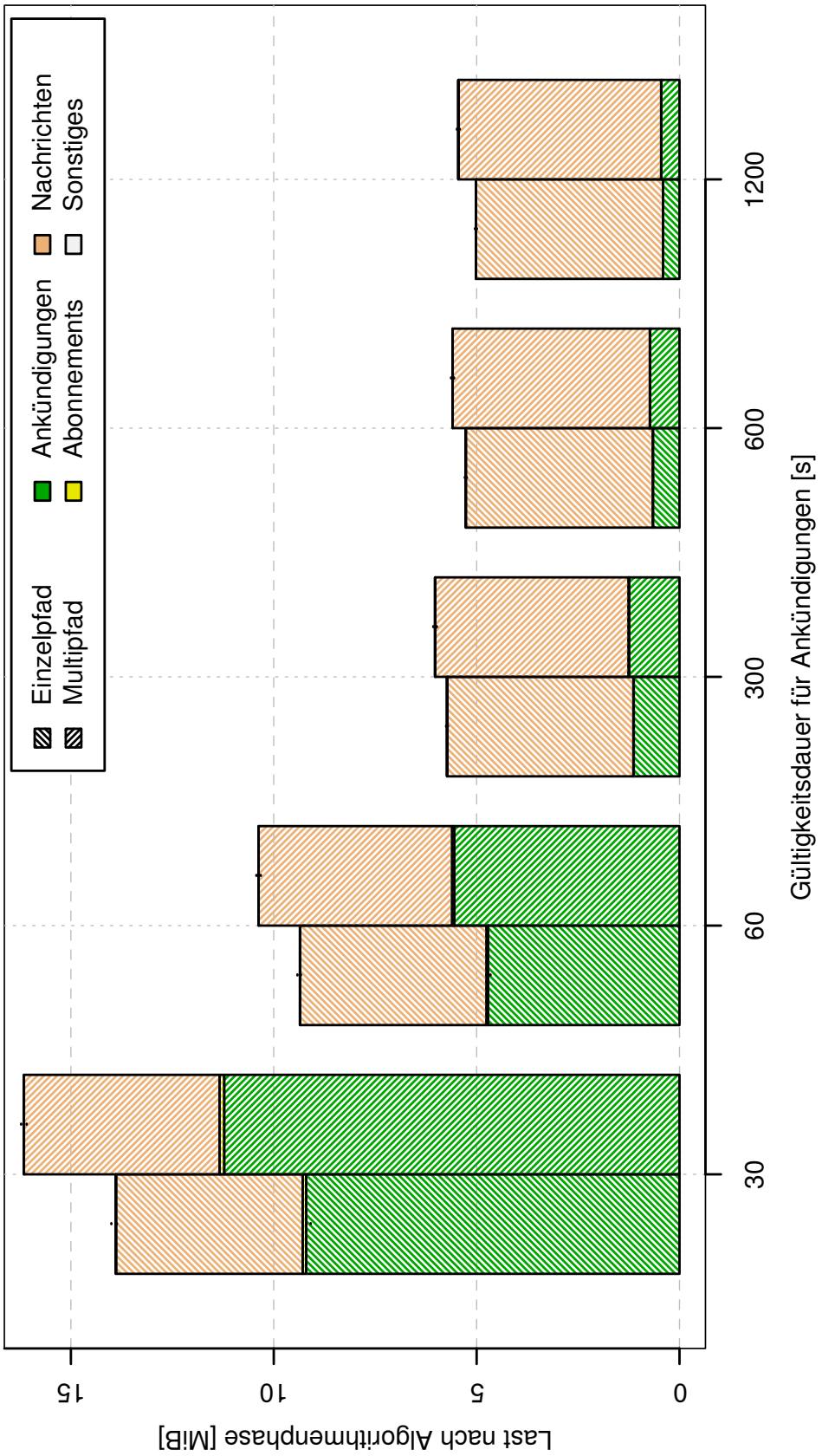


Abbildung B.49.:
Die erzeugte Netzlast für die Übertragung von Fahrplaninformationen für Versuch 12 in Abhängigkeit von der Versuchskonfiguration für beide Ansätze aufgeteilt auf die einzelnen Phasen der Algorithmen. Die 95 % Konfidenzintervalle werden durch die kleinen Punkte am oberen Ende des jeweiligen Blocks gekennzeichnet.

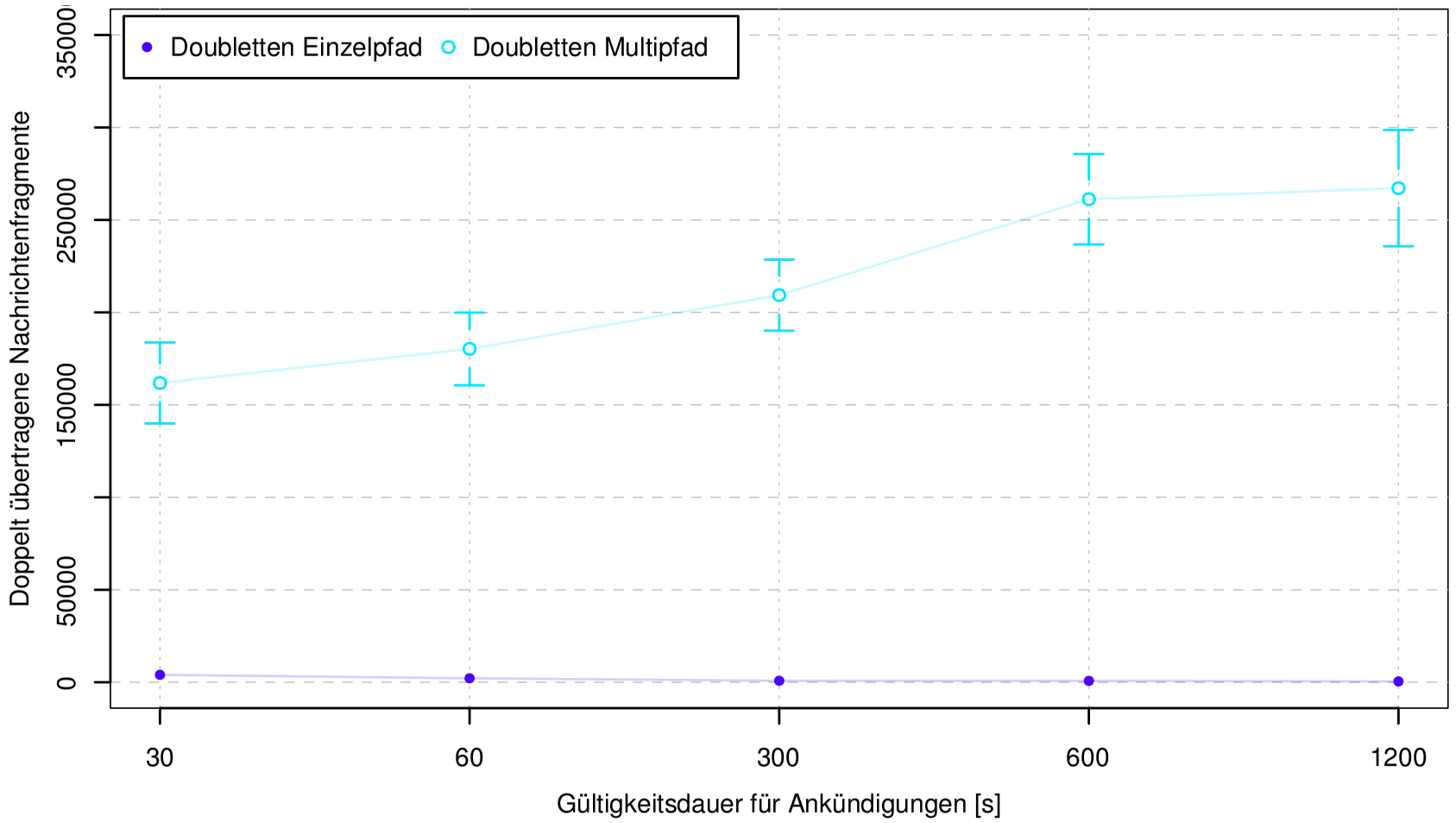


Abbildung B.50.:
Die Anzahl insgesamt doppelt übertragener Nachrichtenfragmente für Versuch 12 in Abhängigkeit von der Versuchskonfiguration für beide Ansätze mit 95 % Konfidenzintervallen. Die halbtransparenten Verbindungslinien dienen der besseren Erkennbarkeit der Zusammengehörigkeit der einzelnen Messwerte und tragen selbst keine Informationen.

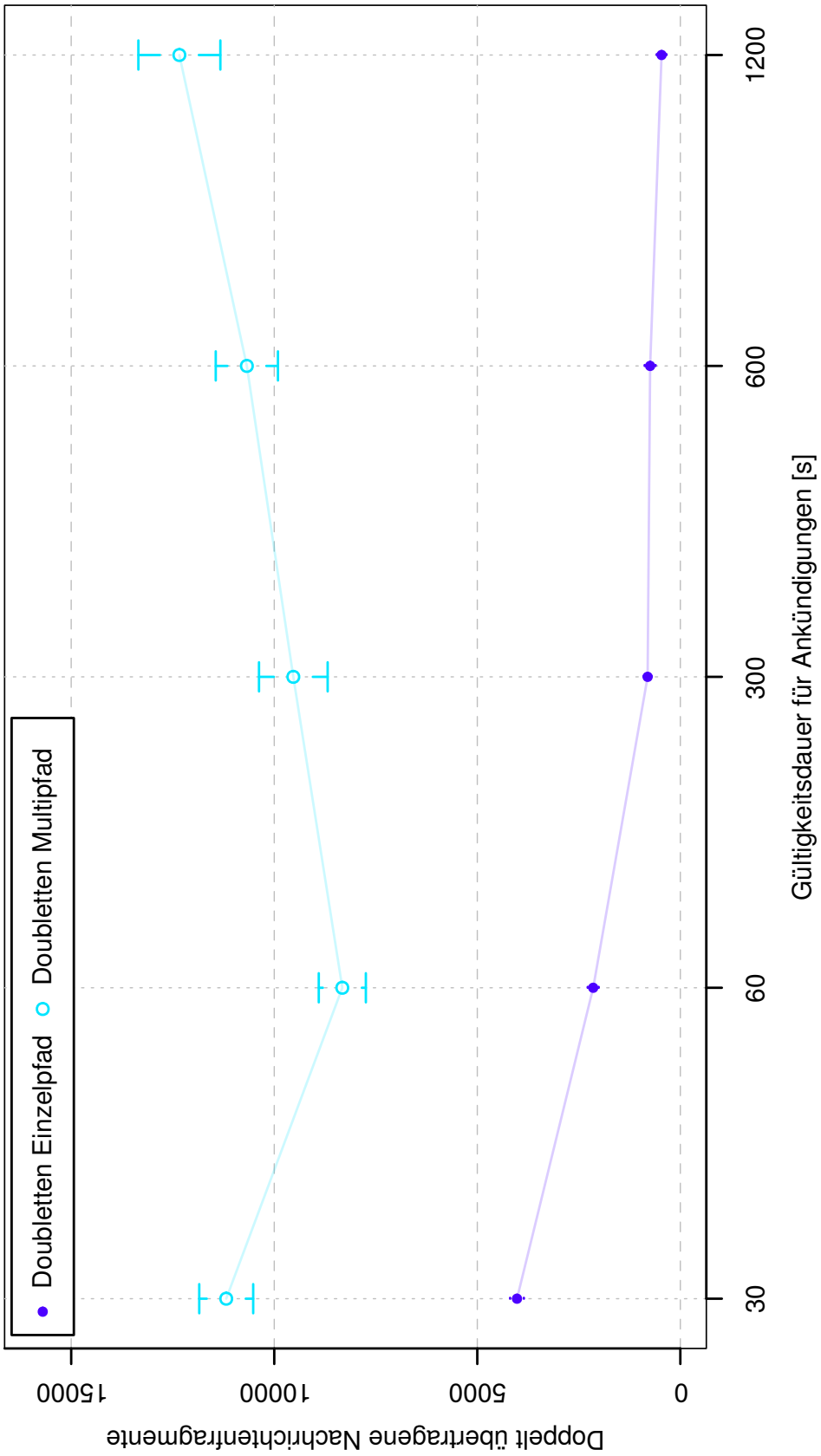


Abbildung B.51.:
Die Anzahl doppelt übertragener Nachrichtenfragmente für Versuch 12 in Abhängigkeit von der Versuchskonfiguration für beide Ansätze bereinigt um die durch die in Abschnitt 6.6.2 beschriebene Fehlfunktion mehrfach übertragenen Nachrichten mit 95 % Konfidenzintervallen. Die halbtransparenten Verbindungslinien dienen der besseren Erkennbarkeit der Zusammengehörigkeit der einzelnen Messwerte und tragen selbst keine Informationen.

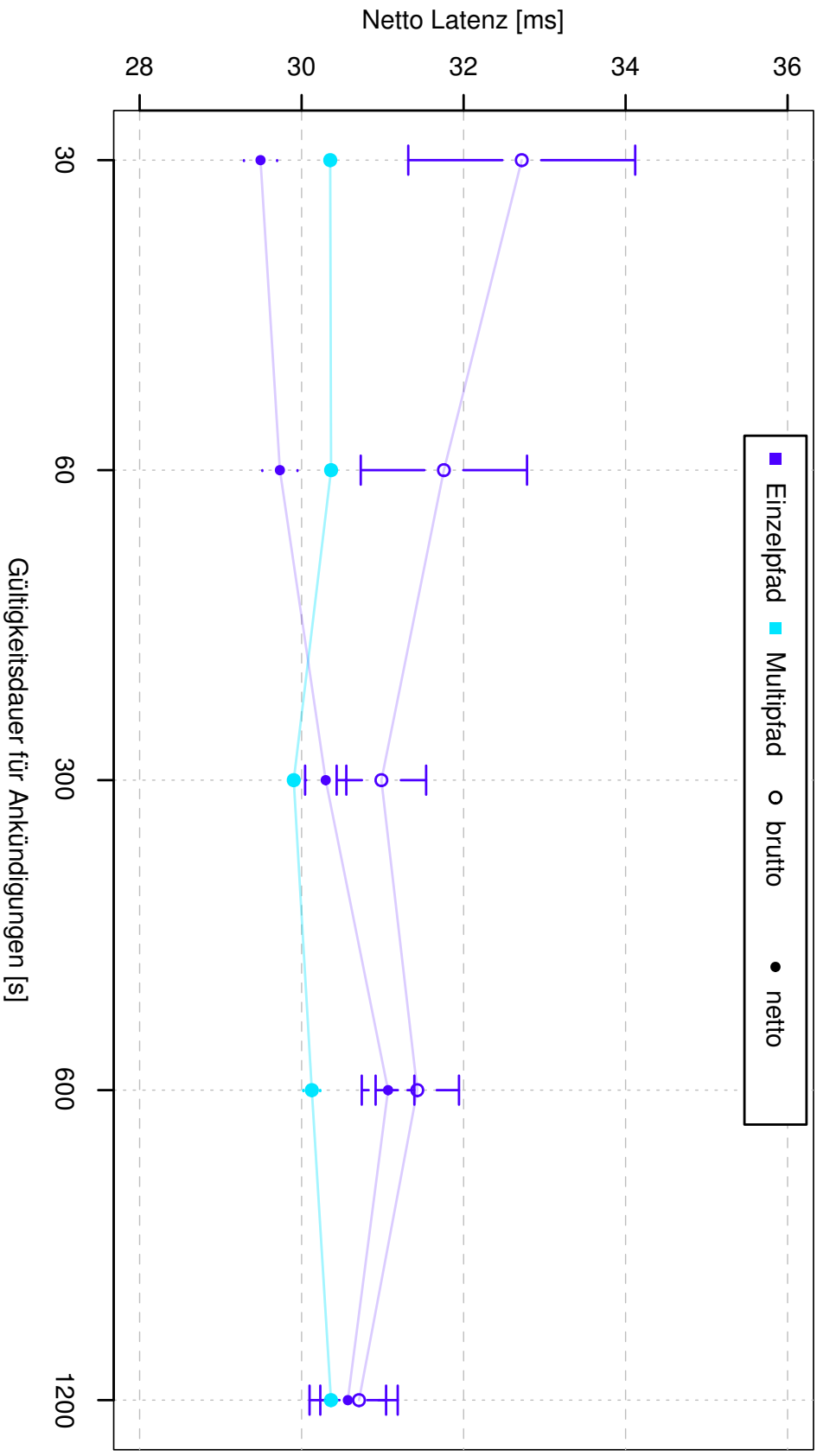


Abbildung B.52.:
Die Netto-Latenz für die Übertragung von Bilddaten an stationäre Senken in Abhängigkeit von der Versuchskonfiguration für Versuch 12 mit 95 % Konfidenzintervallen. Die halbtransparenten Verbindungslinien dienen der besseren Erkennbarkeit der Zusammengehörigkeit der einzelnen Messwerte und tragen selbst keine Informationen.

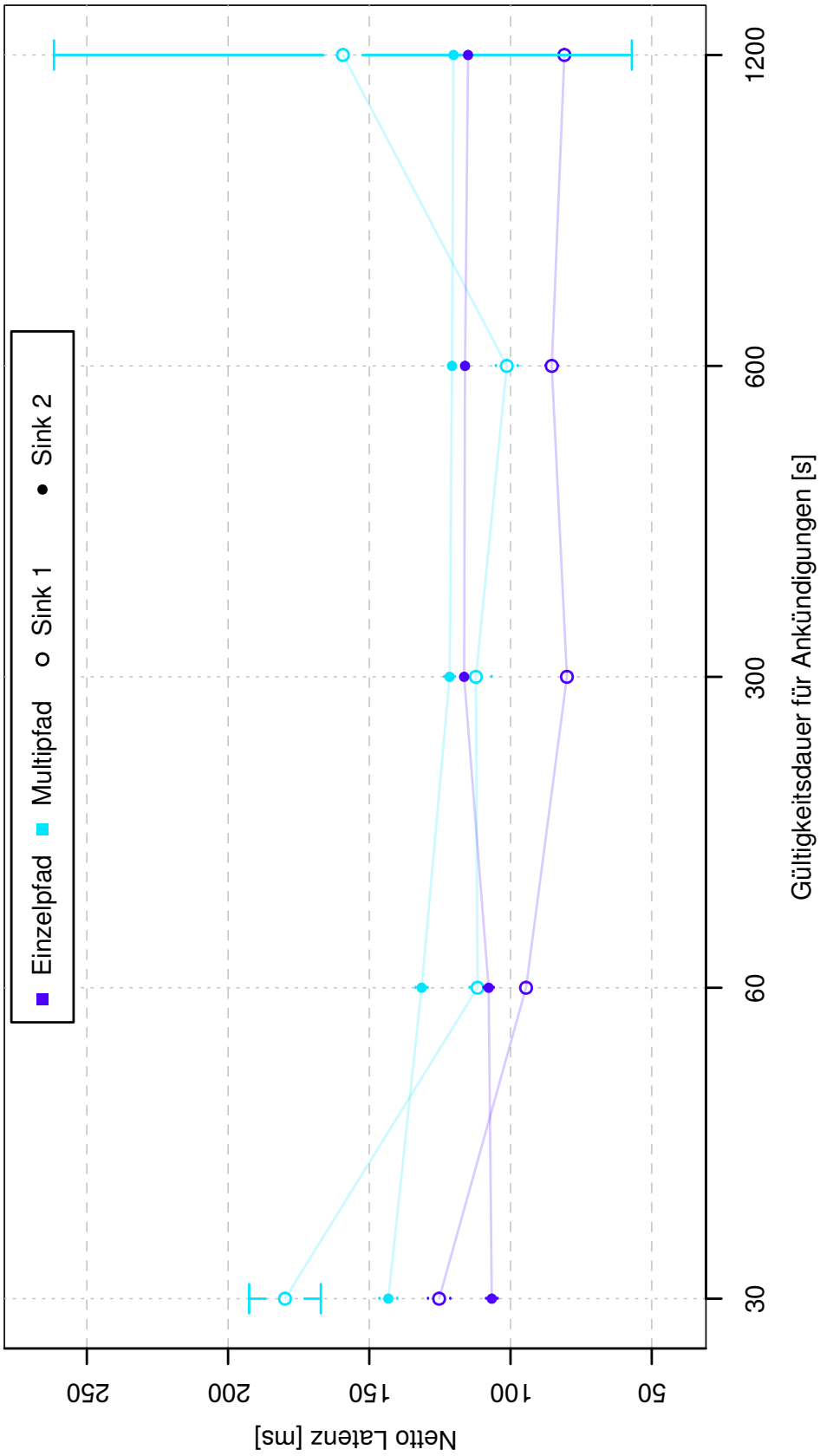


Abbildung B.53.:
Die Netto-Latenz für die Übertragung von Bilddaten an mobile Senken in Abhängigkeit von der Versuchskonfiguration für Versuch 12 mit 95 % Konfidenzintervallen. Die halbtransparenten Verbindungslinien dienen der besseren Erkennbarkeit der Zusammengehörigkeit der einzelnen Messwerte und tragen selbst keine Informationen.

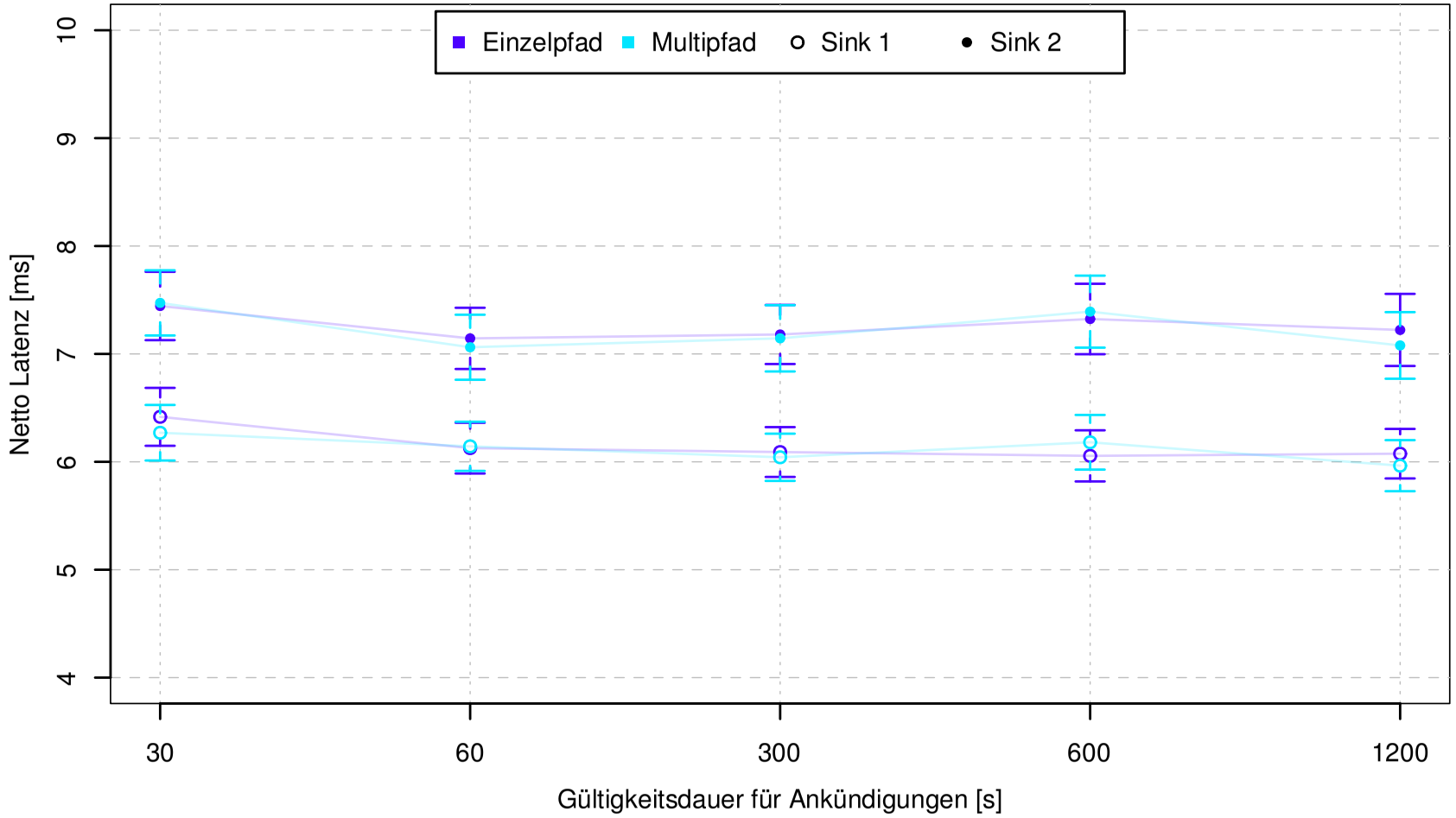


Abbildung B.54.:
Die Netto-Latenz für die Übertragung von Fahrplaninformationen an stationäre Senken in Abhängigkeit von der Versuchskonfiguration für Versuch 12 mit 95 % Konfidenzintervallen. Die halbtransparenten Verbindungslinien dienen der besseren Erkennbarkeit der Zusammengehörigkeit der einzelnen Messwerte und tragen selbst keine Informationen.

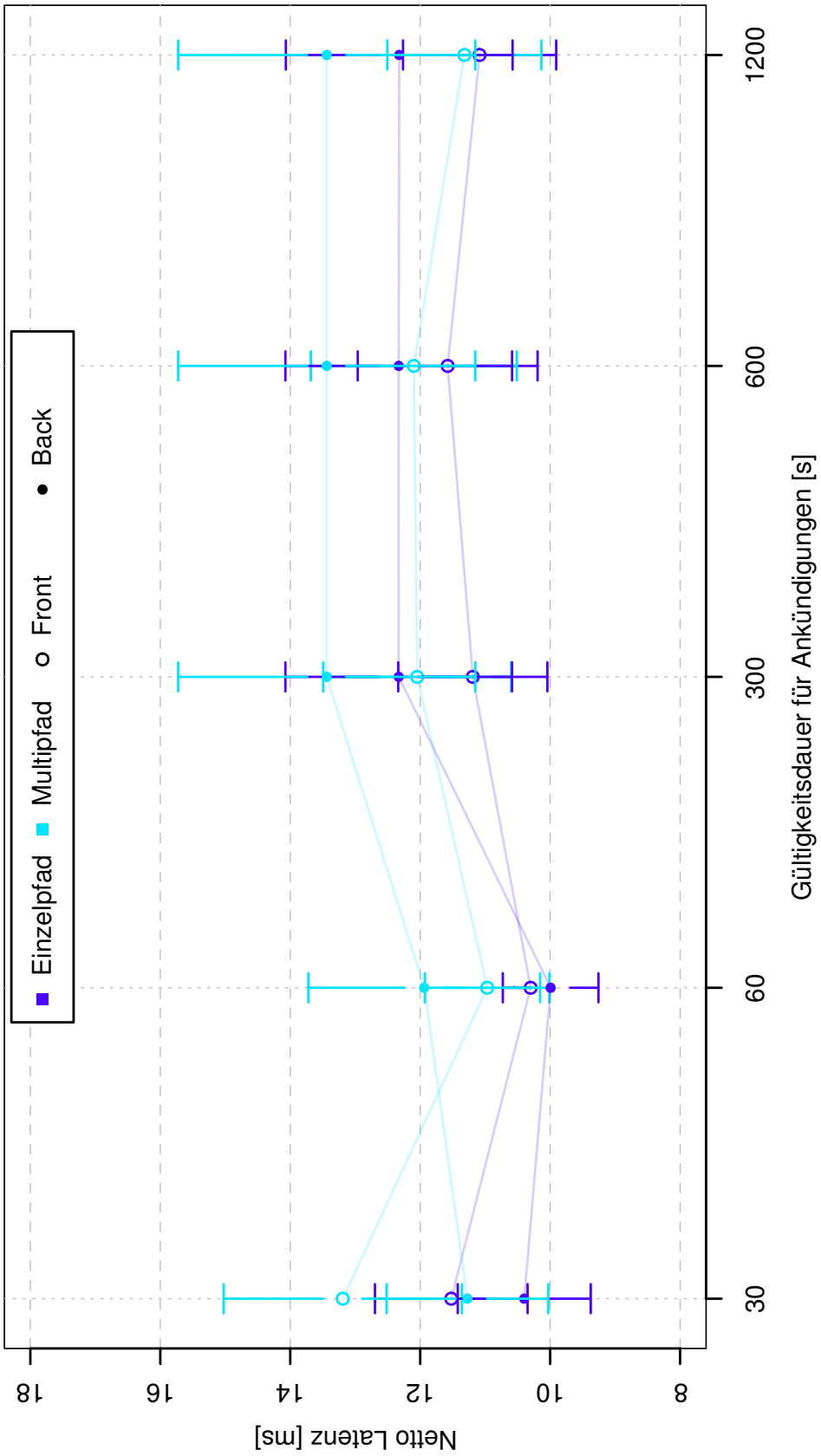


Abbildung B.55.:
Die Netto-Latenz für die Übertragung von Fahrplaninformationen an mobile Senken in Abhängigkeit von der Versuchskonfiguration für Versuch 12 mit 95 % Konfidenzintervallen. Die halbtransparenten Verbindungslinien dienen der besseren Erkennbarkeit der Zusammengehörigkeit der einzelnen Messwerte und tragen selbst keine Informationen.

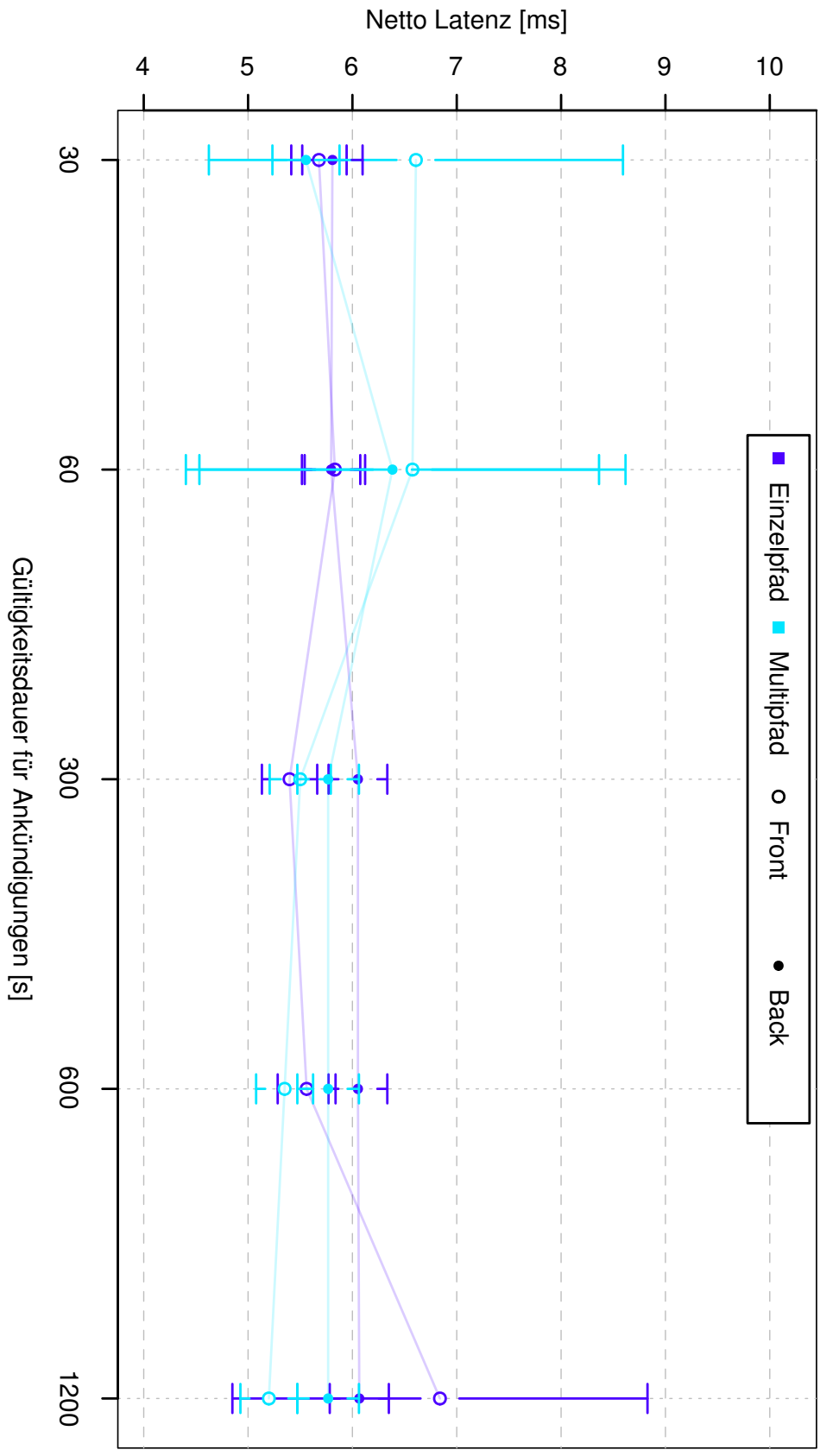


Abbildung B.56.:
Die Anknüpfungslatenz für die Übertragung von Fahrplaninformationen an mobile Senken in Abhängigkeit von der Versuchskonfiguration für Versuch 12 mit 95 % Konfidenzintervallen. Die halbtransparenten Verbindungslinien dienen der besseren Erkennbarkeit der Zusammengehörigkeit der einzelnen Messwerte und tragen selbst keine Informationen.

Kurzfassung

Die spontane Kooperation von Anwendungen auf Geräten in der Umgebung eines Nutzers, mit dem Ziel, diesen zu unterstützen, wie sie zur Realisierung spontaner intelligenter Umgebungen in einer heterogenen Anwendungs- und Kommunikationslandschaft notwendig ist, stellt sehr spezifische Anforderungen an die Entkopplungseigenschaften einer Kommunikations-Middleware. Da bestehende Taxonomien zur Einordnug dieser Entkopplungseigenschaften nicht hinreichend sind, wird in dieser Arbeit eine aus fünf Dimensionen der Entkopplung – räumlich, zeitlich, inhaltlich sowie in Bezug auf den Produzenten und den Konsumenten – bestehende Taxonomie entworfen. Anhand dieser wird die Problemstellung in den Stand der Forschung eingeordnet. Für die spezifischen Anforderungen in heterogenen, dynamischen Umgebungen existiert keine geeignete Kommunikations-Middleware, lässt sich aber auf Basis des Publish-Subscribe-Paradigmas mit einigen Erweiterungen realisieren.

Dafür wird die räumlich und inhaltlich entkoppelte Kommunikation zwischen Nachrichtenproduzenten und -konsumenten auf eine Pfadsuche in einer Vermittlungstopologie abgebildet. Diese entsteht als Zusammenführung von Netzwerk- und Verarbeitungstopologie und kann durch eine geeignete Systemarchitektur bestehend aus drei Schichten – Netzwerkabstraktion, Anwendungsabstraktion und Routing – wiederum auf die reale heterogene Geräteinfrastruktur abgebildet werden. Da das Routing in der Vermittlungstopologie ausschließlich auf Nachbarschaftsbeziehungen basiert, kann eine Unabhängigkeit von jeglicher Kommunikationsinfrastruktur sowie Topologieinformationen und somit maximale Flexibilität im Umgang mit Heterogenität und Dynamik der Umgebung gewährleistet werden. Zur Bewertung gefundener Pfade werden die Anforderungen an geeignete Metriken beschrieben und eine solche entworfen.

Für die Pfadsuche in der Vermittlungstopologie werden zwei geeignete Algorithmen entworfen, die jeweils in drei Phasen arbeiten. In der Ankündigungsphase wird die Verfügbarkeit von Nachrichten per Flooding in der Vermittlungstopologie propagiert. In der Abonnementphase werden ein oder mehrere Pfade markiert. Anschließend werden die Nachrichten übertragen. Der Ein-Wege-Ansatz nutzt ohne großen Verwaltungsaufwand den besten Pfad aus der Abonnementphase solange er verfügbar ist und fängt dann von vorne an. Der Mehr-Wege-Ansatz folgt einem Distanzvektorverfahren. Der beste noch verfügbare Pfade wird erst bei der Übermittlung der Nachrichten ausgewählt. Eine theoretische Analyse ausgewählter Eigenschaften, insbesondere Komplexität und Skalierbarkeit, wird durchgeführt.

Die beiden Algorithmen werden in einem Testszenario und einem praxisnahen Szenario simuliert. Sie sind einem Flooding-Algorithmus deutlich überlegen wo ein solcher nutzbar ist. Der Mehr-Wege-Ansatz zeigt deutliche Vorteile gegenüber dem Ein-Wege-Ansatz, insbesondere weil die Zuverlässigkeit nicht von der Gültigkeitsdauer der Ankündigungen abhängt. Diese müssen somit seltener aufwendig in der Vermittlungstopologie verbreitet werden. Verschiedene Schwächen der Algorithmen werden aufgedeckt, diskutiert und gegebenenfalls mögliche Erweiterungen zur Umgehung derselben aufgezeigt.

Abstract

The main concept of spontaneous smart environments is to have the devices surrounding a user work together to provide assistence to that user. To allow this spontaneous cooperation of applications in a heterogeneous application and communication landscape a communication middleware is necessary that provides specific decoupling characteristics. Existing taxonomies are not adequate to classify these characteristics. Therefore this thesis formulates a new taxonomy consisting of five decoupling dimensions - space, time, contents as well as relating to the producer and the consumer of information. With the help of this new taxonomy the problem is put into context of current research. Currently there exists no communication middleware to fulfill the given demands in a heterogeneous dynamic environment. However the analysis showed, that based on the publish-subscribe paradigm with small enhancements such a communication middleware can be developed.

To acomplish that task, a combination of networking and processing topology is defined that allows to map space and content related decoupling to a path search inside this so called brokering topology. This brokering topology is mapped onto the heterogeneous device infrastructure using a layered system architecture. The three layers of each device's implementation are network abstraction, application abstraction and routing. Since routing inside the brokering topology is strictly based on neighboring relations, independency of a communication infrastructure and any network topology information is guaranteed which provides maximum flexibility in heterogeneous and dynamic environments. To evaluate detected paths, requirements for a suitable metric are formulated and such a metric is defined.

Two algorithms for routing inside the brokering topology are designed, each of them working in three phases. In the announcement phase the availability of messages of a given type is propaged throughout the entire brokering topology using a flooding algorithm. In the subscription phase one or multiple paths are tagged and in the publication phase messages are transmitted. The single-path-approach establishes the best weighted path learned in the subscription phase with little administration overhead and uses it for message transmission as long as possible. If the path is lost, it starts over. The multi-path-approach tagges multiple possible paths using a distance vector approach and decides which one to use at message transmission time allowing for graceful degradation. A theoretical analyses of selected characteristics of both algorithms, especially complexity and scalability, is conducted.

Both algorithms are analyzed afterwards by simulation in a test scenario and a more practice like scenario. They are both superior to a simple flooding approach in settings that would allow for such an approach. The multi-path-algorithm shows significant advantages over the single-path-algorithm, especially because its reliability does not depend on the length of the validity period of announcements. Therefor less announcements need to be propagated which would be very expensive in terms of network utilization. Different weaknesses of both approaches are discovered, discussed and possible extensions to avoid these are presented.