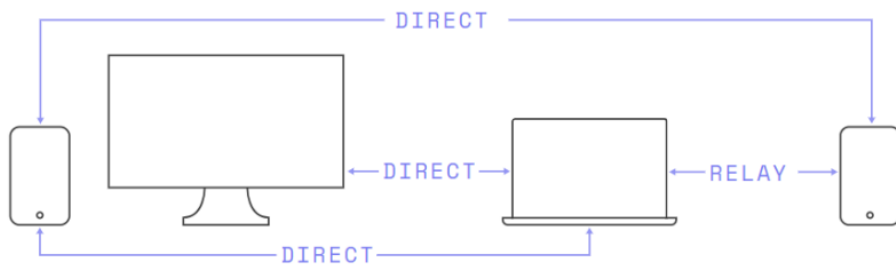


Experimentelle Untersuchung der Zuverlässigkeit des iroh-Gossip-Protokolls unter realistischen Netzwerkbedingungen

iroh



Bericht zum Seminar «New Trends for Local and Global
Interconnects for P2P Applications»

Universität Basel, Herbstsemester 2025/2026

vorgelegt von Andrea Seehuber, 28.12.2025

Inhaltsverzeichnis

| | | |
|-------|--|----|
| 1 | Einleitung..... | 1 |
| 1.1 | Motivation | 1 |
| 1.2 | Zielsetzung | 1 |
| 2 | Theoretischer Hintergrund | 1 |
| 2.1 | Epidemisches Broadcasting | 1 |
| 2.2 | Nachbarschaftsverwaltung und Nachrichtenverteilung | 2 |
| 2.2.1 | HyParView | 2 |
| 2.2.2 | Plumtree | 2 |
| 2.3 | Das iroh-gossip Protokoll im iroh-Stack..... | 3 |
| 3 | Experimentelles Setup..... | 4 |
| 3.1.1 | Variierte Parameter und Messziele | 4 |
| 3.1.2 | Ablauf eines Testlaufs | 5 |
| 4 | Ergebnisse | 5 |
| 4.1 | Netzwerkbildung und Overlay-Dynamik | 5 |
| 4.1.1 | Fehlgeschlagene Testläufe | 5 |
| 4.1.2 | Join-Verhalten | 6 |
| 4.1.3 | Active View und NeighborUp/Down-Ereignisse..... | 7 |
| 4.2 | Nachrichtenübertragung | 8 |
| 4.2.1 | Delivery Rate, Duplikate und Out-of-Order Nachrichten..... | 8 |
| 4.2.2 | Latenz und LDH | 9 |
| 4.3 | Auswirkungen von Churn und Recovery | 10 |
| 5 | Diskussion | 11 |
| 5.1 | Netzwerkbildung..... | 11 |
| 5.2 | Overlay-Dynamik | 12 |
| 5.3 | Nachrichtenübertragung | 12 |
| 6 | Fazit..... | 13 |
| 7 | Literaturverzeichnis | 14 |
| 8 | Abbildungsverzeichnis..... | 15 |

1 Einleitung

1.1 Motivation

Peer-to-Peer-Netzwerke (P2P) spielen in vielen modernen digitalen Anwendungen eine zentrale Rolle. Im Gegensatz zu klassischen Client-Server-Strukturen verteilen sie Aufgaben, Daten und Kommunikation auf viele gleichberechtigte Teilnehmer. Je nach Struktur lassen sich verschiedene Formen solcher Netzwerke unterscheiden:

- Reine, vollständig **DEZENTRALE P2P-NETZWERKE** kommen ohne zentrale Serverinstanz aus. Jeder Knoten übernimmt gleichzeitig die Rolle eines Clients und eines Servers. Ein bekanntes Beispiel ist das BitTorrent Filesharing-Protokoll, das unter anderem von Facebook genutzt wird (van der Sar, Ernesto, 2010).
- Daneben existieren **SERVERGESTÜTZTE P2P-NETZWERKE**, bei denen zentrale Server bestimmte Aufgaben übernehmen, etwa Koordination der Benutzerverwaltung, während Daten teilweise direkt zwischen Peers übertragen werden. Ein prominentes Beispiel hierfür ist der Online-dienst Discord, der WebRTC als Kommunikationsprotokoll-Stack nutzt (Vass, Jozsef, 2018).

Die iroh-Bibliothek stellt eine Art hybrider Technologie dar, die sowohl vollständig dezentrale als auch Relay-Server-gestützten Kommunikation ermöglicht. Sie bietet einen modularen Netzwerk-Stack, dessen Komponenten unter anderem einen dezentralen Broadcast-Dienst (iroh-gossip) einschließen. Dieser soll Nachrichten effizient in einem dynamischen Peer-Netzwerk verteilen, selbst wenn Knoten ausfallen, Netzwerkverbindungen instabil sind oder sich Knoten ständig neu verbinden (n0, 2025). Das iroh-gossip Protokoll wird seit Kurzem vom Instant-Messaging-Dienst Delta Chat implementiert, der für alle gängigen Betriebssysteme als datenschutzfreundliche Alternative zu WhatsApp zur Verfügung steht (merlinux GmbH, 2025).

1.2 Zielsetzung

Ziel dieses Projekts ist es, das Verhalten von iroh-gossip systematisch zu untersuchen. Dazu wurde ein eigenes Messframework entwickelt, das Linux Network Namespaces und netem nutzt, um realistische Netzwerkbedingungen zu simulieren – von stabilen LAN-Topologien über künstlich erhöhte Latenzen und Paketverluste bis hin zu dynamischen Churn-Situationen, in denen einzelne Knoten temporär das Netzwerk verlassen. Die Analyse soll zeigen, wie zuverlässig, schnell und robust iroh-gossip unter solchen Bedingungen arbeitet und welche Rückschlüsse sich daraus für den praktischen Einsatz ziehen lassen.

2 Theoretischer Hintergrund

2.1 Epidemisches Broadcasting

Epidemische oder gossip-basierte Broadcastverfahren sind eine Klasse verteilter Algorithmen, die sich an der Ausbreitungsdynamik biologischer Epidemien orientieren. Anstatt Nachrichten über eine feste Baumstruktur oder eine zentrale Koordinationsinstanz zu verteilen, erfolgt die Verbreitung durch wiederholtes Weiterleiten zwischen zufällig ausgewählten Nachbarn. Dadurch

breiten sich Informationen stochastisch und hochredundant im Netzwerk aus und erreichen im Idealfall alle aktiven Teilnehmer.

Der zentrale Vorteil epidemischer Verfahren ist ihre Robustheit: Da keine feste Struktur vorausgesetzt wird, bleibt die Kommunikation auch bei hohen Ausfallraten einzelner Knoten funktionsfähig. Dies macht Gossip-Ansätze attraktiv für dynamische P2P-Systeme, in denen Knoten häufig beitreten, ausfallen oder sich bewegen. Zudem ist die Methode hochskalierbar, da der Kommunikationsaufwand pro Knoten unabhängig von der Gesamtgrösse des Systems bleibt.

Demgegenüber steht ein wesentlicher Nachteil: Die stochastische Natur des Verfahrens führt zu hoher Redundanz. Dieselbe Nachricht erreicht viele Knoten mehrfach, was bei grossen Netzwerken zu erheblichem Bandbreitenverbrauch führt. Aus diesem Grund beschäftigen sich aktuelle Varianten des Gossip-Broadcastings mit Mechanismen zur Reduktion redundanter Übertragungen, ohne die Zuverlässigkeit zu gefährden (Leitão, et al., 2007).

2.2 Nachbarschaftsverwaltung und Nachrichtenverteilung

Moderne Gossip-basierte Broadcastprotokolle bestehen in der Regel aus einem **MITGLIEDSCHAFTS- BZW. PEER-SAMPLING-PROTOKOLL**, das die lokale Nachbarsicht eines Knotens verwaltet, und einem **BROADCAST-PROTOKOLL**, das Nachrichten basierend auf diesen Nachbarsichten verteilt. Nachfolgend werden die beiden von iroh-gossip genutzten Protokolle beschrieben.

2.2.1 HyParView

HYPARVIEW (Hybrid Partial View) adressiert eine Kernschwäche früherer Gossip-Netzwerke: Wenn Knoten ausfallen, werden klassische Partial-View-Protokolle wie Cyclon oder Scamp schnell inkonsistent, und das Overlay kann seine Konnektivität verlieren. HyParView begegnet diesem Problem durch eine Zweiteilung der Nachbarsicht in

1. eine **AKTIVE VIEW**, sehr klein (typisch 4 – 6 Einträge), das dauerhaft Verbindungen zu Nachbarn aufrechterhält, und
2. eine **PASSIVE VIEW**, grösser (typisch 20 – 40 Einträge), das lediglich eine Liste erreichbarer Backup-Knoten enthält.

Die aktive View dient für den eigentlichen Nachrichtentransport; die passive View stellt sicher, dass ausgefallene Nachbar jederzeit ersetzt werden können. Knoten werden über periodische Shuffle-Operationen in der passiven View verteilt, sodass das Overlay dauerhaft durchmischt bleibt. Leitão et. al konnten in ihrer Evaluation zeigen, dass HyParView auch bei 80 – 90 % ausgefallenen Knoten das Netzwerk in wenigen Runden wieder verbinden kann (Leitão, et al., 2007).

2.2.2 Plumtree

Während HyParView lediglich feststellt, *mit wem* ein Knoten verbunden ist, definiert der Plumtree-Algorithmus, *wie* Nachrichten über diese Nachbarn verteilt werden. Der zentrale Ansatz besteht darin, die Vorteile strukturierter und unstrukturierter Broadcastverfahren zu kombinieren:

- **EAGER-PUSH (BAUMKOMPONENTE):** Jeder Knoten baut beim Empfang der ersten Nachricht dynamisch eine Baumkante zu dem Nachbarn auf, von dem er sie zuerst erhalten hat, und leitet die Nachricht aktiv an ausgewählte Nachbarn weiter. Dadurch entsteht ein effizienter, redundanzarmer Verbreitungsbaum.

- **LAZY-PUSH (GOSSIPKOMPONENTE):** Alle weiteren Nachbarn erhalten nur kurze **IHAVE**-Notifikationen. Fehlt ihnen eine Nachricht, können sie diese aktiv anfordern (**GRAFT**). So werden fehlende Kanten repariert, wenn der Baum unvollständig oder durch Ausfälle beschädigt ist.

Dadurch ist Plumtree deutlich effizienter als reines Gossip, behält aber die Robustheit, die reine Baumverfahren nicht liefern können (Leitão, et al., 2007).

2.3 Das iroh-gossip Protokoll im iroh-Stack

Das **IROH-GOSSIP-PROTOKOLL** bettet die in Kapitel 2.1 und 2.2 beschriebenen Algorithmen in den iroh-Netzwerkstack ein. iroh selbst ist eine Bibliothek für P2P-Kommunikation über QUIC (Netzwerkprotokoll auf der Transportschicht), die direkte Verbindungen zwischen Endpunkten herstellt und bei Bedarf automatisch auf Relay-Server zurückfällt (n0, 2025). Der zentrale Einstiegspunkt von iroh ist der **Endpoint**. Ein **Endpoint** repräsentiert einen Peer im Netzwerk, besitzt eine kryptographische Identität und verwaltet eingehende und ausgehende QUIC-Verbindungen. Auf diesen Verbindungen können verschiedene Anwendungsprotokolle parallel laufen. Die Auswahl erfolgt über ALPN (Application-Layer Protocol Negotiation). Ein **Router** nimmt die eingehenden Verbindungen entgegen und leitet sie an den jeweils passenden Protokoll-Handler weiter (n0, 2025).

Damit iroh-gossip überhaupt ein Overlay aufbauen kann, müssen sich die beteiligten Endpunkte zunächst gegenseitig finden. Diese Aufgabe übernimmt nicht das Gossip-Protokoll selbst, sondern die **Discovery**-Schicht des Endpoints. **Discovery** verknüpft die **EndpointId** mit konkreten Wählinformationen, also entweder einer Relay-URL oder direkt wählbaren Adressen (IP/Port). Jeder iroh-Endpoint veröffentlicht beim Start seine eigenen Adressinformationen automatisch bei den konfigurierten **Discovery**-Diensten. Andere Endpunkte können dann allein anhand der **EndpointId** die aktuellen Verbindungsdaten nachschlagen und eine Verbindung aufbauen. iroh stellt dafür mehrere **Discovery**-Backends bereit, u. a. (The Rust Foundation, 2025):

- **DNS DISCOVERY** nutzt einen speziellen DNS-Dienst, bei dem **EndpointIds** in TXT-Einträge auf Adressinformationen abgebildet werden. Number0 (die iroh Community) betreibt hierfür eine öffentliche Instanz, die in der Standardkonfiguration verwendet wird. Von dort aus versucht iroh, falls möglich, eine direkte Verbindung per Hole Punching aufzubauen; gelingt dies nicht, bleibt der Datenverkehr dauerhaft über den Relay-Server geroutet.
- **LOCAL DISCOVERY** ermöglicht es, Endpunkte im selben lokalen Netzwerk ohne zentralen Dienst zu finden. Dafür verwendet iroh einen mDNS-ähnlichen Mechanismus, bei dem sich Endpunkte im LAN gegenseitig per Multicast ankündigen.

Standardmässig verwendet iroh DNS Discovery, kann aber zur Laufzeit so konfiguriert werden, dass andere oder mehrere Discovery-Mechanismen parallel verwendet werden.

Auf Anwendungsebene bietet iroh-gossip eine relativ einfache API:

- Gossip-Gruppen (Swarms) werden über eine **TopicId** definiert.
- Um einem Swarm beizutreten, ruft ein Peer `subscribe_and_join(topic_id, bootstrap_peers)` auf. Die `bootstrap_peers` sind iroh-EndpointIds, über die der Knoten zunächst Verbindungen aufbaut; dann übernimmt HyParView die weitere Verteilung der Nachrichten.

- Die Anwendung erhält einen Sender/Empfänger-Split: Über den Sender werden die Nachrichten in das Topic verteilt, der Receiver liefert `Event::Received`-Ereignisse, sowie Strukturereignisse wie `NeighborUp` oder `NeighborDown`.

Der gesetzte `Discovery`-Modus ist schlussendlich entscheidend dafür, wie schnell sich ein Gossip-Swarm aufbauen kann, ob alle Peers rechtzeitig Bootstrap-Verbindungen finden und ob Broadcast-Nachrichten über direkte Verbindungen oder über Relays transportiert werden.

3 Experimentelles Setup

Die nachfolgenden Messungen wurden lokal auf einem ASUS Vivobook S14 Flip unter Linux Mint 22.2 (Cinnamon Edition) durchgeführt. Sämtliche Experimente wurden vollständig automatisiert über eine Reihe von Shell-Skripten ausgeführt. Die Messungen wurden stets über 10 unabhängige Testläufe pro Parameterkombination gemittelt, um zufällige Schwankungen und Ausreisser systematisch zu reduzieren. Für die Versuchsreihen wurde eine eigene, vereinfachte Rust-Implementierung eines iroh-gossip-Peers entwickelt. Ziel war es, die von iroh-gossip bereitgestellten Mechanismen unter kontrollierten Bedingungen möglichst transparent und ohne applikations-spezifische Nebeneffekte untersuchen zu können.

3.1.1 Variierte Parameter und Messziele

Um unterschiedliche Belastungs- und Umgebungsszenarien untersuchen zu können, wurden die folgenden Parameter variiert:

1. **DISCOVERY-MODUS.** Im Direct-Modus verbinden sich Peers direkt im LAN, Discovery erfolgt ohne Relay. Im Relay-Modus erfolgt der Erstkontakt über einen Relay-Server; direkte Verbindungen werden nur aufgebaut, wenn Hole-Punching gelingt.

Messziele: Einfluss auf das Join-Verhalten, Nachrichtenauslieferungsrate und Latenzen.

2. **ANZAHL DER PEERS.** Alle Testläufe wurden mit 10, 20, 30, 40 und 50 Peers im simulierten Netzwerk durchgeführt. Pro 10 gestarteter Peer-Prozesse wurde jeweils eine Bootstrap-NodeID extrahiert und für den Join-Prozess übergeben.

Messziele: Untersuchung von Skalierungseffekten auf Pfadlängen, Nachbarschaftsgrößen, Join-Verhalten und Zustellraten bei wachsendem Overlay.

3. **NETZWERKSZENARIO.** Mit Hilfe des Linux Kernel Tools netem wurden 50 ms Verzögerung \pm 5 ms Jitter sowie 30 % Paketverlust injiziert und gegen das ungestörte LAN verglichen.

Messziele: Robustheit in Bezug auf Latenzen, Zustellraten und Out-of-Order-Events bei degradierter Netzqualität.

4. **CHURN AND RECOVERY.** 40 Sekunden nach Beginn des Testlaufs werden zufällig ausgewählte Peers aus dem Netzwerk isoliert (Churn). Die Churn-Rate beträgt 10 %, 20 % oder 30 %, wobei Bootstrap-Peers nie isoliert werden. Die Peers werden nach 10 Sekunden wieder ins Netz integriert.

Messziele: Untersucht wird die Fähigkeit des Protokolls, ein beschädigtes Overlay zu reparieren und die Broadcast-Reichweite wiederherzustellen.

3.1.2 Ablauf eines Testlaufs

Für jede Konfiguration wurde derselbe Ablauf verwendet:

1. **AUFBAU EINER ISOLIERTEN TESTUMGEBUNG VIA LINUX NAMESPACES.** Jeder Peer-Prozess läuft in einem eigenen Namespace mit separatem Netzwerkstack, eigener Routing-Tabelle und eigener virtuellen Netzwerkschnittstelle. Alle Peers sind über eine gemeinsame virtuelle Linux-Bridge verbunden, die das simulierte LAN darstellt.
2. **START DER RECEIVER-PROZESSE.** In allen Peers wird zunächst der Receiver gestartet, um sicherzustellen, dass das Overlay vollständig aufgebaut werden kann.
3. **STARTEN DES SENDERS.** Der Sender-Prozess startet erst, nachdem alle Receiver bereit waren.
4. **VERSAND DES TESTDATENSATZES.** 2000 Nachrichten werden mit einer Frequenz von 10 Nachrichten pro Sekunde und einer Payload-Größe von 256 Bytes verschickt. Anzahl und Sendefrequenz wurden so gewählt, dass die passive View im HyParView-Protokoll während eines vollständigen Testlaufs dreimal durchmischt wird. Die Nachrichten enthalten einen eindeutigen Testlauf-Indikator, eine Sequenznummer, einen Sender-Zeitstempel sowie die erwartete Gesamtzahl aller Nachrichten.
5. **ERFASSUNG ALLER EREIGNISSE IN STRUKTURIERTER JSONL-FORM.** Die Zusammenfassung, die hauptsächlich zur Evaluation herangezogen wird, enthält u. a. die folgenden Informationen:
 - Auslieferungsrate
 - Anzahl an Duplikaten
 - Out-of-Order Events
 - Latenzstatistiken
 - Last-Delivery-Hop Metriken
 - durchschnittliche Zeit ohne aktive Nachbarn
 - durchschnittliche Anzahl aktiver Nachbarn
 - Join-Zustand
6. **AUTOMATISIERTES TERMINIEREN DER PEERS.** Alle Prozesse werden nach Ablauf eines dynamisch berechneten Join-Timeouts beendet, um auch fehlerhafte oder unvollständig gestartete Läufe zuverlässig zu beenden. Der Timeout übersteigt jeweils sicher die Länge eines Testlaufs, bei dem die maximale Anzahl an Nachrichten verschickt wird.

4 Ergebnisse

Die Auswertung der im Rahmen der Experimente erzeugten Log-Dateien erfolgte vollständig automatisiert mithilfe eines selbst implementierten Python-Skripts unter der Verwendung der Python-Bibliothek `pandas`. Dieses Skript liest die pro Testlauf erzeugten JSON-basierten Zusammenfassungen ein, aggregiert die relevanten Metriken über mehrere Wiederholungen und berechnet statistische Kennwerte wie Mittelwerte und Verteilungen.

4.1 Netzerkennung und Overlay-Dynamik

4.1.1 Fehlgeschlagene Testläufe

Ein erster zentraler Befund ergab sich bereits bei der ersten Analyse der initialen Testläufe, die nach Discovery-Modus und Netzwerkqualität unterscheiden sollten. In diesen frühen Konfigurationen wurde – unabhängig von der Gesamtzahl der gestarteten Peer-Prozesse – jeweils nur eine

einzelne Bootstrap-NodeID an alle Peers übergeben. In mehreren Szenarien führte dies zu einer hohen Rate an nicht erfolgreichen Testläufen, da der sendende Peer dem Gossip-Topic nie beitreten konnte und folglich keine Nachrichten versendet wurden. Insbesondere im Relay-Modus konnte weiterhin beobachtet werden, dass die Fehlerraten abhängig vom Zeitpunkt der Messung einer sehr hohen Varianz unterliegen. Auf Basis dieser Beobachtungen wurde die Bootstrap-Strategie angepasst und pro jeweils 10 gestarteten Peer-Prozessen eine Bootstrap-NodeID übergeben. Diese Anpassung führte zu einer verbesserten Stabilität der Testläufe über alle betrachteten Szenarien hinweg. Die Ergebnisse sind in Tab. 1 zusammengefasst:

Tab. 1: Anteil an nicht erfolgreichen Testläufen bei einer übergebenen Bootstrap-NodeID pro 10 gestarteter Peer-Prozesse. Die Werte pro Zelle beziehen sich jeweils auf die Anzahl an Peer-Prozessen des jeweiligen Testlaufs (10, 20, 30, 40 und 50) und sind als grobe Mittelwerte zu verstehen.

| | ohne Netzwerkstörung | | | | | mit Netzwerkstörung | | | | |
|------------------|----------------------|------|------|------|------|---------------------|-----|-----|-----|------|
| Discovery Direct | 0 % | 10 % | 10 % | 0 % | 10 % | 0 % | 0 % | 0 % | 0 % | 0 % |
| Discovery Relay | 10 % | 10 % | 10 % | 30 % | 30 % | 0 % | 0 % | 0 % | 0 % | 25 % |

Die teilweise nicht intuitive Zuordnung von Use-Case-Bezeichnern im Auswertungsskript, in den Verzeichnissen der Log-Dateien sowie den Shell-Skripten ergibt sich aus dieser Anpassung. Eine detaillierte Beschreibung hierzu ist in einer separaten Textdatei im Repository dokumentiert. Da die Anzahl der verwendeten Bootstrap-NodeIDs selbst keine untersuchte Metrik darstellt, wurden die Use-Case-Bezeichnungen für die Darstellung und Diskussion der Ergebnisse vereinheitlicht.

4.1.2 Join-Verhalten

Das **JOIN-VERHALTEN** beschreibt den Anteil der gestarteten Receiver-Prozesse, die einem Gossip-Topic erfolgreich beigetreten sind und infolgedessen Broadcast-Nachrichten empfangen konnten. In die Auswertung gehen ausschliesslich gültige Testläufe ein, d. h. Läufe, in denen der sendende Peer dem Topic erfolgreich beigetreten ist. Im Discovery-Modus **Direct** zeigt sich über alle untersuchten Szenarien hinweg ein sehr stabiles Join-Verhalten: Der Anteil der erfolgreich beigetretenen Receiver liegt unabhängig von der Gesamtzahl der gestarteten Peers und vom Vorhandensein künstlicher Netzwerkstörungen nahezu konstant bei 100 %. Im Modus **Relay** erreichen ohne Netzwerkstörungen ebenfalls nahezu alle Receiver einen erfolgreichen Topic-Join. Unter aktivierter Netzwerkstörung sinkt der Anteil beigetretener Receiver jedoch auf etwa 90 % bei 10 Peers im Netzwerk und etwa 80 % bei 20 – 50 Peers im Netzwerk (siehe Abb. 1).

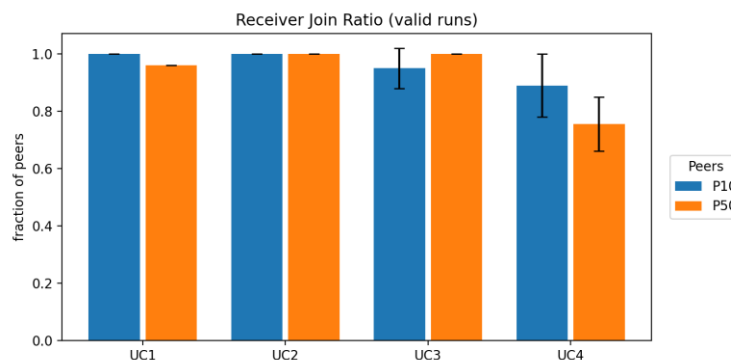


Abb. 1: Anteil an Peer-Prozessen, die dem Gossip-Topic beigetreten sind und Nachrichten empfangen haben. UC1/UC3: Discovery-Modus Direct, ohne bzw. mit injizierter Netzwerkstörung; UC2/UC4: Discovery-Modus Relay, ohne bzw. mit injizierter Netzwerkstörung.

4.1.3 Active View und NeighborUp/Down-Ereignisse

Die **ACTIVE VIEW** beschreibt die Menge an Peers, zu denen ein Knoten aktuell eine aktive Overlay-Verbindung unterhält. In der zugrundeliegenden HyParView-Implementierung von iroh ist die maximale Grösse der Active View auf fünf Peers begrenzt (n0, 2025). Die Ergebnisse zeigen, dass ohne Netzwerkstörung im Direct-Modus tendenziell eine etwas höhere durchschnittliche Anzahl an Peers in der Active View erreicht wird als im Relay-Modus. Dieser Effekt ist über alle betrachteten Szenarien konsistent. Im Direct-Modus fällt die durchschnittliche Grösse der Active View in grösseren Netzwerken geringfügig niedriger aus als in kleineren Netzwerken (siehe Abb. 2).

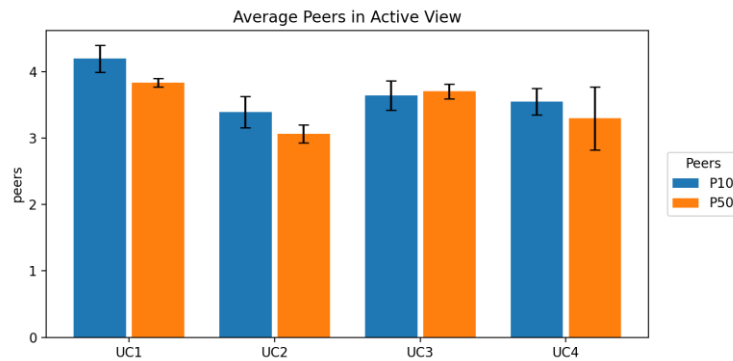


Abb. 2: Durchschnittliche Anzahl an benachbarten Knoten in der Active View eines Peers. UC1/UC3: Discovery-Modus Direct, ohne bzw. mit injizierter Netzwerkstörung; UC2/UC4: Discovery-Modus Relay, ohne bzw. mit injizierter Netzwerkstörung.

Zusätzlich wurden auch NeighborUp- und NeighborDown-Ereignisse ausgewertet. Diese Ereignisse zählen, wie häufig Peers in die Active View aufgenommen beziehungsweise daraus entfernt werden und dienen somit als Indikator für die Dynamik und Stabilität der Overlay-Nachbarschaft. In allen betrachteten Szenarien skaliert die Anzahl solcher Ereignisse im Wesentlichen mit der Netzwerkgrösse, ein relevanter Unterschied zwischen den beiden Discovery-Modi kann hingegen nicht festgestellt werden (siehe Abb. 3).

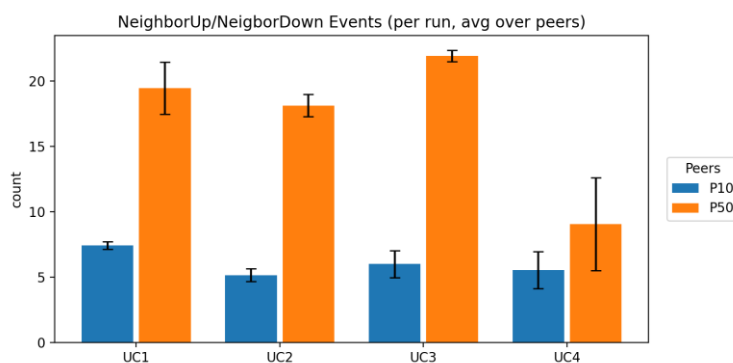


Abb. 3: Durchschnittliche Anzahl von Änderungen der Active View eines Peers, durch Hinzufügen oder Entfernen von anderen Knoten im Netzwerk. UC1/UC3: Discovery-Modus Direct, ohne bzw. mit injizierter Netzwerkstörung; UC2/UC4: Discovery-Modus Relay, ohne bzw. mit injizierter Netzwerkstörung.

Ohne injizierte Netzwerkstörung haben die meisten Peers während der gesamten Dauer eines Testlaufs mindestens einen benachbarten Knoten in ihrer Active View. Nur im Relay-Modus treten bei grösseren Netzwerken vereinzelt Fälle von Peers auf, die akkumuliert über den gesamten Testlauf während 4 – 5 s eine leere Active View haben. Bei beiden Discovery-Modi nimmt unter simulierter, schlechter Netzwerkqualität die durchschnittliche Zeit zu, während der die Peers im

Netzwerk eine leere Active View haben (siehe Abb. 4). Die Varianz ist hier allerdings sehr gross, da es im Relay-Modus in kleinen Netzwerken z. B. auch einzelne Peers gibt, deren Active View während des gesamten Testlaufs nie leer ist.

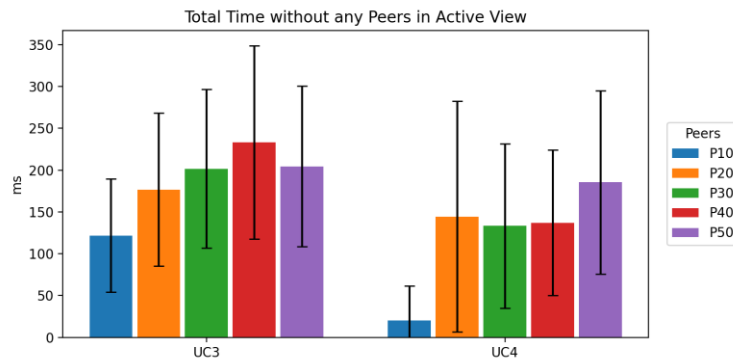


Abb. 4: Während eines Testlaufs aufsummierte Zeitdauer, die ein Peer eine leere Active View hatte. Die Werte sind über alle Peers im Netzwerk gemittelt. UC3: Discovery-Modus Direct; UC4: Discovery-Modus Relay. In beiden Szenarien wurde eine schlechte Netzqualität simuliert.

4.2 Nachrichtenübertragung

4.2.1 Delivery Rate, Duplikate und Out-of-Order Nachrichten

Unabhängig vom Discovery-Modus liegt bei gültigen Testläufen die Rate an zugestellten Nachrichten liegt – sofern die Receiver dem Topic beigetreten sind – praktisch durchgängig bei 100 %. Duplikate traten in keinem der ausgewerteten Testläufe auf.

OUT-OF-ORDER-EREIGNISSE bezeichnen empfangene Nachrichten, deren Sequenznummer kleiner ist als die einer zuvor empfangenen Nachricht und die somit nicht in der ursprünglichen Sendereihenfolge eintreffen. Solche Ereignisse wurden ausschliesslich unter aktivierter Netzwerkstörung beobachtet, die Ergebnisse sind in der Abb. 5 dargestellt. Im Direct-Modus treten bei 10 Peers im Netzwerk im Mittel rund 170 Out-of-Order Nachrichten auf; bei 50 Peers steigt dieser Wert auf etwa das Doppelte. Im Relay-Modus ist die Anzahl der Out-of-Order-Nachrichten bei mehr Peers im Netzwerk im Vergleich geringer.

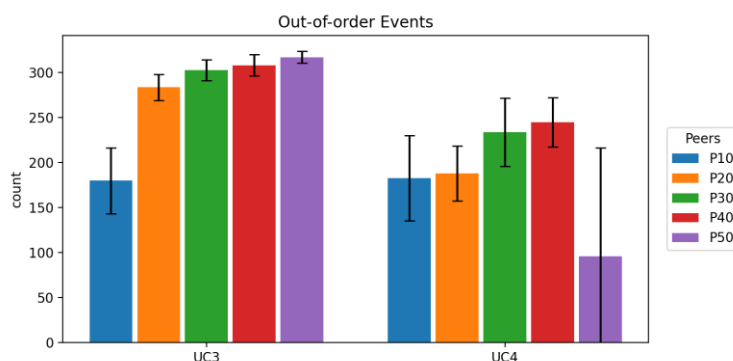


Abb. 5: Anzahl an Nachrichten, die nicht in der vorgesehenen Reihenfolge empfangen wurden UC3: Discovery-Modus Direct; UC4: Discovery-Modus Relay. In beiden Szenarien wurde eine schlechte Netzqualität simuliert.

4.2.2 Latenz und LDH

Die **LATENZ** beschreibt die **ENDE-ZU-ENDE-VERZÖGERUNG** einer Nachricht zwischen dem Zeitpunkt ihres Versands und ihrem Empfang am Zielknoten. Zur Charakterisierung der Latenzverteilung werden Quantile betrachtet: Das 50-Perzentil (p50) gibt die mediane Latenz an, während höhere Quantile (p90, p99) sowie das Maximum Auskunft über das Verhalten im oberen Rand der Verteilung und über Ausreisser geben. Auf diese Weise lassen sich sowohl typische als auch seltene, besonders verzögerte Zustellungen erfassen.

Für die Use Cases ohne Netzwerkstörung (UC1 und UC2 in der Abb. 6) liegen die Latenzen insgesamt auf einem niedrigen und stabilen Niveau. Vor allem die maximalen Übertragungszeiten skalieren hierbei aber deutlich mit der Grösse des Netzwerks. Unter Netzwerkstörung (UC3 und UC4 in der Abb. 6) steigen die Latenzen an und erreichen insbesondere bei den höheren Quantilen und Maximalwerten das bis zu 10fache der Zeiten, die ohne Netzwerkstörung gemessen wurden.

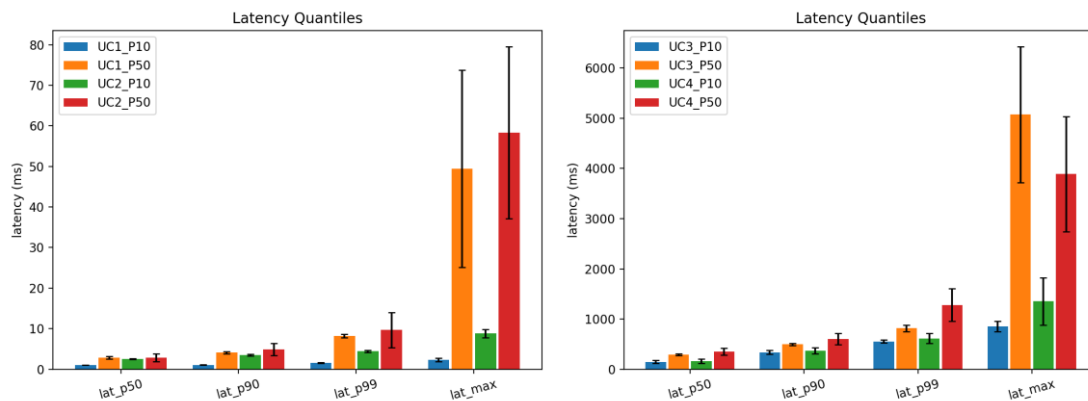


Abb. 6: Ende-zu-Ende-Latenzen ohne (links) bzw. mit (rechts) simulierter Netzwerkbeeinträchtigung. UC1/UC3: Discovery-Modus Direct; UC2/UC4: Discovery-Modus Relay. Dargestellt sind die Werte für die Szenarien mit der jeweils kleinsten und grössten untersuchten Anzahl an Peers im Netzwerk (10 bzw. 50).

Die **LAST DELIVERY HOP (LDH)-METRIK** beschreibt die Anzahl der Overlay-Weiterleitungsschritte, die eine Nachricht bis zu ihrem endgültigen Empfänger durchläuft. Hier wurden ebenfalls Quantile ausgewertet, um neben dem typischen Verhalten auch die maximale Pfadlänge sichtbar zu machen (siehe Abb. 7).

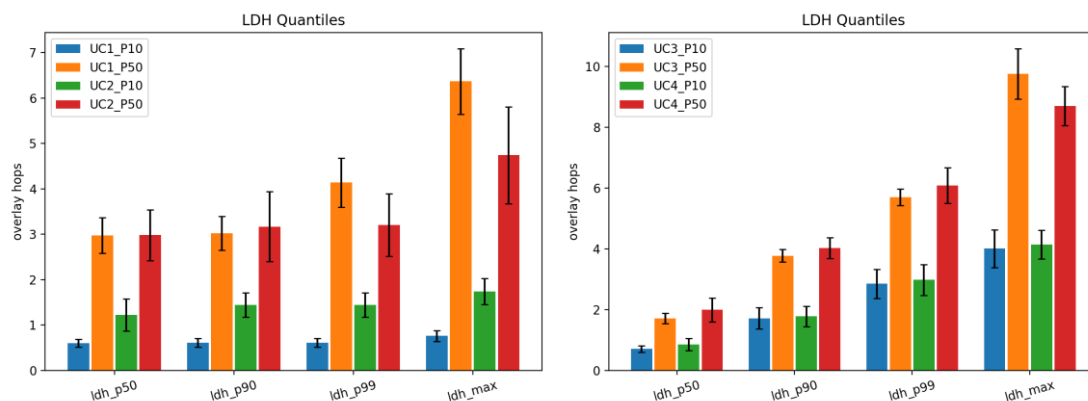


Abb. 7: Durchschnittliche Anzahl an Last-Delivery-Hops ohne (links) bzw. mit (rechts) simulierter Netzwerkbeeinträchtigung. UC1/UC3: Discovery-Modus Direct; UC2/UC4: Discovery-Modus Relay. Dargestellt sind die Werte für die Szenarien mit der jeweils kleinsten und grössten untersuchten Anzahl an Peers im Netzwerk (10 bzw. 50).

Die LDH-Werte sind in störungsfreien Szenarien (UC1 und UC2 in der Abb. 7) und bei kleinen Netzwerken für den Discovery-Modus Relay höher als für den Direct-Modus, gleichen sich für grössere Netzwerke aber an bzw. werden im Relay-Modus tendenziell sogar weniger Hops gezählt. Mit simulierter Netzwerkstörung (UC3 und UC4 in der Abb. 7) werden für beide Discovery-Modi sehr ähnliche Werte gemessen.

4.3 Auswirkungen von Churn und Recovery

Alle bisher beschriebenen Ergebnisse beziehen sich auf Szenarien, bei denen während eines Testlaufs die Anzahl an Peers im Netzwerk stabil bleibt. Wird eine Teilmenge der Peers temporär aus dem Netzwerk isoliert, hat dies im Vergleich zu den bisher vorgestellten Ergebnissen keinen signifikanten Einfluss auf die folgenden Metriken:

- Die **ZUSTELLRATE** der versendeten Nachrichten bleibt stabil bei nahezu 100 %.
- Es werden weiterhin keine **NACHRICHTENDUPLIKATE** gezählt.
- Das unter 4.1.2 beschriebene **JOIN-VERHALTEN** lässt sich reproduzieren (vgl. Abb. 1).
- Die durchschnittliche **ANZAHL AN KNOTEN IN DER ACTIVE VIEW** eines Peers bleibt vergleichbar zu den unter 4.1.3 geschilderten Ergebnissen (vgl. Abb. 2). Auch die Anzahl gezählter Ereignisse, die zu einer Änderung der Active View führen, ändert sich nicht wesentlich (vgl. Abb. 3).
- Die Pfadlängen der Nachrichtenübertragung (Anzahl der **LAST-DELIVERY-HOPS**) bleiben im Wesentlichen unverändert zu den unter 4.2.2 beschriebenen Ergebnissen (vgl. Abb. 7).

Quasi unabhängig vom Discovery-Modus, aber korrelierend mit dem prozentualen Anteil an Knoten, die während 10 s isoliert werden, werden nun auch im ungestörten Netzwerk vereinzelte **OUT-OF-ORDER-EREIGNISSE** gezählt (siehe Abb. 8). Davon betroffen sind allerdings nur die Peers, die aus dem Netzwerk isoliert wurden, was die hohe Varianz der Durchschnittswerte erklärt.

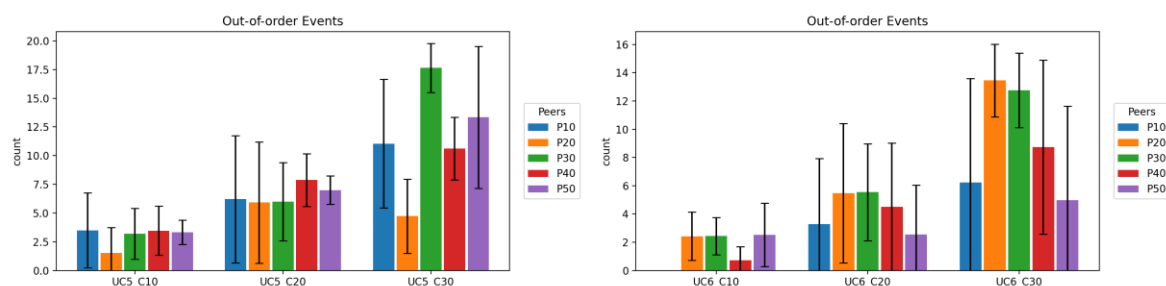


Abb. 8: Anzahl an Nachrichten, die nicht in der vorgesehenen Reihenfolge empfangen wurden. UC5: Discovery-Modus Direct; UC6: Discovery-Modus Relay. Beide Szenarien sind ohne Netzwerkstörung. In Netzwerken mit jeweils 10, 20, 30, 40 und 50 Peers wurden 10 %, 20 % und 30 % der Peers für 10 s isoliert.

Unter schlechter Netzwerkqualität nimmt die Anzahl an Out-of-Order-Ereignissen deutlich zu, bleibt aber in einer ähnlichen Grössenordnung wie in den unter 4.4 geschilderten Szenarien (siehe Abb. 9). Betroffen sind nun auch wieder alle Peers, wobei kein wesentlicher Unterschied zwischen temporär isolierten und nicht-isolierten Peers festzustellen ist. Auffällig ist allerdings, dass im Discovery-Modus Direct die Anzahl an Nachrichten, die nicht in der korrekten Reihenfolge zugestellt werden, bei einer Netzwerkgrösse von 20 Peers ein Maximum erreicht und bei einer Netzwerkgrösse von 50 Peers am kleinsten ist (Abb. 9 links). Im Discovery-Modus Relay korrelieren die Out-of-Order-Ereignisse näherungsweise mit der Netzwerkgrösse, die Unterschiede sind allerdings nicht signifikant.

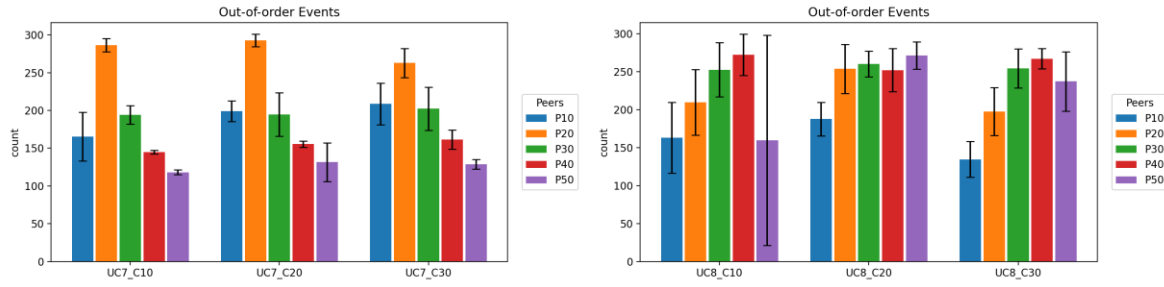


Abb. 9: Anzahl an Nachrichten, die nicht in der vorgesehenen Reihenfolge empfangen wurden. UC7: Discovery-Modus Direct; UC8: Discovery-Modus Relay. Die beide Szenarien sind mit simulierter Netzwerkstörung. In allen Netzwerken wurden 10 %, 20 % und 30 % der Peers für 10 s isoliert.

Die Auswertung der Latenzquantile zeigt in den Churn-Szenarien einen Anstieg der maximalen Ende-zu-Ende-Latenzen. Dieser Effekt ist sowohl in Konfigurationen ohne zusätzliche Netzwerkstörung (Abb. 10 links) als auch in Szenarien mit künstlich eingeführter Verzögerung (Abb. 10 rechts) zu beobachten. Die niedrigeren Quantile bleiben hingegen in der gleichen Größenordnung wie in den Szenarien ohne Churn (vgl. Abb. 6).

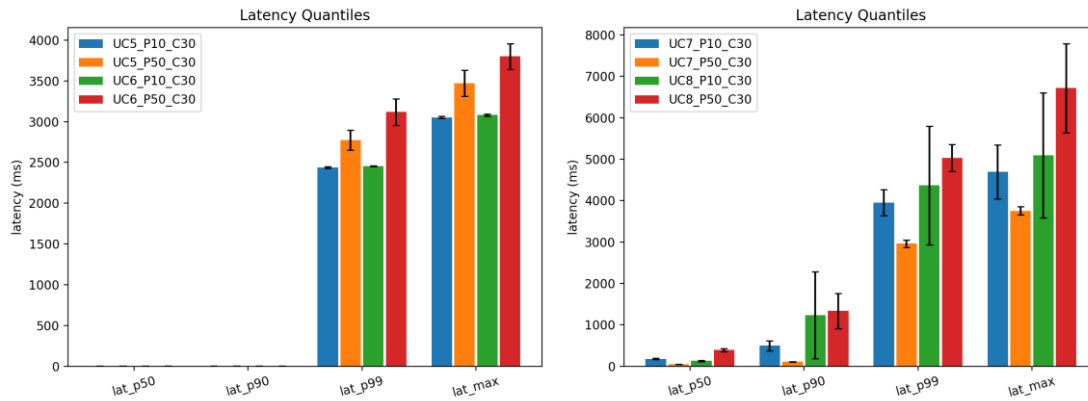


Abb. 10: Ende-zu-Ende-Latenzen ohne (links) bzw. mit (rechts) simulierter Netzwerkbeeinträchtigung. UC5/UC7: Discovery-Modus Direct; UC6/UC8: Discovery-Modus Relay. Dargestellt sind die Werte für die Szenarien mit der jeweils kleinsten und grössten untersuchten Anzahl an Peers im Netzwerk (10 bzw. 50). Es wurden jeweils 30 % der Peers im Netzwerk für 10 s isoliert.

Die Analyse der Logdateien der während des Churns isolierten Peers zeigt, dass die dort erfassten maximalen Latenzwerte zeitlich mit der Dauer der Netzisolation übereinstimmen.

5 Diskussion

5.1 Netzbildung

Das beobachtete **JOIN-VERHALTEN** verdeutlicht die zentrale Rolle der Discovery-Schicht für die Stabilität des Gossip-Overlays. Während im Direct-Modus über alle Messreihen hinweg ein reproduzierbares Join-Verhalten beobachtet wurde, zeigte sich im Relay-Modus eine ausgeprägte Variabilität hinsichtlich der Anzahl gültiger Testläufe. Diese Variabilität trat nicht nur innerhalb einzelner Versuchsreihen auf, sondern insbesondere auch zwischen Messungen, die an unterschiedlichen Tagen und an verschiedenen Standorten durchgeführt wurden. In einzelnen Fällen konnten im Relay-Modus über mehrere aufeinanderfolgende Versuche hinweg keine gültigen Testläufe erzielt werden. Diese Beobachtungen legen nahe, dass der Join-Prozess im Relay-Modus

empfindlich gegenüber äusseren Faktoren ist, die ausserhalb des eigentlichen Gossip-Protokolls liegen. Eine plausible Erklärung ist die Abhängigkeit von der Erreichbarkeit des Relay-Servers sowie von der Qualität der zugrundeliegenden Netzwerkverbindung des ausführenden Systems (dieses war immer über einen mobilen Hotspot mit dem 5G-Netzwerk der Swisscom verbunden). Da die Erreichbarkeit des n0-Relay-Dienstes nicht Teil des kontrollierten Versuchsaufbaus war, lassen sich auf Basis der vorliegenden Daten aber keine eindeutigen Aussagen treffen.

Weiterhin zeigte sich, dass sich im Relay-Modus die Anzahl gültiger Testläufe unter simulierter Netzwerkstörung deutlich stabilisiert, gleichzeitig der Anteil der Receiver-Prozesse, die dem Gossip-Topic erfolgreich beitreten, reproduzierbar auf etwa 80 % sinkt. Dieses Verhalten lässt sich dadurch erklären, dass sich der Join-Prozess unter schlechter Netzwerkqualität zeitlich entzerrt und damit deterministischer verhält. Die Beobachtung legt nahe, dass der Relay-Modus unter ungünstigen Netzwerkbedingungen eher zu einem stabilen, aber unvollständigen Overlay tendiert.

Ein weiterer kritischer Punkt betrifft die Notwendigkeit mehrerer **BOOTSTRAP-NODEIDs** für den stabilen Aufbau eines Gossip-Swarms. Die Experimente haben gezeigt, dass die Übergabe lediglich einer einzelnen Bootstrap-NodeID zu einer signifikant erhöhten Rate fehlgeschlagener Testläufe führt, insbesondere im Relay-Modus. Dieses Ergebnis wirft Fragen hinsichtlich der konzeptionellen Rolle der Bootstrap-Peers auf, insbesondere im Discovery-Modus Direct. In einem lokalen Netzwerk, in dem Peers sich über einen mDNS-ähnlichen Multicast-Mechanismus entdecken können, wäre zu erwarten, dass der initiale Join eines Gossip-Swarms ohne explizite Bootstrap-Peers möglich ist. Die iroh-API erzwingt jedoch mit der Methode `subscribe_and_join()` die Übergabe eines Vektors mit mindestens einer NodeID. Alternative Aufrufsequenzen, bei denen `subscribe()` und `join()` getrennt verwendet werden, sind zwar vorhanden, liessen sich im Rahmen dieser Arbeit jedoch nicht erfolgreich für einen vollständigen Swarm-Aufbau nutzen. Weder die verfügbare Dokumentation noch die API-Beschreibung auf der iroh-Webseite geben eindeutig vor, wie ein Join ohne explizite Bootstrap-Peers vorgesehen ist.

5.2 Overlay-Dynamik

Die Analyse der **ACTIVE VIEW** sowie der **NEIGHBORUP-/DOWN-EREIGNISSE** zeigt, dass sich die grundlegende Overlay-Struktur in beiden Discovery-Modi ähnlich verhält. Unter schlechter Netzwerkqualität nimmt die Zeitspanne ohne aktive Nachbarn erwartungsgemäss zu, ohne jedoch zu einem systematischen Zusammenbruch des Overlays zu führen. Dies spricht für eine funktionierende Selbstheilung der Nachbarschaftsstruktur durch HyParView, selbst wenn Verbindungen zeitweise instabil sind. Auch temporär isolierte Peers behalten eine stabile Active View und werden zumindest nach der für diese Arbeit gewählten, relativ kurzen Churn-Zeit zuverlässig wieder ins Gossip-Netzwerk integriert.

5.3 Nachrichtenübertragung

Die nahezu durchgängig erreichte **ZUSTELLRATE** von 100 % in allen gültigen Testläufen bestätigt, dass der kombinierte Einsatz von HyParView und Plumtree auch bei degradierten Netzwerken und temporären Knotenausfällen eine zuverlässige Nachrichtenverteilung ermöglicht. Die vollständige Abwesenheit von **DUPLIKATEN** in allen Szenarien bestätigt die Wirksamkeit des Plumtree-Ansatzes zur Reduktion redundanter Übertragungen. Gleichzeitig zeigen die **OUT-OF-ORDER-EREIGNISSE** unter Netzwerkstörung, dass iroh-gossip auf Transportebene keine strikte Ordnung

erzwingt. Dies ist konsistent mit der Verwendung von QUIC-Verbindungen und der stochastischen Weiterleitung im Overlay.

Die Auswertung der **LAST-DELIVERY-HOP-METRIKEN** zeigt, dass sich die effektiven Pfadlängen der Nachrichtenübertragungen im Discovery-Modus Direct und im Relay-Modus nur geringfügig unterscheiden. Der Relay-Modus wirkt sich somit primär auf die Transportebene einzelner Verbindungen aus, nicht jedoch auf die logische Anzahl der Weiterleitungsschritte im Overlay.

Die Analyse der **ENDE-ZU-ENDE-LATENZEN** zeigt, dass iroh-gossip im stabilen Betrieb vergleichsweise geringe und gut kontrollierte Verzögerungen aufweist. Die unteren Quantile unterscheiden sich bei Netzwerken mit 40 oder 50 Peers kaum zwischen den beiden Discovery-Modi, was darauf hindeutet, dass die Nachrichtenweiterleitung vorrangig über das Peer-to-Peer-Overlay stattfindet. Die beobachtete Zunahme der Maximalwerte mit wachsender Anzahl an Peers lässt vermuten, dass seltene, stark verzögerte Zustellungen durch längere Overlay-Pfade oder temporäre Verzögerungen einzelner Verbindungen verursacht werden, ohne das typische Latenzverhalten wesentlich zu beeinflussen. Weiterhin sprechen die Beobachtungen dafür, dass simulierte Netzwerkstörungen nicht zu einer gleichmässigen Verlangsamung der Nachrichtenverteilung führen, sondern vor allem einzelne Übertragungen stark verzögern. In Churn-Szenarien zeigt sich, dass die maximalen Latenzwerte mit der zeitlichen Dauer der Netzisolation einzelner Peers korrespondieren. Die erhöhten Maximalwerte sind somit primär auf die verzögerte Zustellung von Nachrichten nach der Wiederanbindung isolierter Knoten zurückzuführen.

6 Fazit

Im Rahmen dieser Arbeit wurde das iroh-gossip-Protokoll unter kontrollierten, aber realitätsnahen Netzwerkbedingungen systematisch untersucht. Die durchgeführten Experimente unterliegen dabei mehreren Limitationen: Trotz des Einsatzes von Linux Network Namespaces und netem lassen sich reale, globale Netzwerke mit hoher Heterogenität nur näherungsweise abbilden. Zudem wurden die Messungen auf eine maximale Netzwerkgrösse von 50 Peers begrenzt und unter einer festen Nachrichtenrate mit ebenfalls festgelegter Nachrichtengrösse durchgeführt. Effekte sehr grosser Netzwerke, variabler Lastprofile oder langfristiger Churn-Dynamiken konnten daher nicht untersucht werden. Auch die Erreichbarkeit externer Relay-Dienste war nicht Teil des Versuchsaufbaus und stellt eine potenzielle Störgrösse dar. Aus methodischer Sicht ist kritisch anzumerken, dass die Evaluation bewusst auf eine vereinfachte Testanwendung beschränkt wurde, um applikationsspezifische Einflüsse zu minimieren. Dies erhöht zwar die Vergleichbarkeit der Messergebnisse, schränkt jedoch die Übertragbarkeit auf komplexe reale Anwendungen ein.

Zusammenfassend zeigt diese Arbeit, dass iroh-gossip ein leistungsfähiges Gossip-basiertes Broadcast-Protokoll darstellt, dessen Stärken klar in der robusten Nachrichtenverteilung liegen. Gleichzeitig verdeutlichen die Ergebnisse, dass Discovery- und Bootstrap-Strategien einen entscheidenden Einfluss auf die praktische Nutzbarkeit haben und bei der Entwicklung von Anwendungen mit iroh-gossip besondere Aufmerksamkeit erfordern. Der Relay-Modus, der von der iroh-Community offiziell als Discovery-Standard empfohlen wird, sollte eher in Kombination mit dem Direct-Modus und als Fallback-Mechanismus für diesen implementiert werden. In jedem Fall ist es ratsam, auf mehrere Bootstrap-Peers bereitzustellen und den Join-Prozess explizit zu überwachen.

7 Literaturverzeichnis

Leitão, João, Pereira, José und Rodrigues, Luís. 2007. Epidemic Broadcast Trees. *26th IEEE International Symposium on Reliable Distributed Systems*. 2007, S. 303-310.

—. **2007.** HyParView: a membership protocol for reliable gossip-based broadcast. *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. 2007, S. 419-429.

merlinux GmbH. 2025. Delta Chat ist ein dezentralisierter und sicherer Messenger. *Delta Chat*. [Online] 2025. [Zitat vom: 9. Dezember 2025.] <https://delta.chat/de/>.

n0. 2025. iroh-gossip. [Online] GitHub Repository, 6. November 2025. [Zitat vom: 9. Dezember 2025.]

—. **2025.** p2p that just works. *iroh Computer*. [Online] 23. Oktober 2025. [Zitat vom: 9. Dezember 2025.] <https://www.iroh.computer/>.

The Rust Foundation. 2025. iroh - module discovery. [Online] Docs.rs, 6. November 2025. [Zitat vom: 9. Dezember 2025.] <https://docs.rs/iroh/latest/iroh/discovery/index.html>.

van der Sar, Ernesto. 2010. Facebook Uses BitTorrent, and They Love It. *Torrentfreak*. [Online] Juni 2010. [Zitat vom: 9. Dezember 2025.] <https://torrentfreak.com/facebook-uses-bittorrent-and-they-love-it-100625/>.

Vass, Jozsef. 2018. How Discord Handles Two and a Half Million Concurrent Voice Users using WebRTC. *Discord*. [Online] 10. September 2018. [Zitat vom: 9. Dezember 2025.]

8 Abbildungsverzeichnis

| | |
|--|----|
| ABB. 1: ANTEIL AN PEER-PROZESSEN, DIE DEM GOSSIP-TOPIC BEIGETRETEN SIND UND NACHRICHTEN EMPFANGEN HABEN. UC1/UC3: DISCOVERY-MODUS DIRECT, OHNE BZW. MIT INJIZIERTER NETZWERKSTÖRUNG; UC2/UC4: DISCOVERY-MODUS RELAY, OHNE BZW. MIT INJIZIERTER NETZWERKSTÖRUNG. | 6 |
| ABB. 2: DURCHSCHNITTliche ANZAHL AN BENACHBARTEN KNOTEN IN DER ACTIVE VIEW EINES PEERS. UC1/UC3: DISCOVERY-MODUS DIRECT, OHNE BZW. MIT INJIZIERTER NETZWERKSTÖRUNG; UC2/UC4: DISCOVERY-MODUS RELAY, OHNE BZW. MIT INJIZIERTER NETZWERKSTÖRUNG. | 7 |
| ABB. 3: DURCHSCHNITTliche ANZAHL VON ÄNDERUNGEN DER ACTIVE VIEW EINES PEERS. DURCH HINZUFÜGEN ODER ENTFERNEN VON ANDEREN KNOTEN IM NETZWERK. UC1/UC3: DISCOVERY-MODUS DIRECT, OHNE BZW. MIT INJIZIERTER NETZWERKSTÖRUNG; UC2/UC4: DISCOVERY-MODUS RELAY, OHNE BZW. MIT INJIZIERTER NETZWERKSTÖRUNG. | 7 |
| ABB. 4: WÄHREND EINES TESTLAUFS AUFSUMMIERTE ZEITDAUER, DIE EIN PEER EINE LEERE ACTIVE VIEW HATTE. DIE WERTE SIND ÜBER ALLE PEERS IM NETZWERK GEMITTELT. UC3: DISCOVERY-MODUS DIRECT; UC4: DISCOVERY-MODUS RELAY. IN BEIDEN SZENARIEN WURDE EINE SCHLECHTE NETZQUALITÄT SIMULIERT. | 8 |
| ABB. 5: ANZAHL AN NACHRICHTEN, DIE NICHT IN DER VORGESEHENEN REIHENFOLGE EMPFANGEN WURDEN UC3: DISCOVERY-MODUS DIRECT; UC4: DISCOVERY-MODUS RELAY. IN BEIDEN SZENARIEN WURDE EINE SCHLECHTE NETZQUALITÄT SIMULIERT. | 8 |
| ABB. 6: ENDE-ZU-ENDE-LATENZEN OHNE (LINKS) BZW. MIT (RECHTS) SIMULIERTER NETZWERKBEEINTRÄCHTIGUNG. UC1/UC3: DISCOVERY-MODUS DIRECT; UC2/UC4: DISCOVERY-MODUS RELAY. DARGESTELLT SIND DIE WERTE FÜR DIE SZENARIEN MIT DER JEWELNS KLEINSTEN UND GRÖSSTEN UNTERSUCHTEN ANZAHL AN PEERS IM NETZWERK (10 BZW. 50). | 9 |
| ABB. 7: DURCHSCHNITTliche ANZAHL AN LAST-DELIVERY-HOPS OHNE (LINKS) BZW. MIT (RECHTS) SIMULIERTER NETZWERKBEEINTRÄCHTIGUNG. UC1/UC3: DISCOVERY-MODUS DIRECT; UC2/UC4 DISCOVERY-MODUS RELAY. DARGESTELLT SIND DIE WERTE FÜR DIE SZENARIEN MIT DER JEWELNS KLEINSTEN UND GRÖSSTEN UNTERSUCHTEN ANZAHL AN PEERS IM NETZWERK (10 BZW. 50). | 9 |
| ABB. 8: ANZAHL AN NACHRICHTEN, DIE NICHT IN DER VORGESEHENEN REIHENFOLGE EMPFANGEN WURDEN. UC5: DISCOVERY-MODUS DIRECT; UC6: DISCOVERY-MODUS RELAY. BEIDE SZENARIEN SIND OHNE NETZWERKSTÖRUNG. IN NETZWERKEN MIT JEWELNS 10, 20, 30, 40 UND 50 PEERS WURDEN 10 %, 20 % UND 30 % DER PEERS FÜR 10 S ISOLIERT. | 10 |
| ABB. 9: ANZAHL AN NACHRICHTEN, DIE NICHT IN DER VORGESEHENEN REIHENFOLGE EMPFANGEN WURDEN. UC7: DISCOVERY-MODUS DIRECT; UC8: DISCOVERY-MODUS RELAY. DIE BEIDE SZENARIEN SIND MIT SIMULIERTER NETZWERKSTÖRUNG. IN ALLEN NETZWERKEN WURDEN 10 %, 20 % UND 30 % DER PEERS FÜR 10 S ISOLIERT. | 11 |
| ABB. 10: ENDE-ZU-ENDE-LATENZEN OHNE (LINKS) BZW. MIT (RECHTS) SIMULIERTER NETZWERKBEEINTRÄCHTIGUNG. UC5/UC7: DISCOVERY-MODUS DIRECT; UC6/UC8: DISCOVERY-MODUS RELAY. DARGESTELLT SIND DIE WERTE FÜR DIE SZENARIEN MIT DER JEWELNS KLEINSTEN UND GRÖSSTEN UNTERSUCHTEN ANZAHL AN PEERS IM NETZWERK (10 BZW. 50). ES WURDEN JEWELNS 30 % DER PEERS IM NETZWERK FÜR 10 S ISOLIERT. | 11 |

Redlichkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig und ohne unerlaubte Hilfe angefertigt habe. Es wurden keine weiteren Quellen als jene, die im Literaturverzeichnis ausgewiesen sind, verwendet.

Ich habe im Rahmen der Erstellung dieser Arbeit die Sprach-KI ChatGPT (OpenAI) unterstützend eingesetzt. Die Nutzung erfolgte zur sprachlichen Überarbeitung von Textabschnitten des Berichtes, zur Klärung und Diskussion technischer Konzepte sowie zur Reflexion der experimentellen Ergebnisse. Darüber hinaus wurde ChatGPT bei der Erstellung und Überarbeitung von Teilen der für die Experimente verwendeten Shell-Skripte herangezogen. Der für diese Arbeit implementierte Rust-Code wurde primär auf Basis der offiziellen Rust- und iroh-Dokumentation entwickelt; KI-Unterstützung diente hierbei vor allem der Überprüfung von Entwurfsentscheidungen und der Klärung von Detailfragen. Alle durch ChatGPT generierten Vorschläge wurden von mir kritisch geprüft, bei Bedarf angepasst und in eigener Verantwortung in den Gesamtkontext der Arbeit integriert. Die Konzeption des Messaufbaus, die Durchführung der Experimente, die Auswertung der Ergebnisse sowie deren Interpretation und Diskussion lagen vollständig in meiner Verantwortung.

Für die Erstellung einer englischsprachigen Version dieses Berichts wurde der Übersetzungsdienst DeepL verwendet.

Der gesamte Code sowie die gesamte Dokumentation stehen auf GitHub zur Verfügung:

<https://github.com/Antimon80/iroh-gossip-metrics>

Basel, 28.12.2025

A handwritten signature in blue ink, appearing to read 'Luca Sehl'.