Grid und Torusbasiertes Fast Failover Routing

Abschlussvortrag Bachelorarbeit

Von Sebastian Peters

Motivation

- Ausfall einer 10 Gigabit Verbindung für 1s
- => Verlust 833k Paketen mit einer Größe von jeweils 1500 Bytes

[by Peter H. from Pixaby]

Roadmap

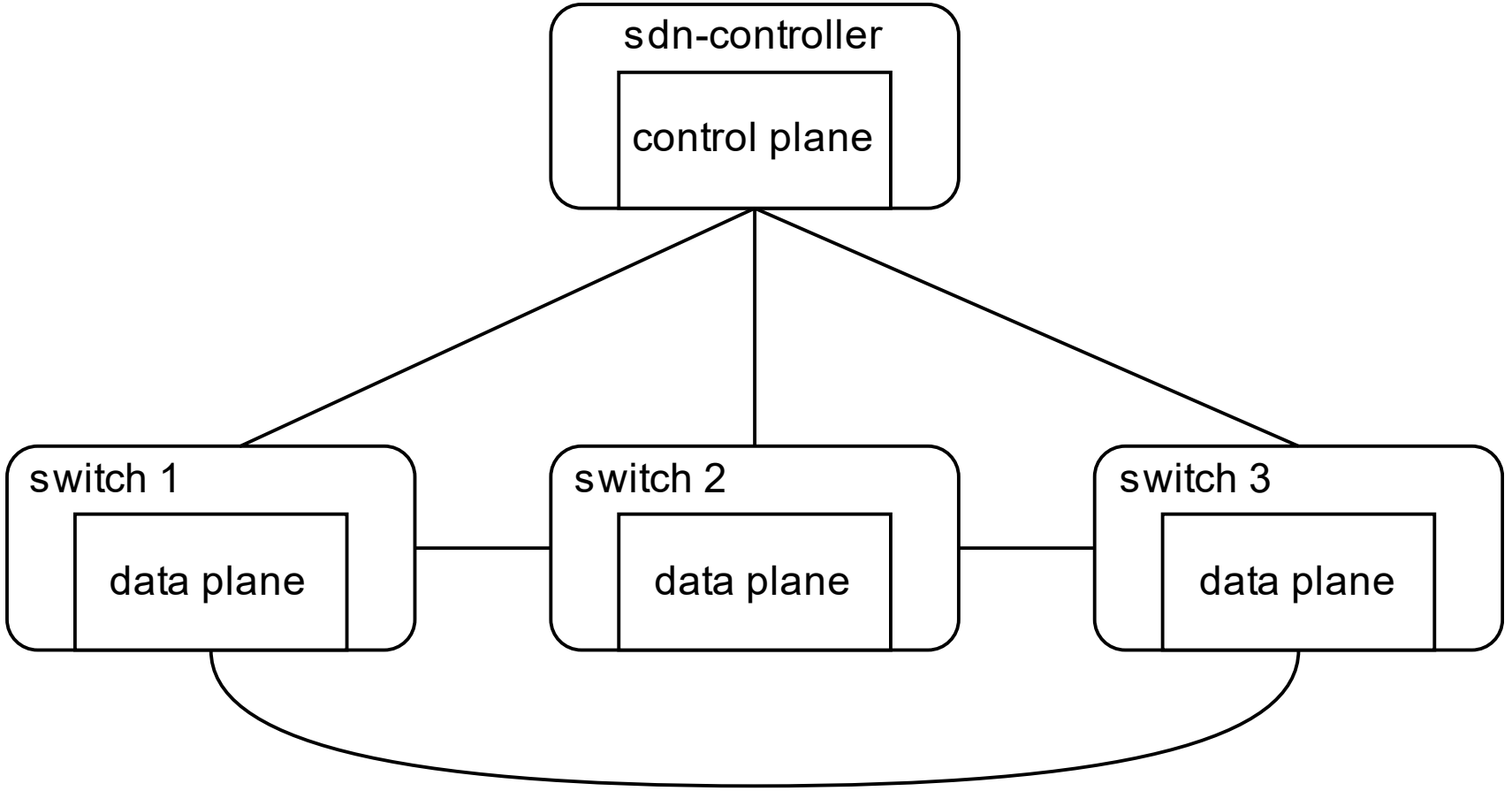
- Zielsetzung und Beitrag
- Hintergrund und Grundlagen
- Verwendete Verfahren
- Methodik und Implementierung
- Evaluation und Diskussion
- Fazit und Ausblick
- Abschluss

Zielsetzung und Beitrag

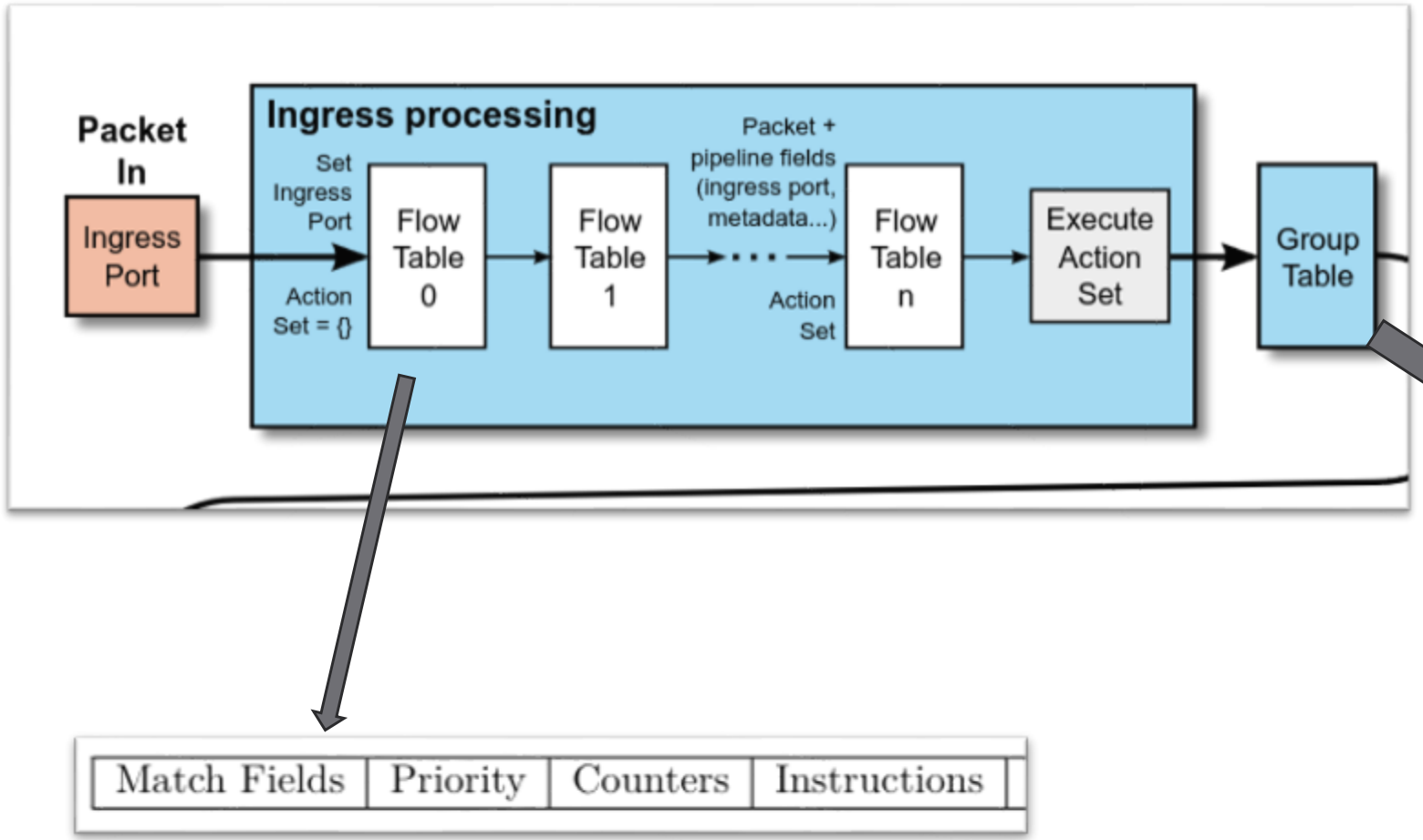
- Entwicklung eines Framework zu Evaluation torusbasierter Fast Failover Verfahren
- Realisierung der Verfahren mittels OpenFlow und der Fast Failover Gruppe in Mininet
- Auswertung unter verschiedener Ausfallmustern und Metriken

Hintergrund und Grundlagen

Software-defined Networking (SDN)

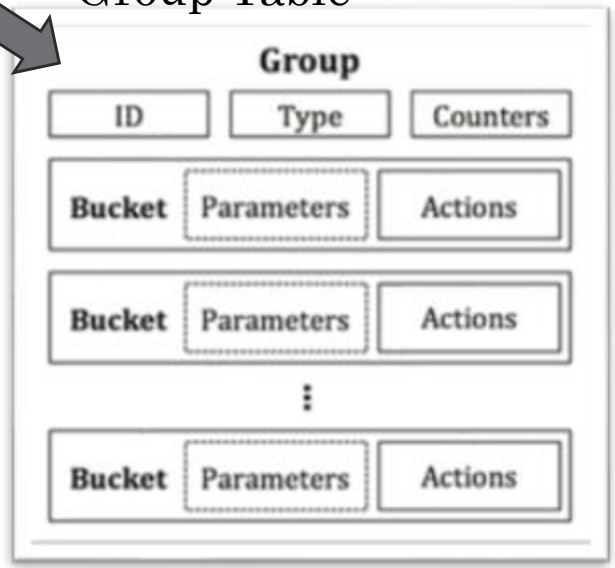


Openflow / Fast Failover Group



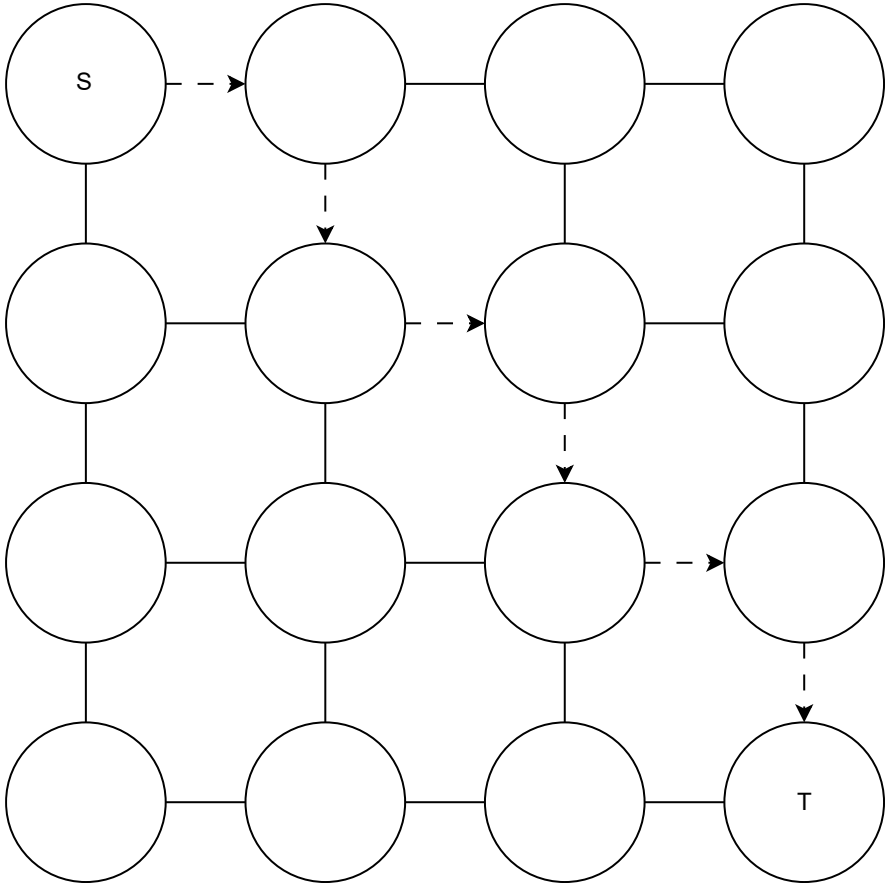
[Openflow Specification 1.5.1]

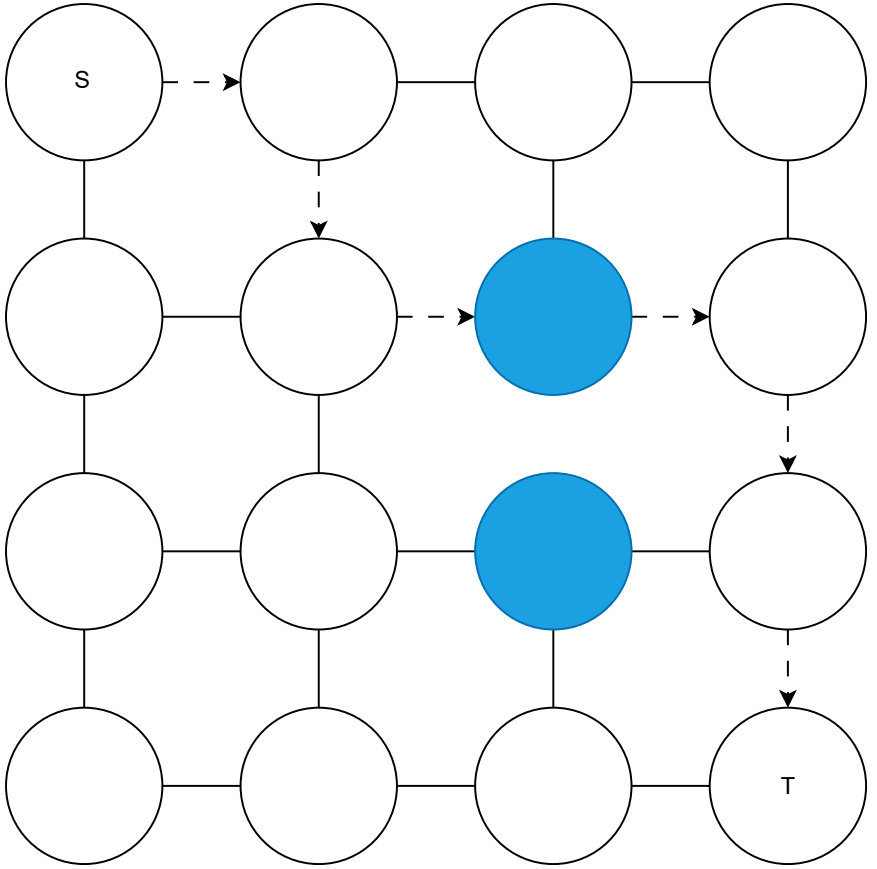
Group Table



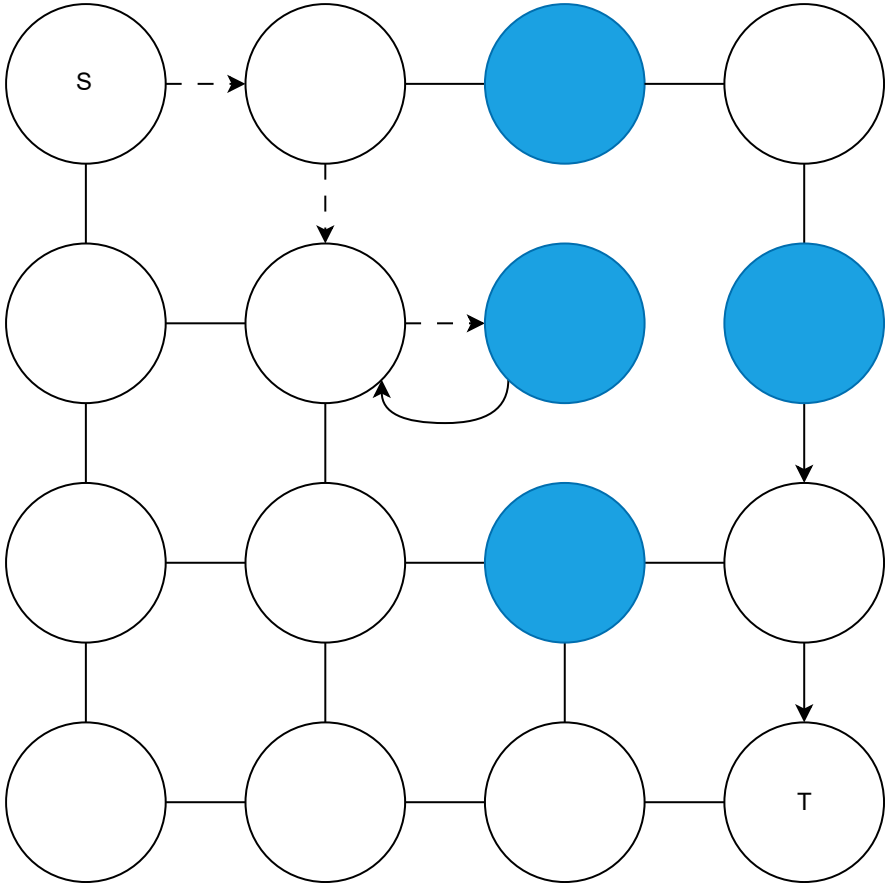
[By Bingdoal]

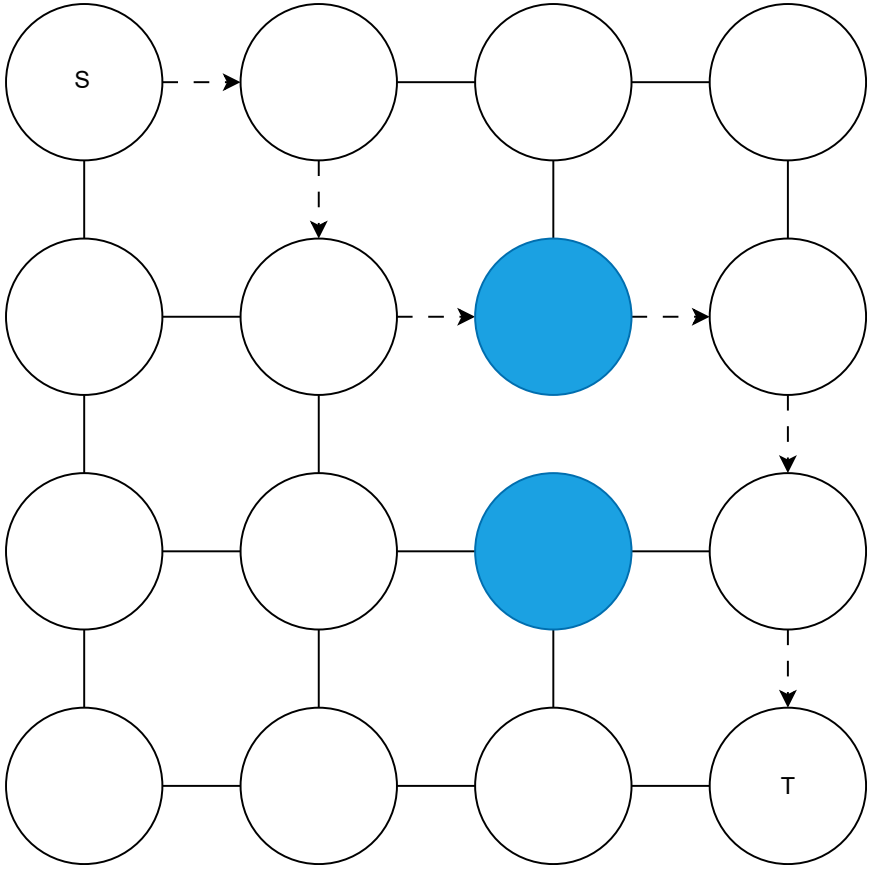
Resilienz schaffen: Fast Failover





Resilienz schaffen: Fast Failover

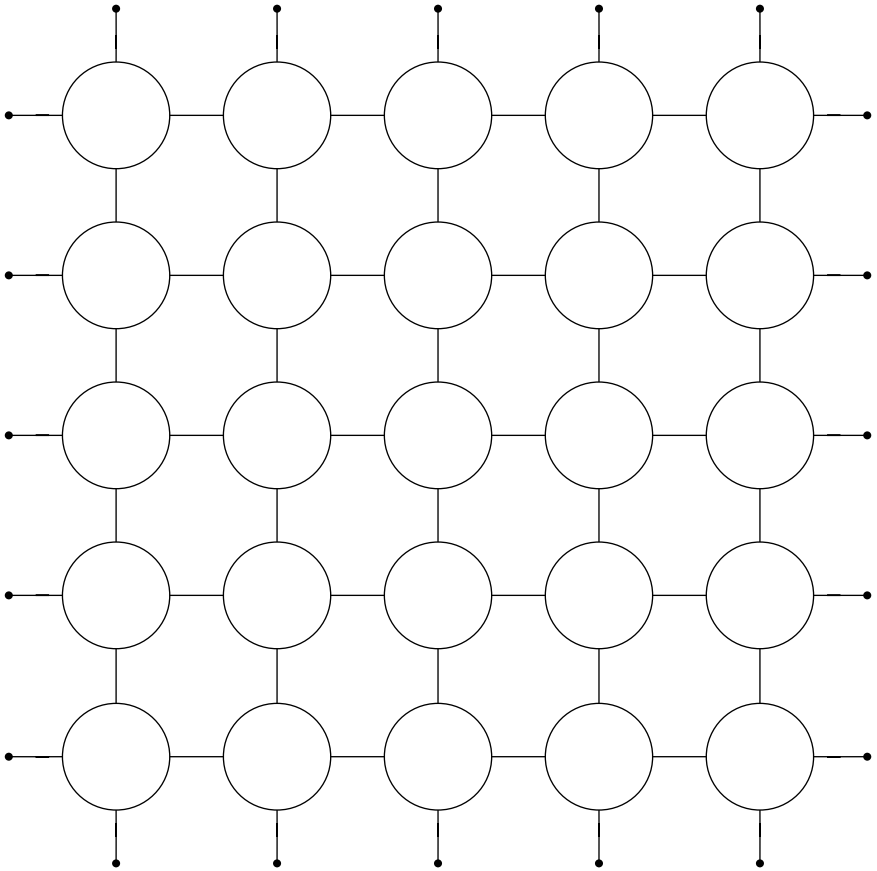


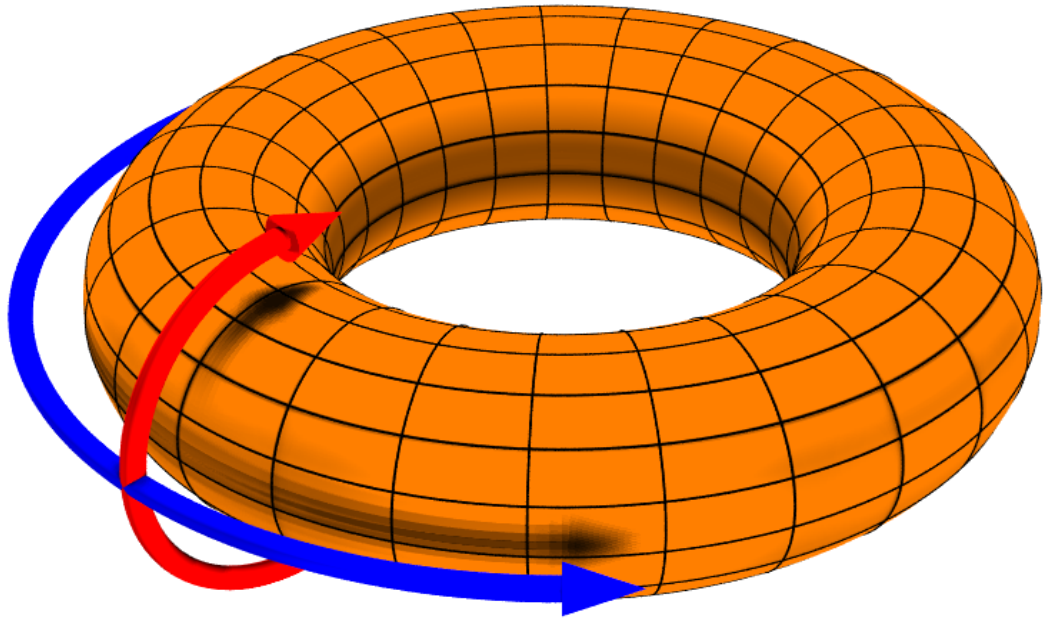


Resilienz schaffen: Fast Failover Modell

- Statische Tabellen
- Keine Duplikation oder Modifikation des Paketes
- Keine Zugriff auf eine Zufallsquelle

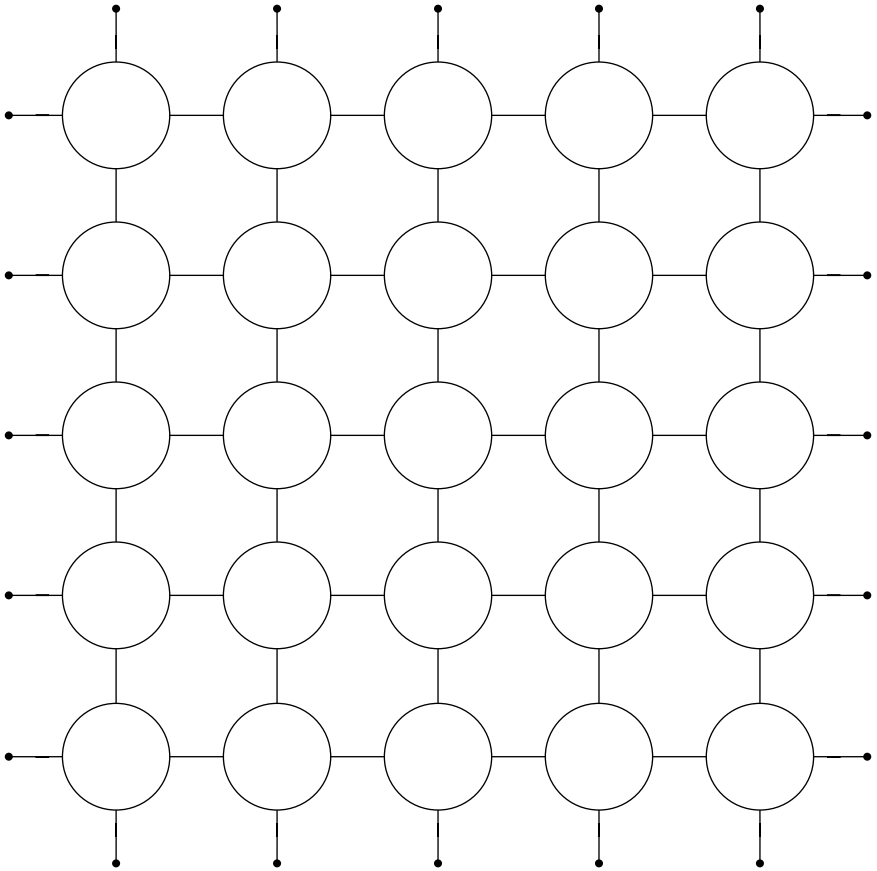
Ein Torus





[By DaveBurke - Own work, CC BY 2.5]

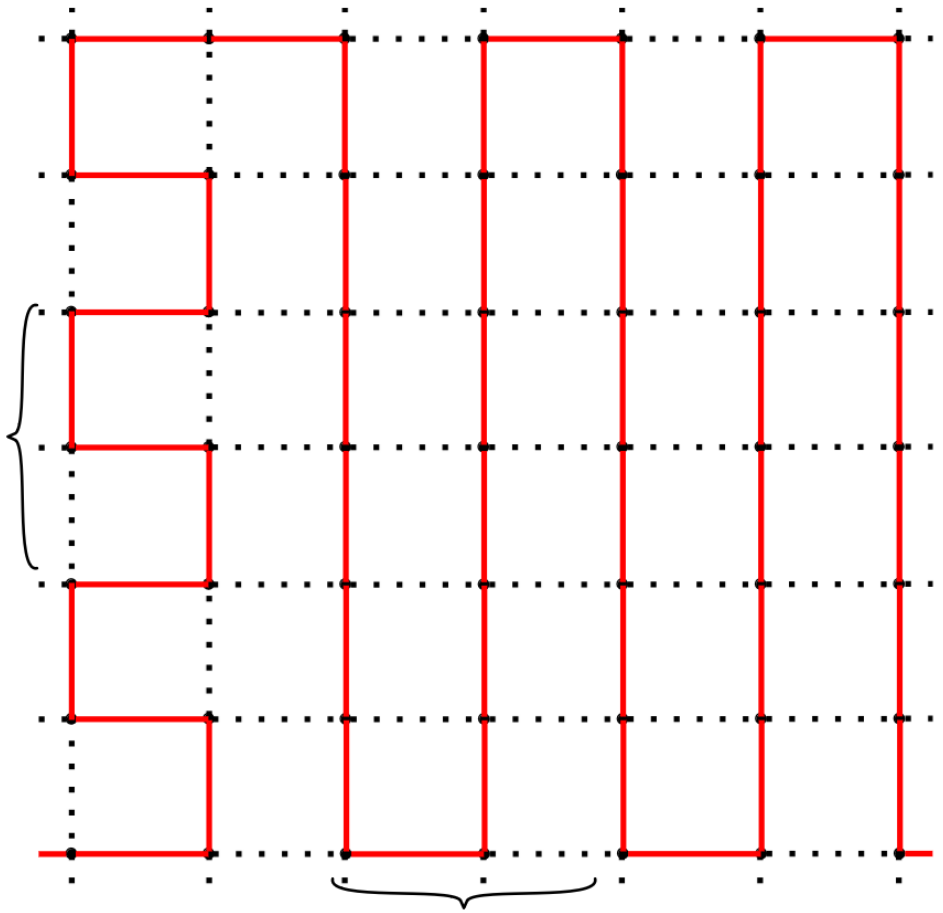
Ein Torus



- "Ideale" Resilienz von 3 Kanten
- Keine "Perfekte" Resilienz möglich
- Zwei kantendisjunkte Hamilton Zyklen

Vewendete Verfahren

"Exploring the Limits of Static Failover Routing" (Chiesa et al.)

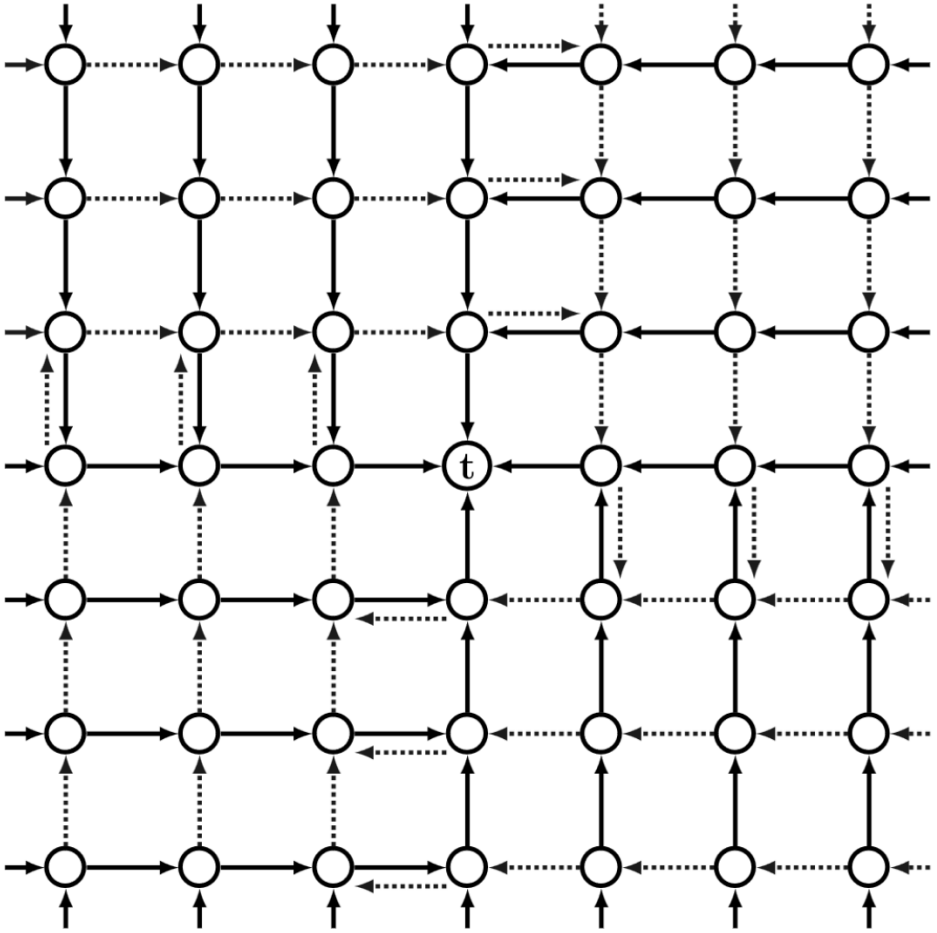


- Nur eingehende Kante wird verwendet
- Wege können über alle Knoten laufen
- Resilienz von 3

Zyklus	Distanz
H1_f	8
H1_b	17
H2_f	4
H2_b	21

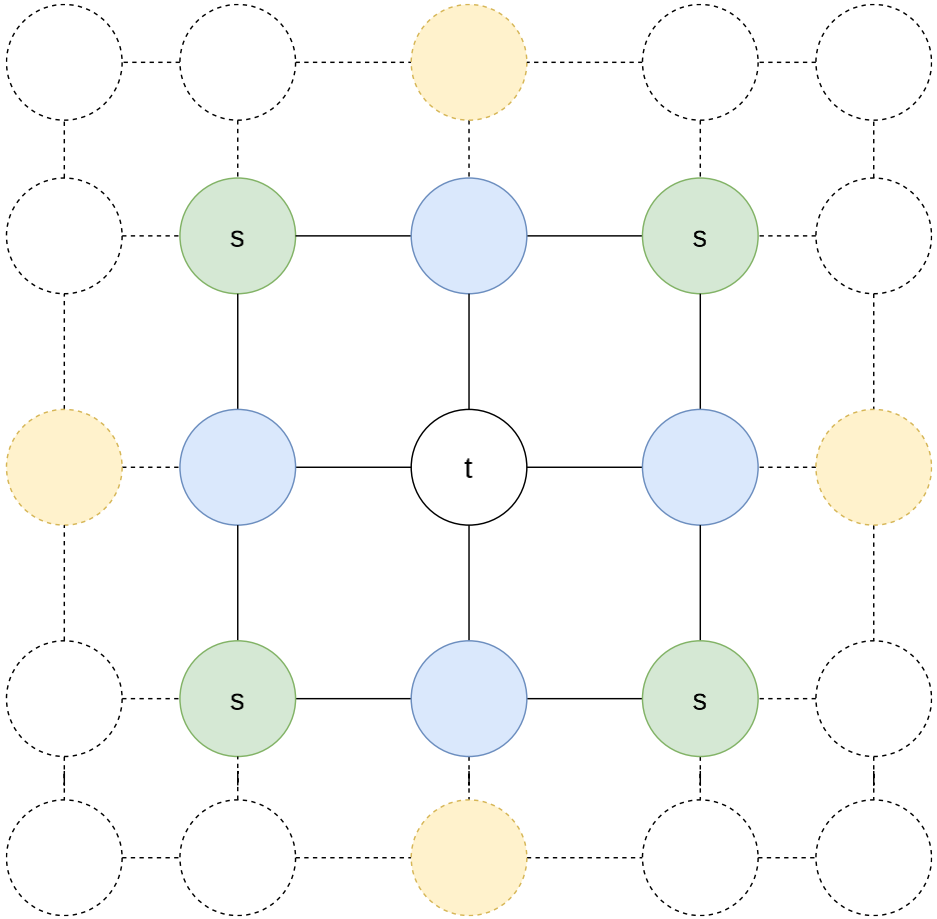
S0x0 => S 3x3 [5x5 Torus]

"Local Fast Failover Routing With Low Stretch" (Foerster et al.)



- Immer kürzester Weg ohne Ausfälle
- Resilienz von 3
- Eingehende Kante und Zieladresse wird verwendet

"On the Feasibility of Perfect Resilience with Local Fast Failover" (Foerster et al.)



Methodik und Implementierung

Netzwerkemulation

OpenVSwitch

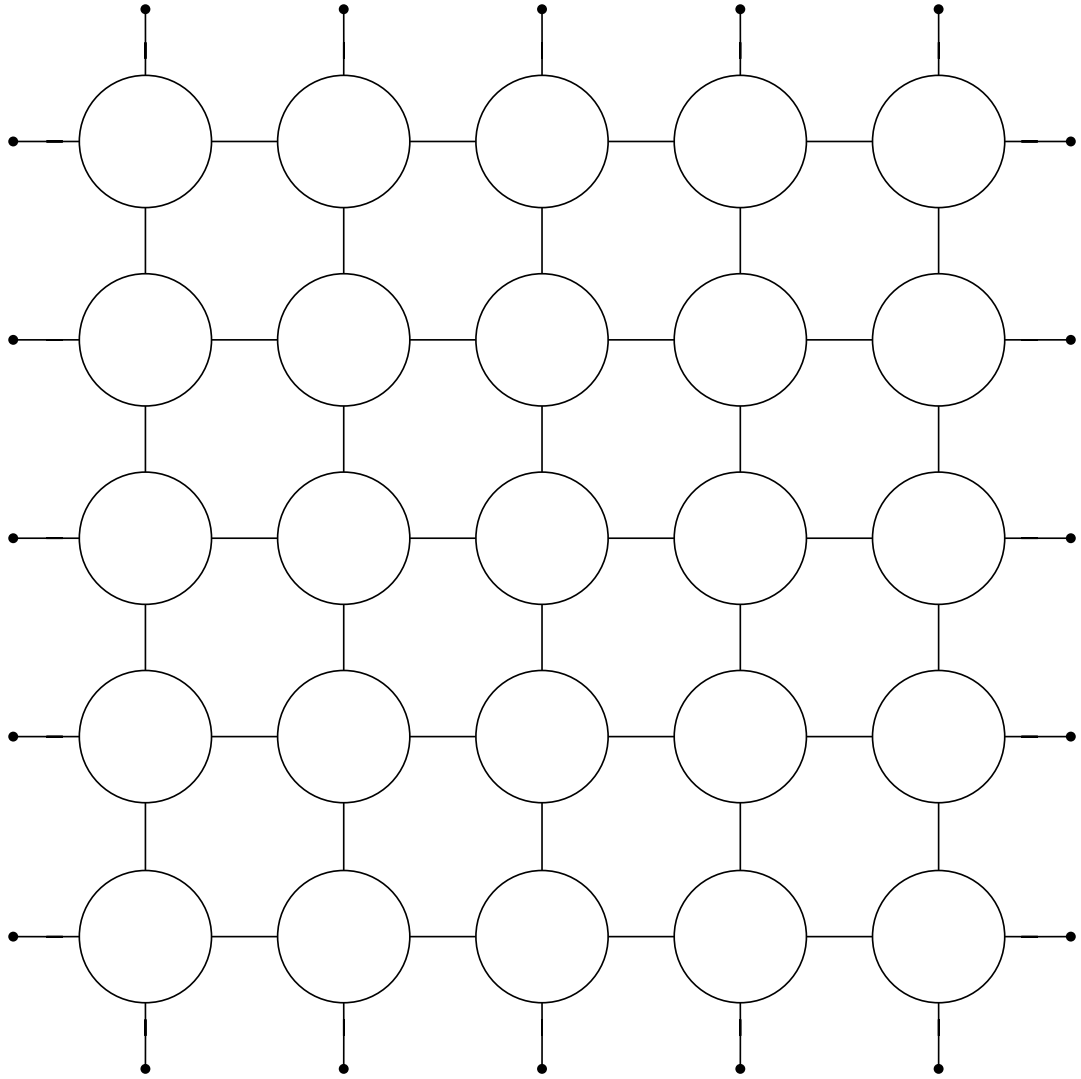
- State of the Art Software Switch
- Vollständiger OpenFlow support
- Aufteilung in Kernel-Modul und Userspace

Mininet

- containerbasierter (Net zwerk-)Emulator
- Virtuelle Hosts auf einem Kernel
- Python interface
- Verzögerung und Bandbreite mittels tc netem

Ausfallmuster

- Random Edge
- Random Node
- Clustered Failure
  - Parameter p und q
- Towards Destination



Erfassen von Messwerten: Latenz/Erreichbarkeit

- Erfassung mittels FPing (ICMP-Ping)
- Zwischen allen Paaren der Produktmenge der Knoten mehrere Stichproben
- Median der Stichproben
- Knotenpaar wird als erreichbar klassifiziert sofern eine Stichprobe zugestellt wird

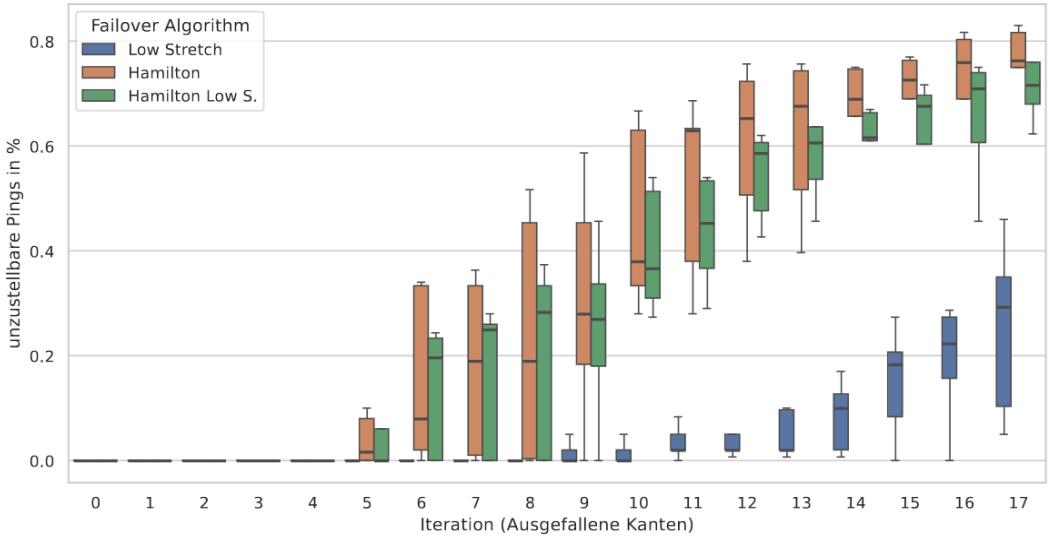
Erfassung von Messwerten: Durchsatz

- Erfassung mittels Iperf2
- Zwei Trafficpattern
  - Client Server Modell
  - Zufällige Paare
- TCP voll Duplex

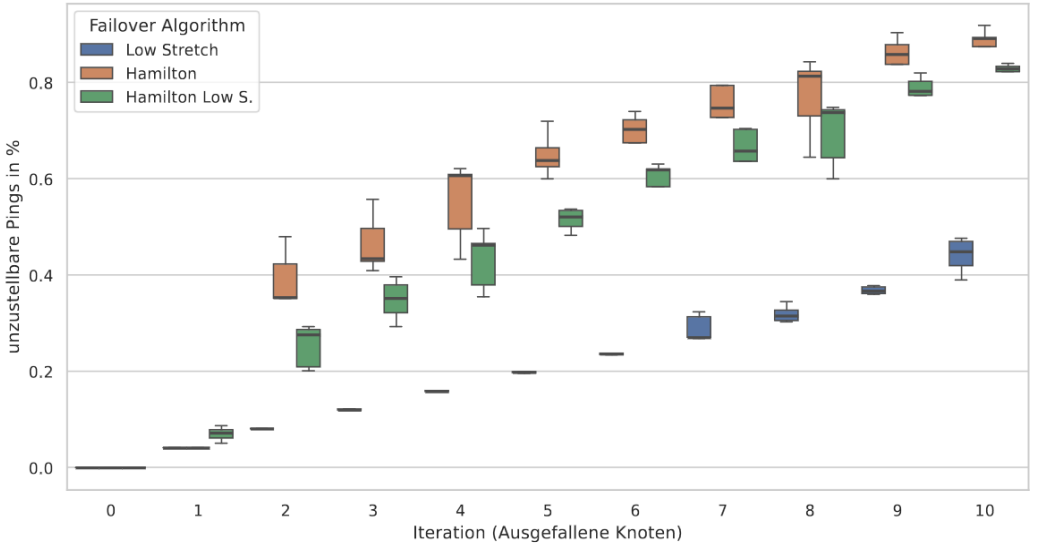
Evaluation und Diskussion

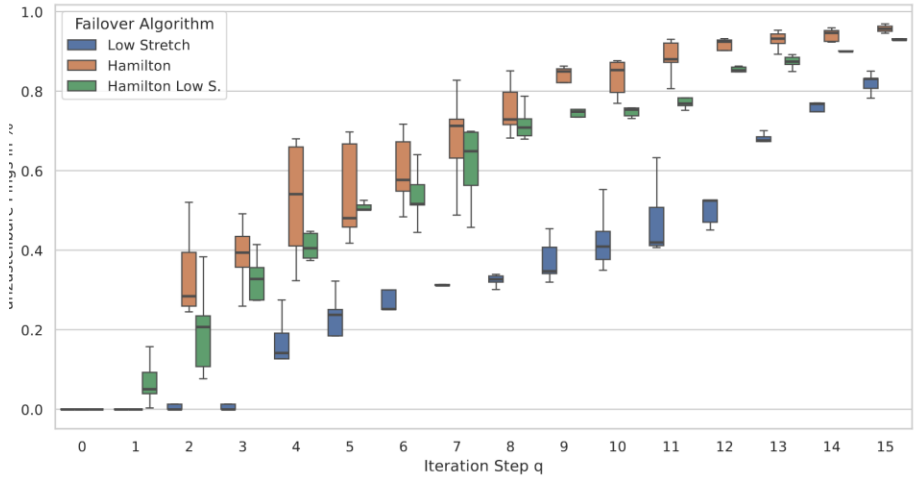
Evaluation Erreichbarkeit

Erreichbarkeit unter zufälligen Kantenausfälle:

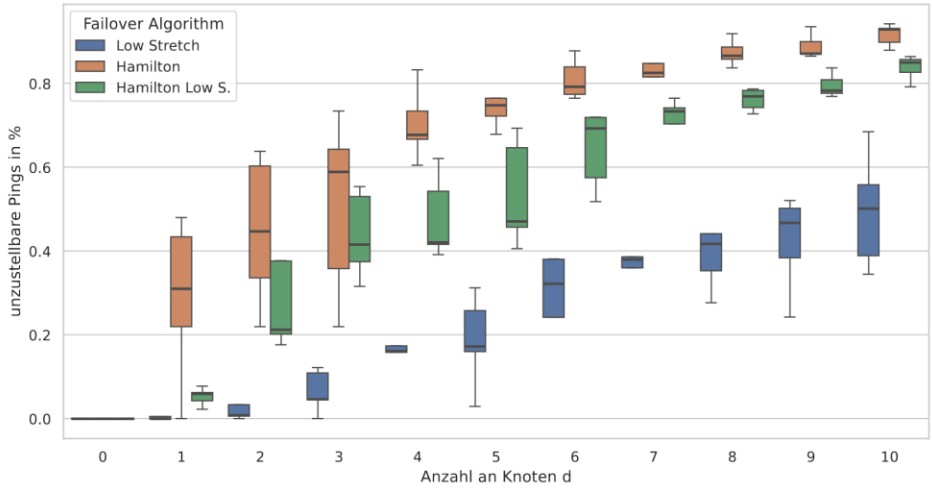


Erreichbarkeit unter zufälligen Knotenausfällen

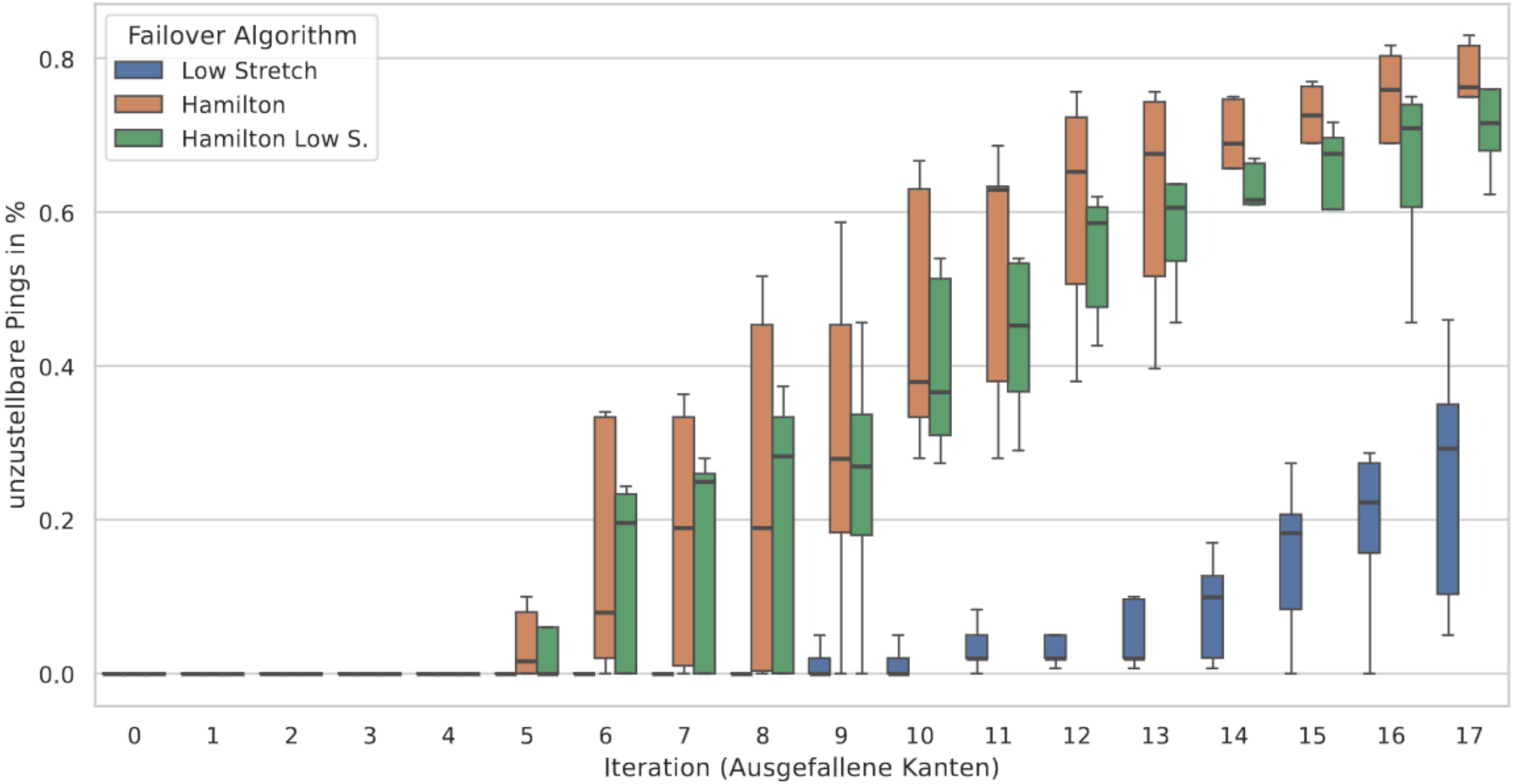




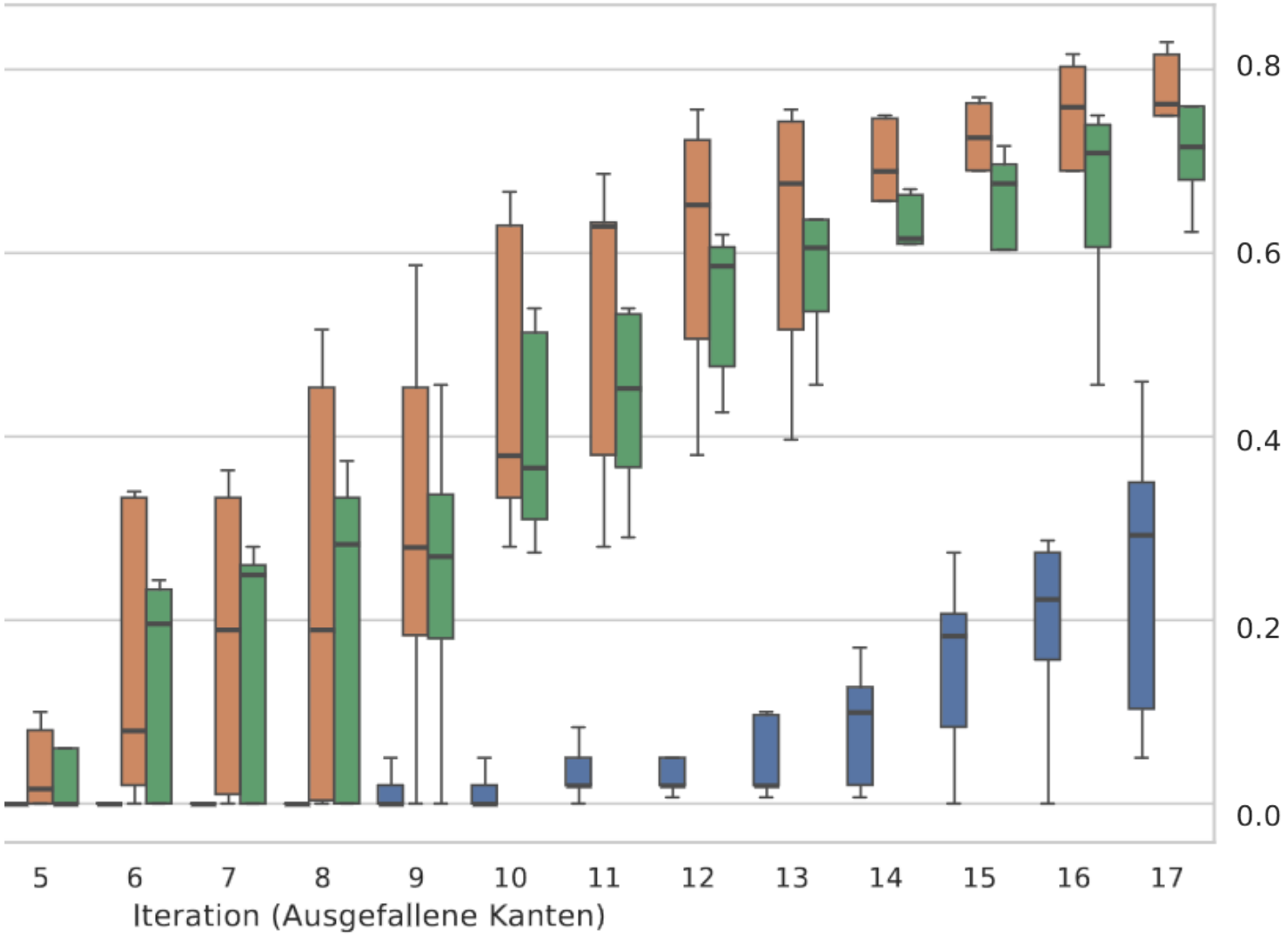
Erreichbarkeit unter Clusterfailure mit p = 0.9 und $q \in (0.3, 0.9)$



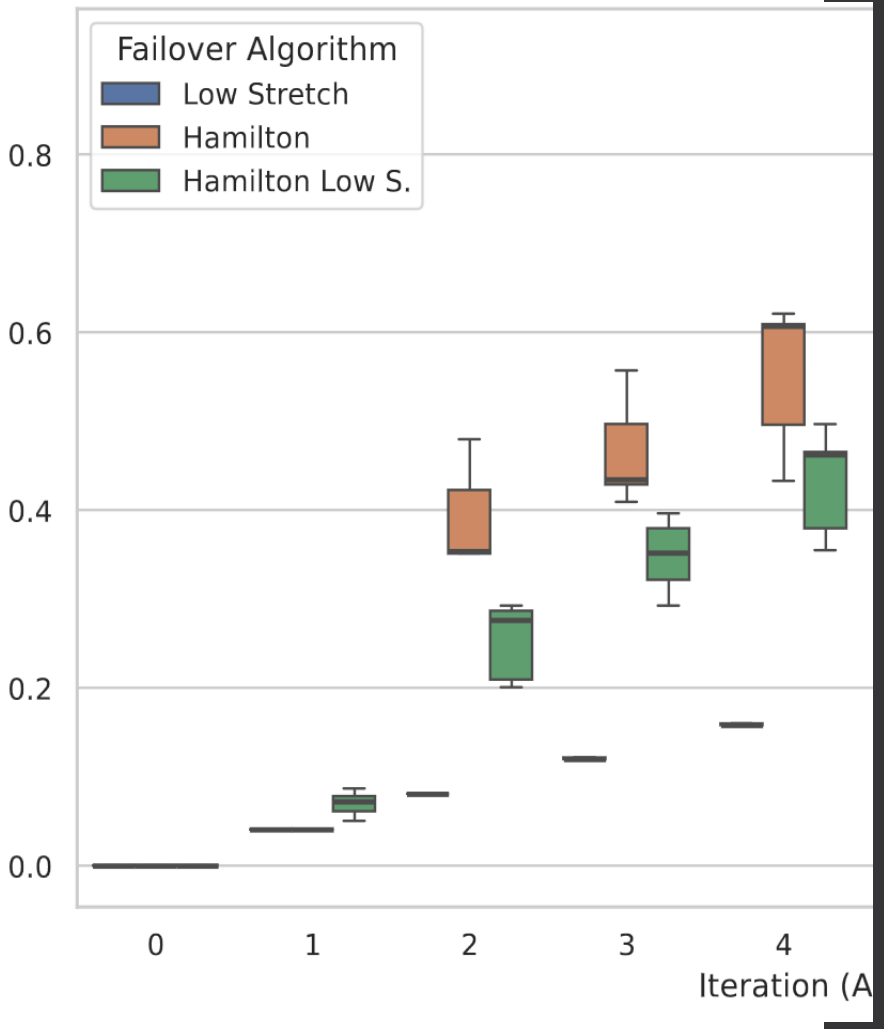
Erreichbarkeit unter Clusterfailure Node. p = 0.9 ∧ q = 0.34



Erreichbarkeit under zufälligen Kantenausfälle (lower is better); 5x5 Torus



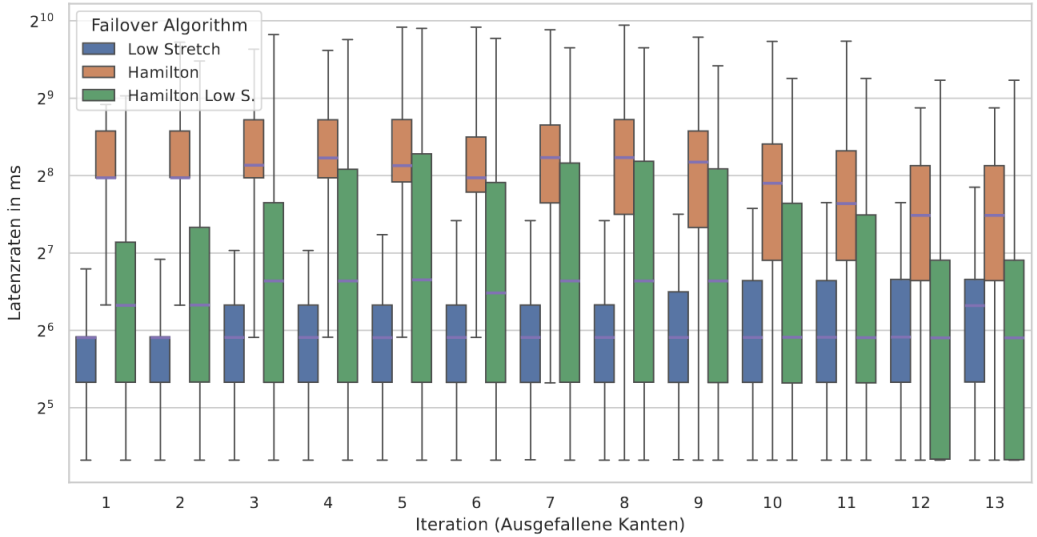
Erreichbarkeit unter zufälligen Kantenausfälle (lower is better); 5x5 Torus

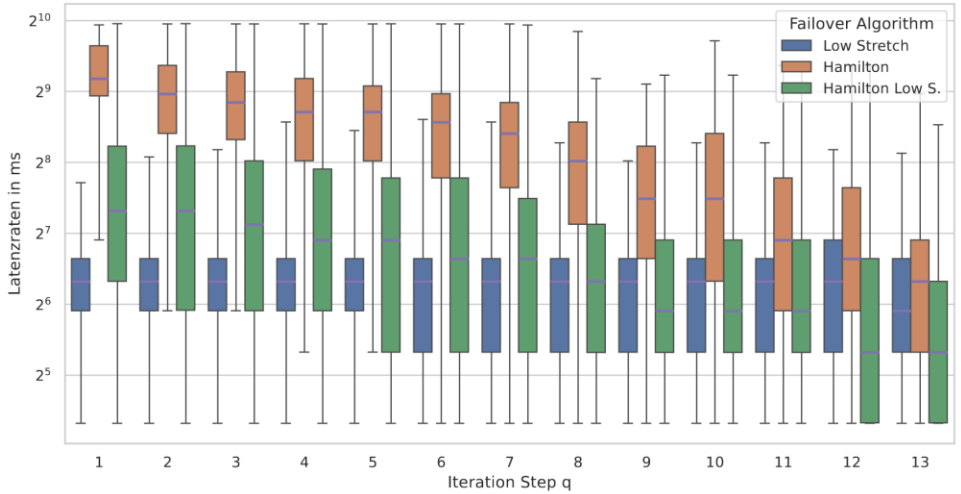


Erreichbarkeit unter zufälligen Knotenausfällen. 7 × 7 Torus.

Evaluation Latenz

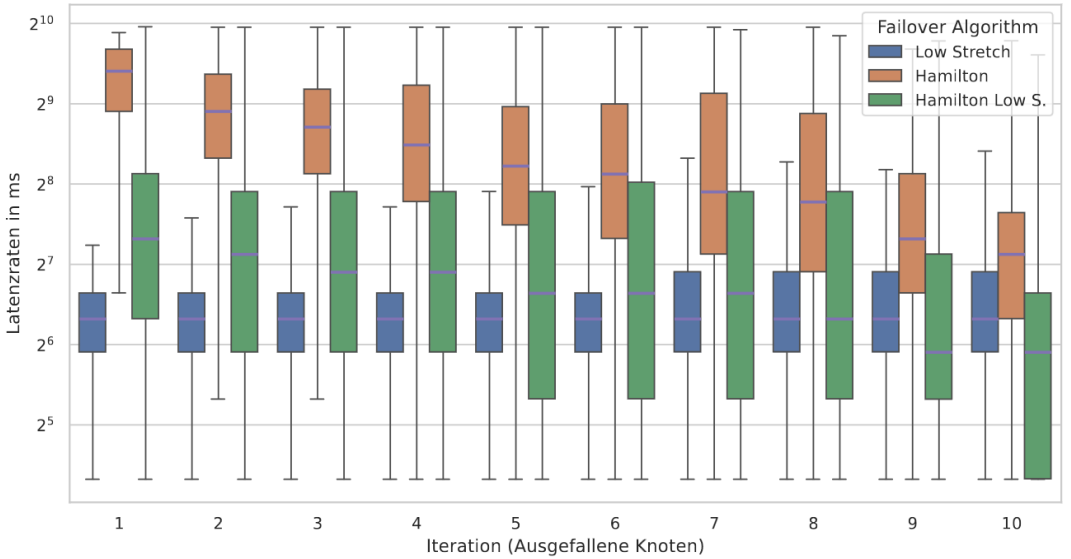
Latenzen unter zufälligen Kantenausfällen

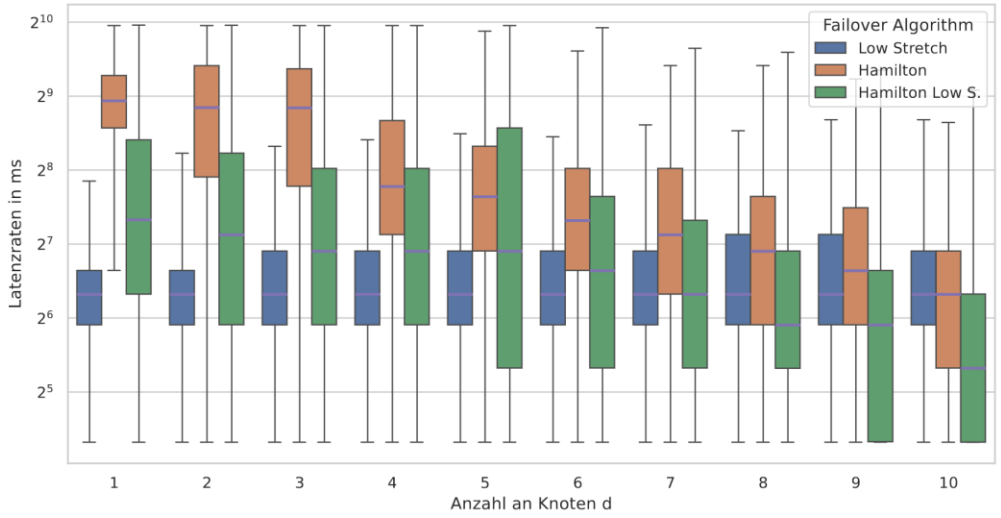




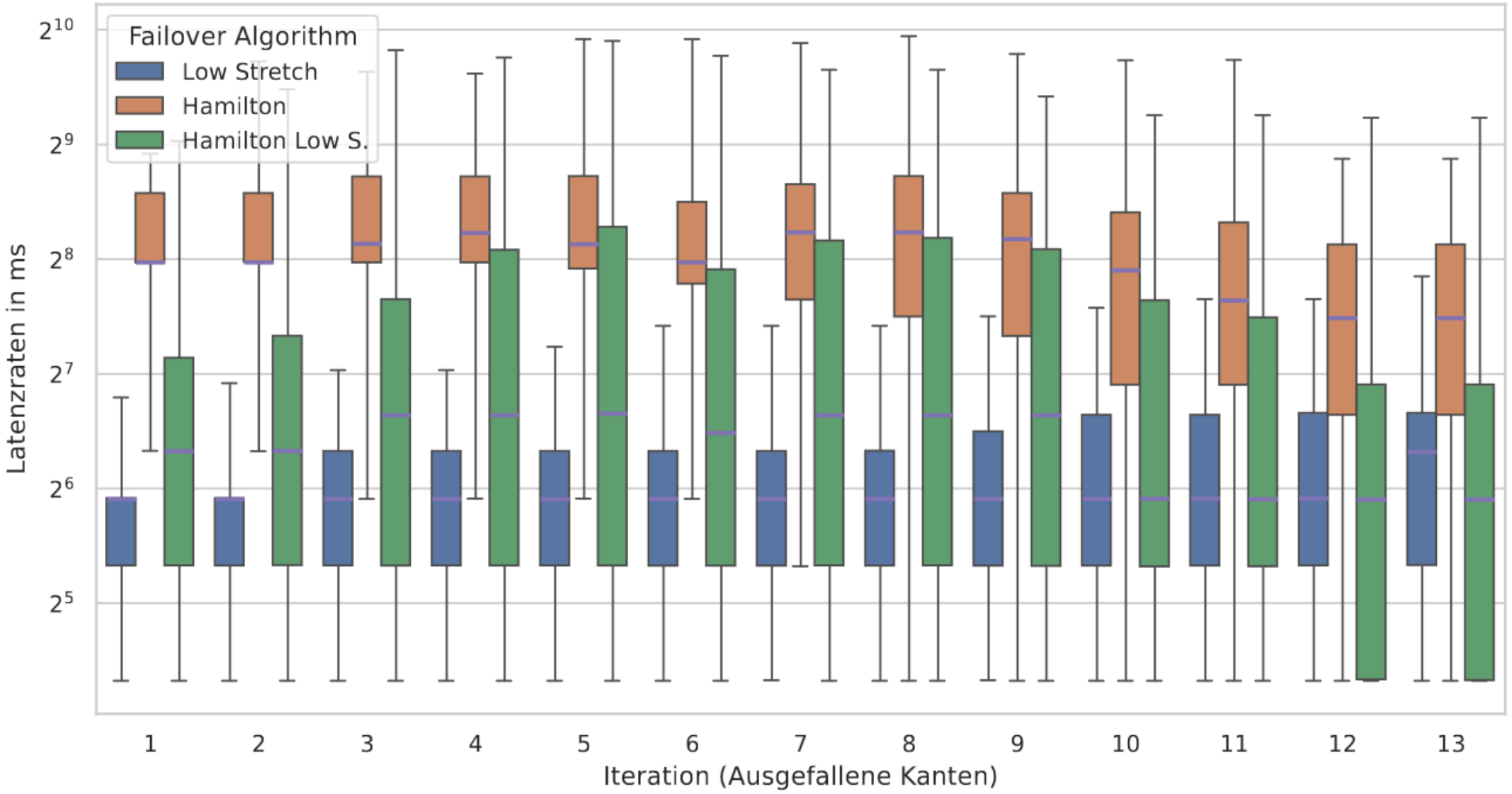
Latenzen unter Clusterfailure mit $p = 0.9$ und $q \in (0.3, 0.9)$

Latenzen unter zufälligen Knotenausfällen





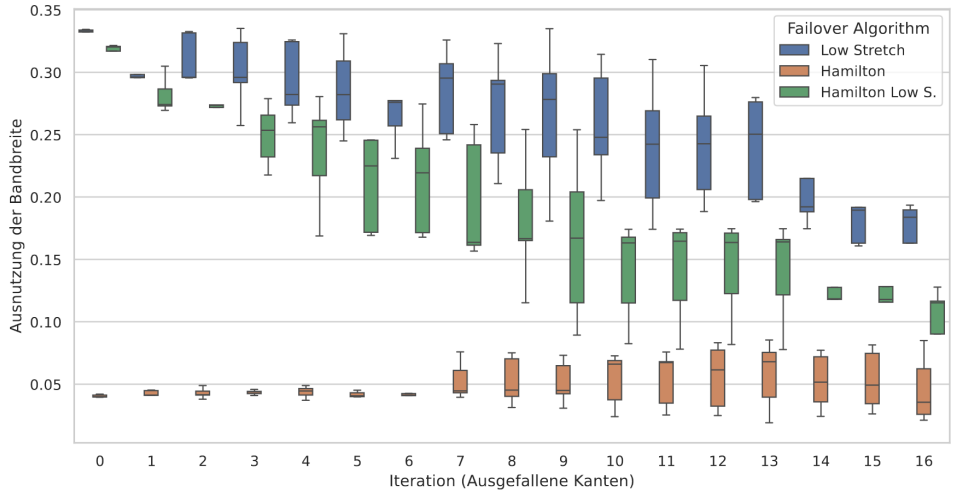
Latenzen unter Clusterfailure Node. $p = 0.9 \wedge q = 0.34$

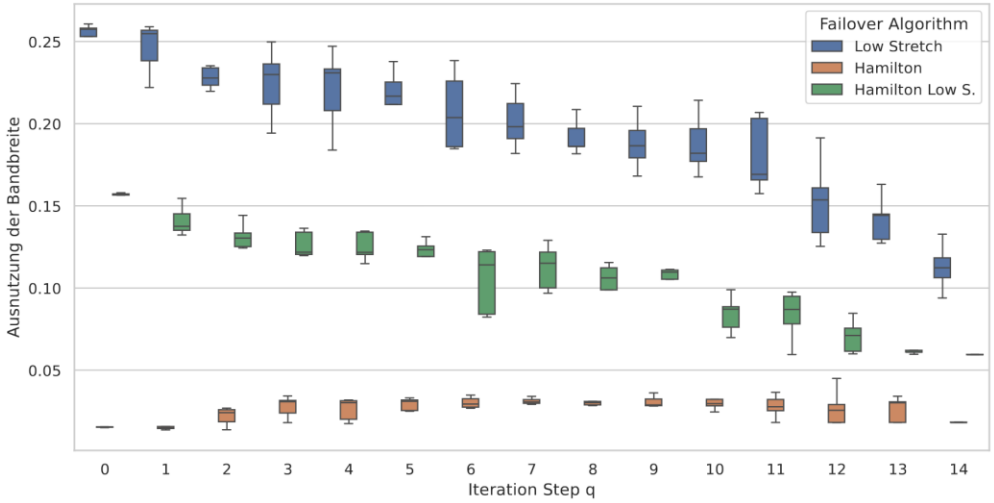


Latenz und zufälligen Kantenausfälle (lower is better);
5x5 Torus

Evaluation: Durchsatz

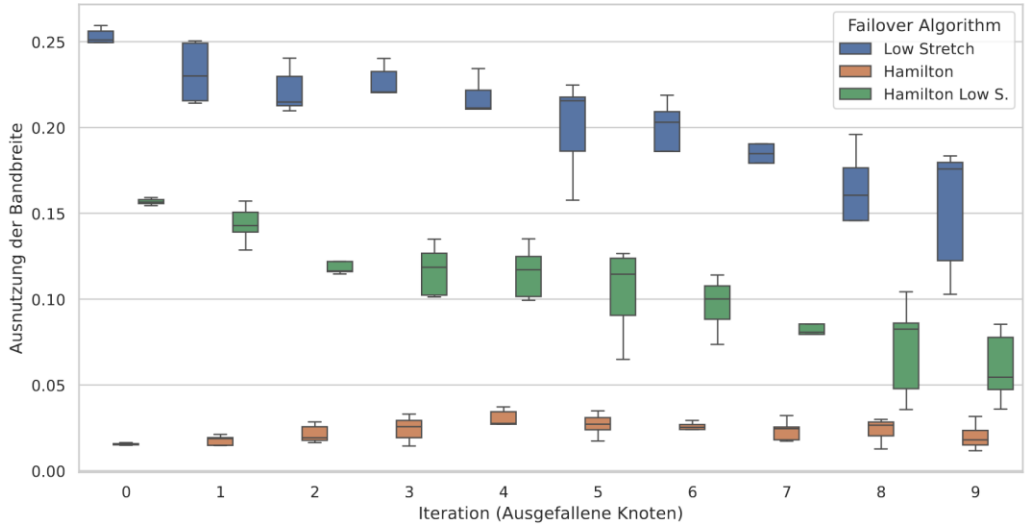
Durchsatz unter zufälligen Kanten Ausfällen (zufällige Paare)

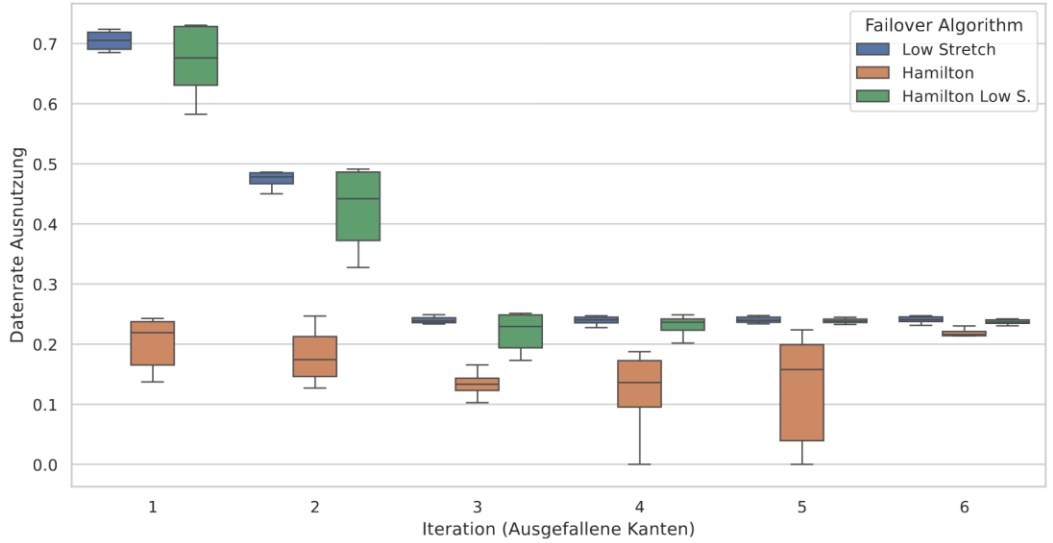




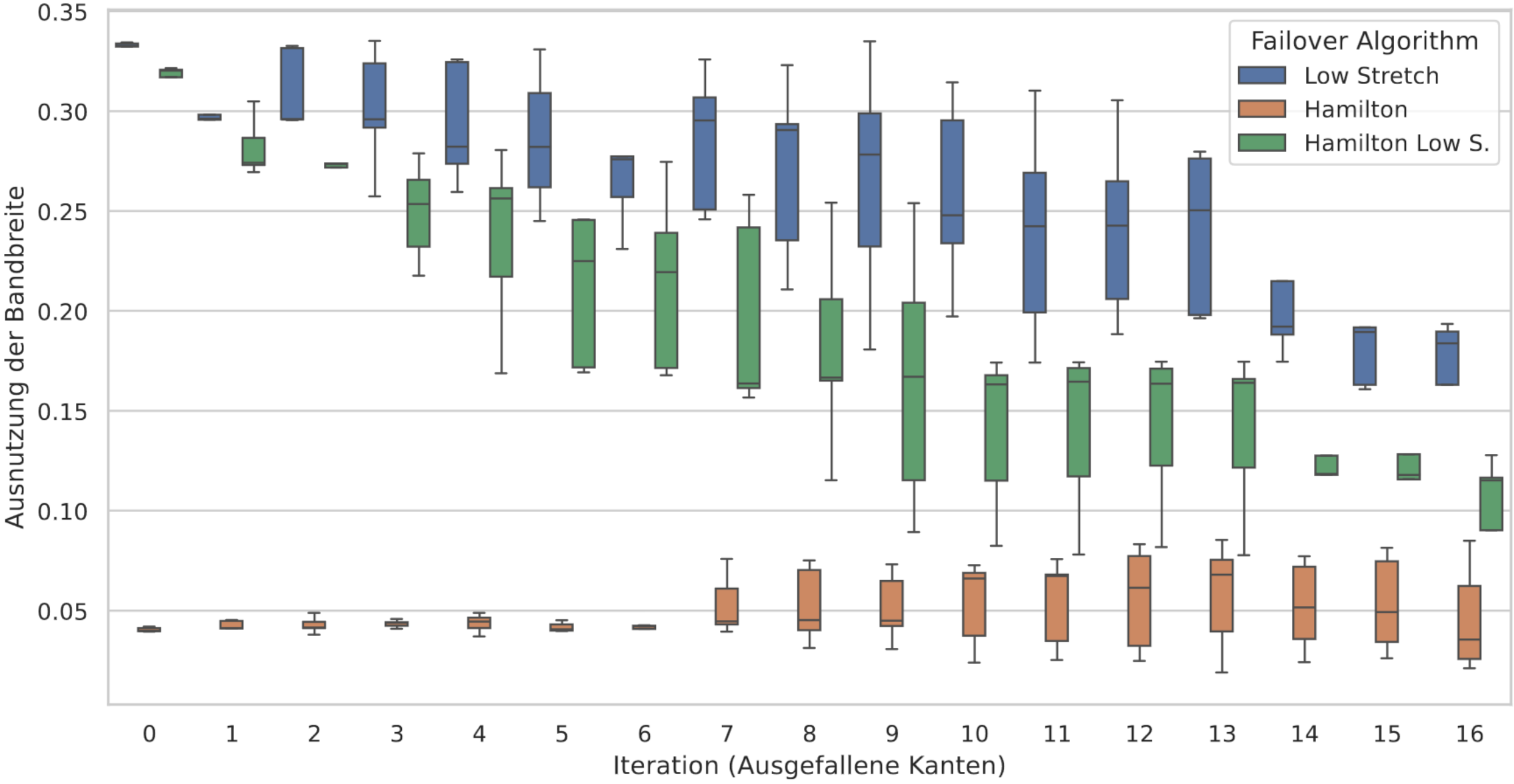
Durchsatz unter Clusterfailure mit $p = 0.9$ und $q \in (0.3, 0.9)$

Durchsatz unter zufälligen Knoten Ausfällen (zufällige Paare)





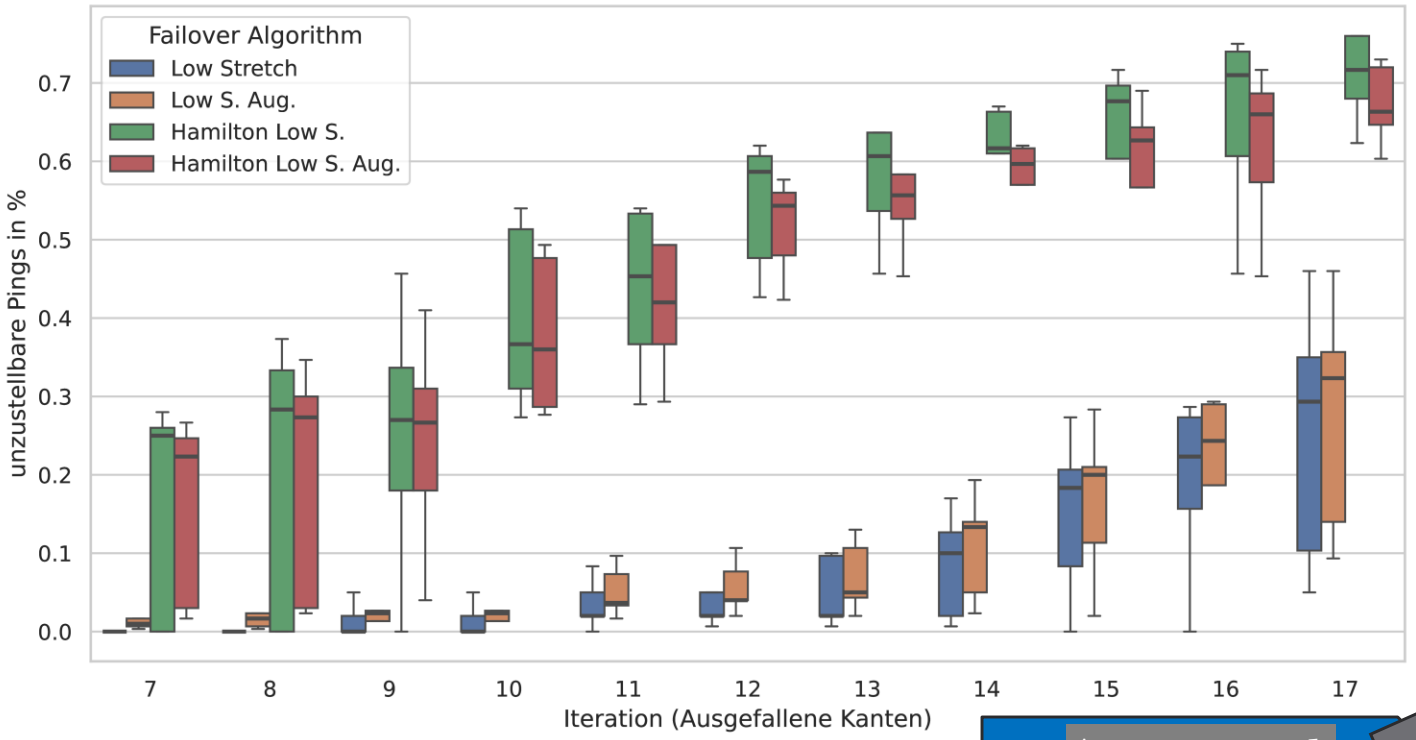
Durchsatz unter Towards Destination (Client Server)

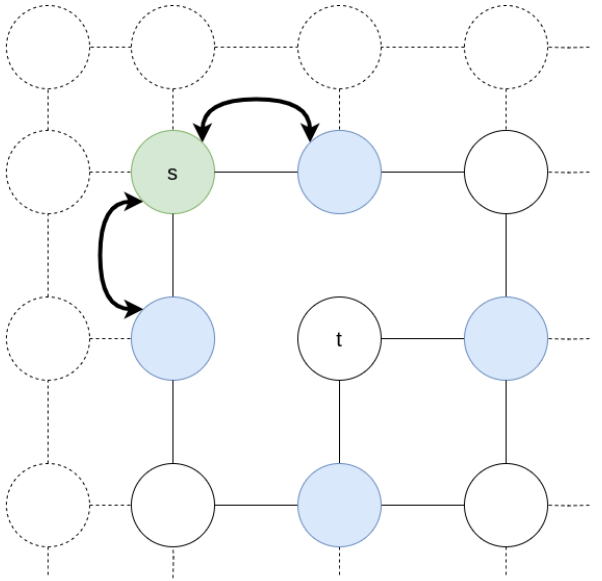


Durchsatz unter zufälligen Kanten Ausfällen; 5×5 Torus; 9 Paare

Evaluation: Augmentation

Erreichbarkeit und zufälligen Kantenausfälle; 5x5 Torus





Augmented Low Stretch

```
mininetFailOver> h0x0 fping h1x1
10.0.0.5 is unreachable
```

Low Stretch

```
mininetFailOver> h0x0 fping h1x1
10.0.0.5 is alive
```

Fazit und Ausblick

Fazit

- Das Low Stretch Verfahren dominiert
- Kurze Routen sind vorteilhaft
- Fast Failure Groups sind praktikabel aber zur Evaluation ggf. übermäßig komplex
- Fast Failure kann die Resilienz erheblich steigern und ist auch opportun außerhalb fester theoretischer Garantien

Ausblick

- Evaluation mittels größerer Datensätze
- Erweiterte Topologien
- Andere Verfahren
- Komplexere Experimente
- Suche nach gezielten Ausfallmustern

Vielen Dank für Ihre Aufmerksamkeit!

Implementierung