

# Abschlusspräsentation Projekt 2

---

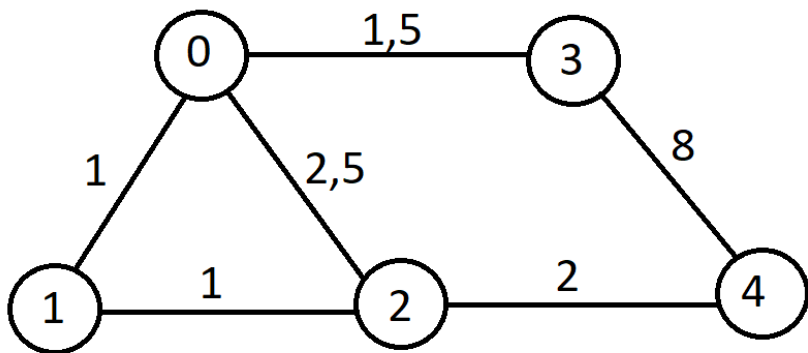
Pouria Araghchi 170468, Kai Lukas Ilmenau 225338, Naveed Niazi 214471

June 29, 2023

TU Dortmund - Fachprojekt zu "Routingalgorithmen"

Kai

---



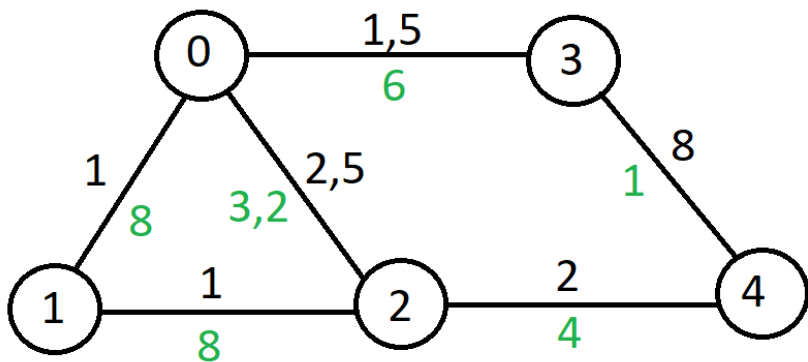
Capacity

InverseCap.

ClosenessInvCap.

ClosenessCentr.

## Inverse Capacity



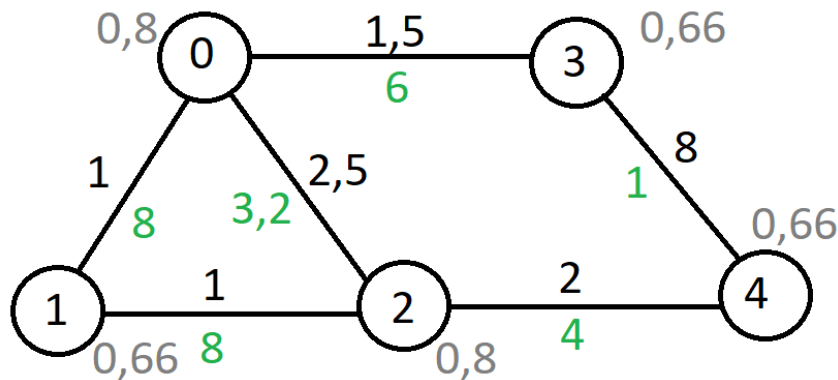
Capacity

InverseCap.

ClosenessInvCap.

ClosenessCentr.

## Closeness Centrality Werte



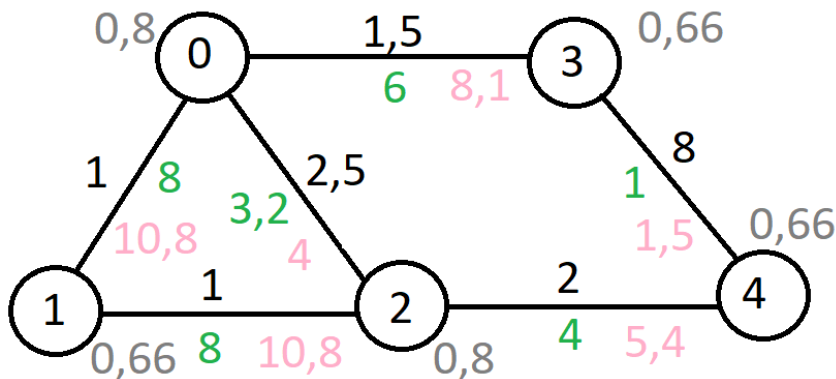
Capacity

InverseCap.

ClosenessInvCap.

ClosenessCentr.

## Closeness Centrality With Inverse Capacity



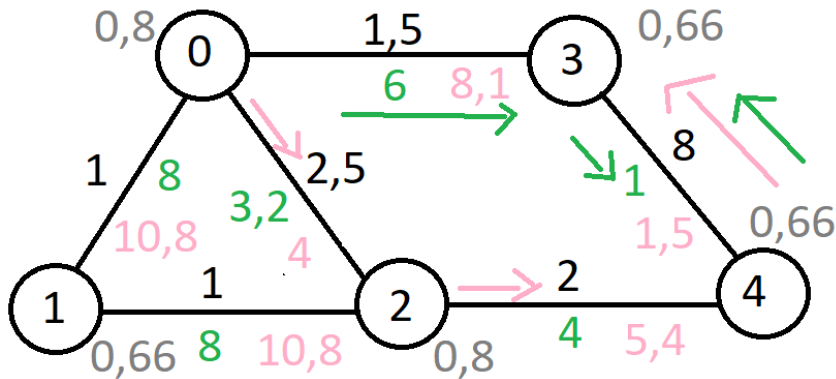
Capacity

InverseCap.

ClosenessInvCap.

ClosenessCentr.

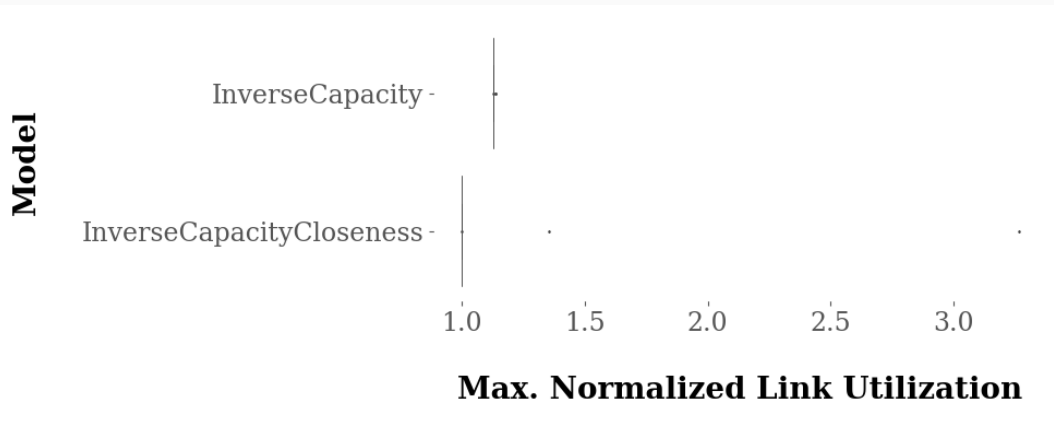
## Mit Demands



Capacity  
InverseCap.  
ClosenessInvCap.  
ClosenessCentr.

Demands von 0 nach 4 (eine Capacity-Einheit)  
und von 4 nach 3 (acht Capacity-Einheiten)

# Boxplots





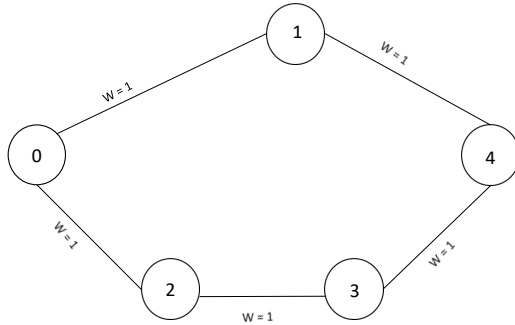
## Ist meine Algorithmus besser?

- in diesem speziellen Setting: Ja
- "designed" Topologie
  - lange Suche nach guter Topologie für den Algorithmus
- Siehe Projekt 1: *Inverse\_Capacity* im Durchschnitt besser als meine Anpassung

Naveed

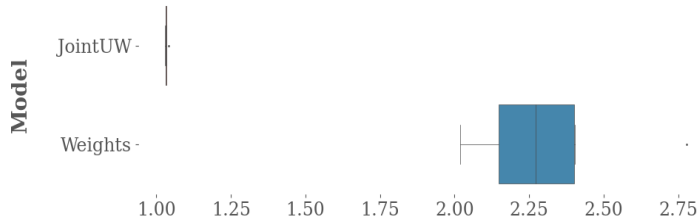
---

## Topologie:



# Ergebnisse

```
100 ; SimpleJointUW.topo.sh ; 1.030732232
101 ; Weights.topo.sh ; 2.27279644
102 ; SimpleJointUW.topo.sh ; 1.029772264
103 ; Weights.topo.sh ; 2.2709963626666667
104 ; SimpleJointUW.topo.sh ; 1.04189532
105 ; Weights.topo.sh ; 2.018401589333333
106 ; SimpleJointUW.topo.sh ; 1.0294664186666667
107 ; Weights.topo.sh ; 2.2703404586666664
100 ; SimpleJointUW.topo.sh ; 1.03383732
101 ; Weights.topo.sh ; 2.2745359973333334
102 ; SimpleJointUW.topo.sh ; 1.0317545653333333
103 ; Weights.topo.sh ; 2.1461416213333333
104 ; SimpleJointUW.topo.sh ; 1.0336421573333334
105 ; Weights.topo.sh ; 2.399926333333333
106 ; SimpleJointUW.topo.sh ; 1.0312037733333332
107 ; Weights.topo.sh ; 2.402300864
100 ; SimpleJointUW.topo.sh ; 1.0311207813333334
101 ; Weights.topo.sh ; 2.1489939546666665
102 ; SimpleJointUW.topo.sh ; 1.033397992
103 ; Weights.topo.sh ; 2.0217069786666667
104 ; SimpleJointUW.topo.sh ; 1.0328527973333332
105 ; Weights.topo.sh ; 2.404023616
106 ; SimpleJointUW.topo.sh ; 1.0313898426666666
107 ; Weights.topo.sh ; 2.7790064053333334
```



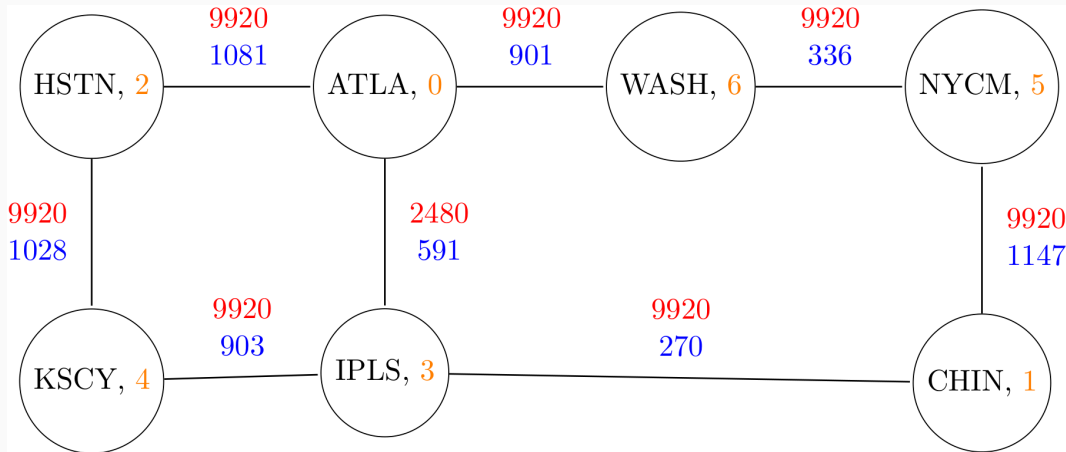
Max. Normalized Link Utilization

```
100_SimpleJointUW.topo.sh 107_Weights.topo.sh joint.json nanonet-master
101_Weights.topo.sh addr.py Joint.topo.py net.py
102_SimpleJointUW.topo.sh batch_result001.csv Joint.topo.sh node.py
103_Weights.topo.sh build.py jointUW.json __pycache__
104_SimpleJointUW.topo.sh Figure_1.png JointUW.topo.py README.md
105_Weights.topo.sh gen_boxplot.py json route.py
106_SimpleJointUW.topo.sh imgs nanonet_batch.py SimpleJointUW.
(base) linuxmint@linuxmint203:~/naveed$ sudo python3 gen_boxplot.py
JOINTUW Median: 1.031572204
WEIGHTS Median: 2.2718964013333336
JOINTUW Minimum: 1.0294664186666667
WEIGHTS Minimum: 2.018401589333333
JOINTUW Maximum: 1.04189532
WEIGHTS Maximum: 2.7790064053333334
(base) linuxmint@linuxmint203:~/naveed$
```

Sequential Combination aus  
InverseCapacity und  
DemandFirstWaypoints

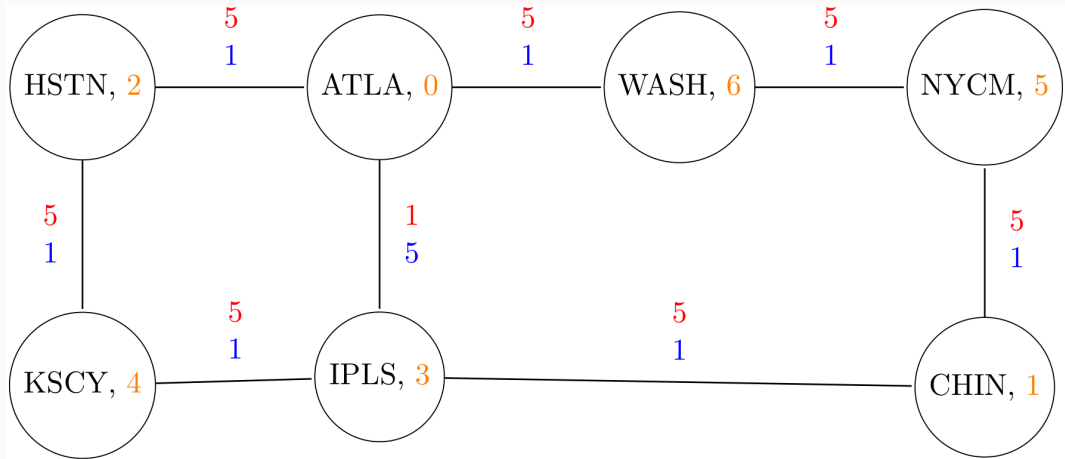
---

# Netzwerktopologie (Anlehnung an Abilene)<sup>1</sup>



<sup>1</sup>FaPro\_P2/pouria/network\_origin.pdf

## Anwendung von InverseCapacity<sup>2</sup>



<sup>2</sup>FaPro\_P2/pouria/network\_origin.pdf

## Demandtabelle (vereinfacht)<sup>3</sup>

↓ von, nach →	HSTN	KSCY	ATLA	IPLS	CHIN	WASH	NYCM
HSTN		1	13	2	6	8	3
KSCY	1		1	2	4	2	2
ATLA	28	1		5	3	19	4
IPLS	7	2	4		14	7	9
CHIN	165	17	18	7		11	12
WASH	13	5	13	5	17		20
NYCM	16	3	9	4	61	24	

---

<sup>3</sup>FaPro\_P2/pouria/network\_origin.pdf



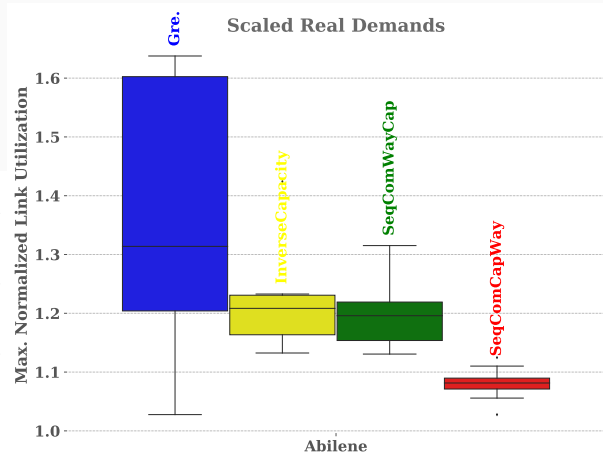
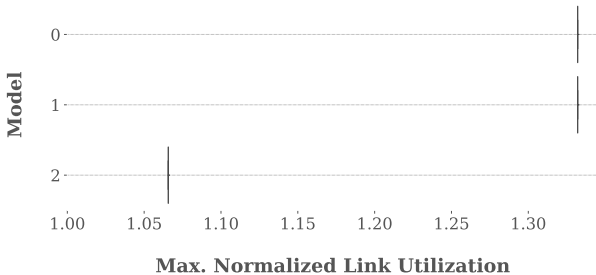
## Demandtabelle (stärker vereinfacht)<sup>4</sup>

↓ von, nach →	HSTN	KSCY	ATLA	IPLS	CHIN	WASH	NYCM
HSTN							
KSCY							
ATLA				5			
IPLS						7	
CHIN							
WASH							
NYCM							

---

<sup>4</sup>FaPro\_P2/pouria/network\_origin.pdf

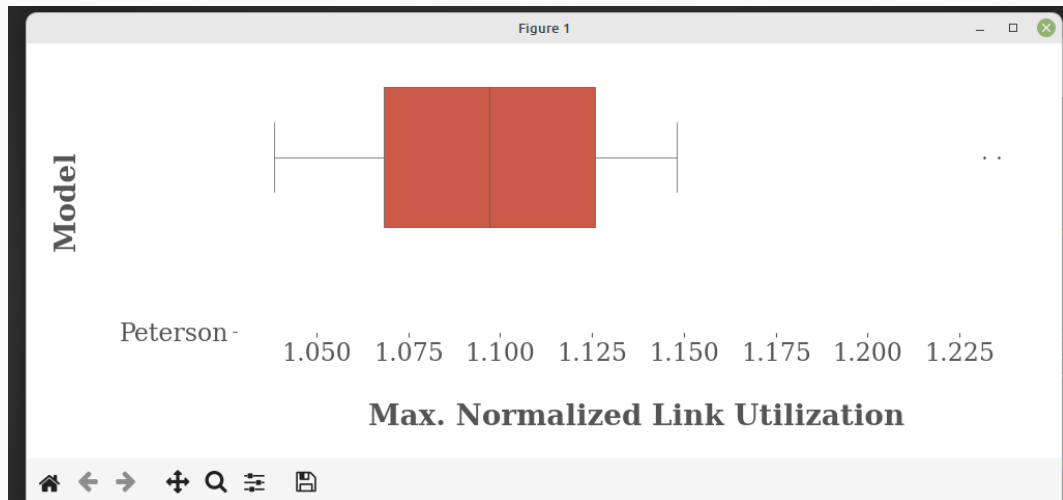
# Resultierender Boxplot





- keine Boxen:
  - vermutlich zu schwacher Rechner (2 Minuten Timeout reicht nicht)
- Sequential Combination besser:
  - meine Topologie wurde vereinfacht, von Grund auf (**.json**→**.topo.py**→**.topo.sh**)
  - Joint/Weights evtl nicht

# Ergebnisse Reproduktion

---

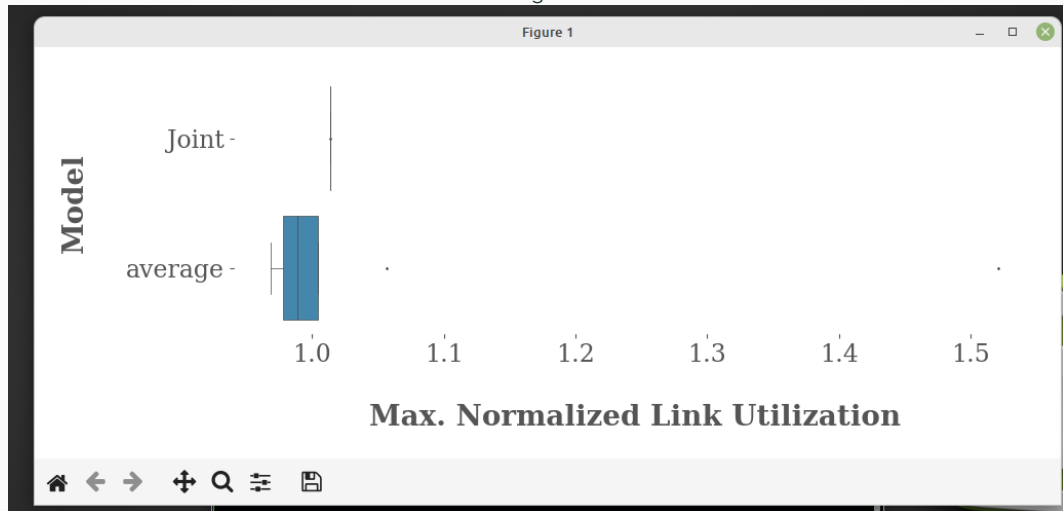


 *batch\_result001.csv* U Xrepli >  batch\_result001.csv

1	130 ; Peterson.topo.sh ; 1.3226666666666667e-06
2	131 ; Peterson.topo.sh ; 1.3226666666666667e-06
3	132 ; Peterson.topo.sh ; 1.3226666666666667e-06
4	

```
12.throughput.json 2.throughput.json averageweighted.topo.py flow_0-4.txt flow_13-4.txt  
13.throughput.json 3.throughput.json averageweighted.topo.sh flow_11-4.txt flow_1-3.txt  
(base) linuxmint@linuxmint203:~/FPRouting2/testbench$ sudo python3 gen_boxplot.py  
JOINT Median: 1.0137915706666667  
average Median: 0.988982292  
JOINT Minimum: 1.0137913413333333  
average Minimum: 0.9689171573333333  
JOINT Maximum: 1.0138813386666667  
average Maximum: 1.5211524506666667  
(base) linuxmint@linuxmint203:~/FPRouting2/testbench$
```

S





Fragen oder Anmerkungen?