

# ROUTINGALGORITHMEN

ZWISCHENPRÄSENTATION

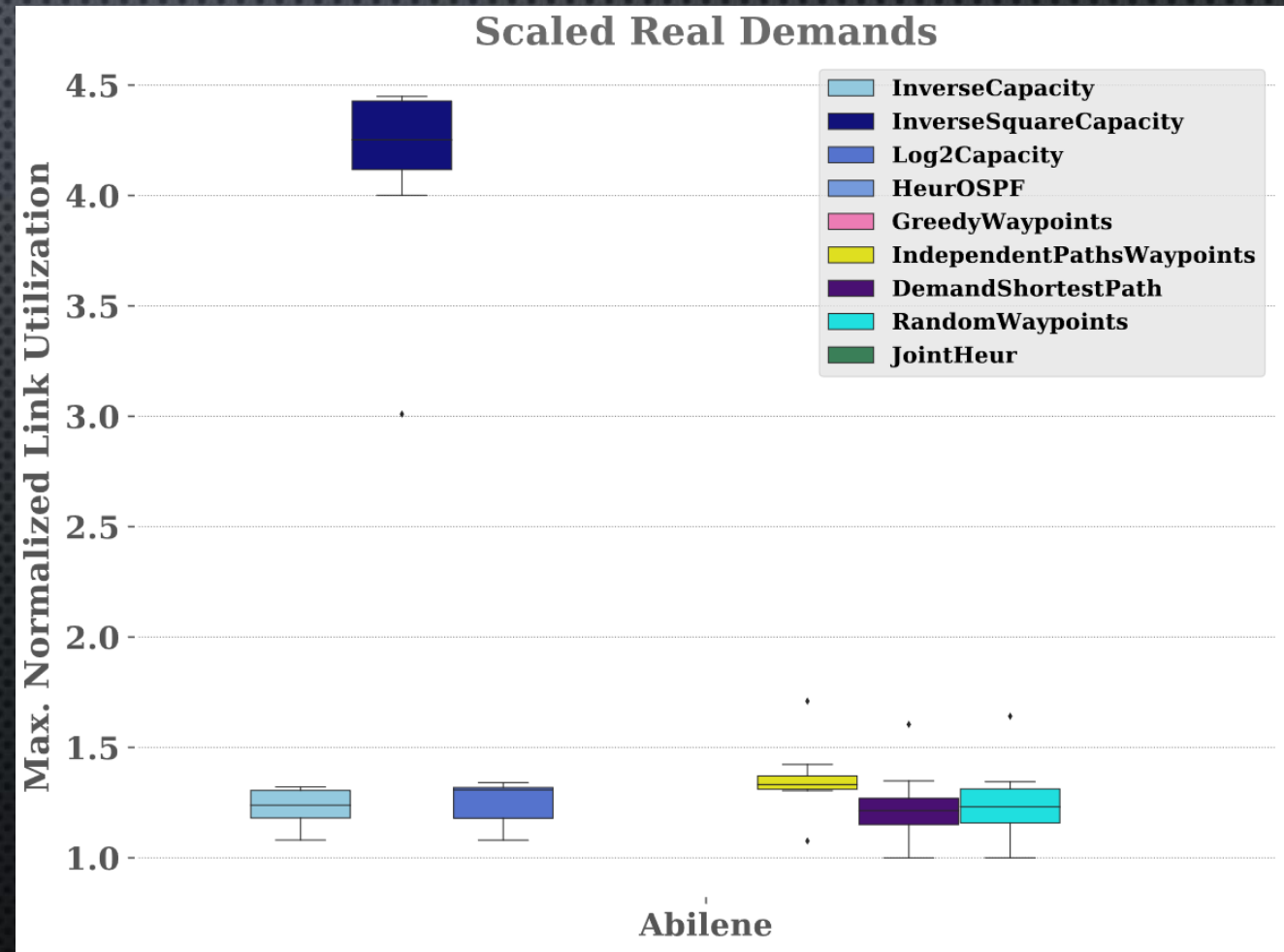
MARVIN WEILER

GEORGIOS KARAMOISSANLIS

SEBASTIAN PETERS

# VERWORFENE IDEEN

- ÄNDERUNG DER MATHEMATISCHEN OPERATION BEI BESTIMMUNG DER KAPAZITÄT
  - KEINE ÄNDERUNG DER LAUFZEIT
  - NICHT BESSER ALS INVERSE\_CAPACITY
- LOG2CAPACITY
- INVERSE\_SQUARE
- ...





# DEMAND\_SHORTEST\_PATH

- IDEE :
  - SHORTEST PATH FÜR EINEN DEMAND BESTIMMEN
  - MÖGLICHEN WEGPUNKTE LIEGEN ENTLANG DES SHORTEST PATHS

```
demand_shortest_path (g)
1  For (s,t) in demands
2    SPNL ← Dijkstra(g,s,t)
3    pw ← {}
4    For n in SPNL
5      For b in adj(n)
6      | pw ← pw ∪ {b}
demand_first_waypoint(pw)
```



# RANDOM\_WAYPOINTS

- IDEE : EINGRENZEN DER ANZAHL AN TESTFÄLLEN
  - DURCH RANDOMISIERTE AUSWAHL AN WEGPUNKTEN
  - AUS DEN ZUFÄLLIGEN WP DEN BESTEN BESTIMMEN
- CHANCE : VERBESSERUNG DURCH ERHÖHUNG VON MÖGLICHEN WEGPUNKTEN

Random\_Waypoints( $g$ )

$best \leftarrow None$

for all demands

$n \leftarrow nodes(g)$

$rw \leftarrow \text{choose 3 random points from } n$

    for  $w$  in  $rw$

$o \leftarrow test\_utilisation()$

        if  $o > best$

$best \leftarrow o$

return  $best$



# INDEPENDENT\_PATHS\_WAYPOINTS

- IDEE : JEDER DEMAND WIRD ENTLANG EINES KÜRZESTEN WEGES GELEITET
- BESTEN AUS N WEGEN TESTEN WENN ES MEHRERE WEGE ZUR AUSWAHL GIBT

```
independent_paths_waypoints(s)  
    best ← None  
    for (sit) in demands  
        i ← 0  
        for path in shortest_paths(s, sit)  
            if i > 5  
                break  
            i ← i + 1  
            if len(path) == 2  
                continue  
            update_flow_map(path)  
            o ← test_utilisation()  
            if o > best  
                best ← o  
    return best
```

# STATISTIKEN ÜBER UNSERE IDEEN

