Fachprojekt 1

Phil Seißelberg, Gajann Sivarajah, Jan Schulte

Table of Contents

- Foundation
- Algorithm 1 : kWPO-JointHeur
- Algorithm 2 : Topo_kWP-JointHeur
- Algorithm 3 : Nodes_kWP-JointHeur
- Conclusion
- Replication: Other Group

Foundation

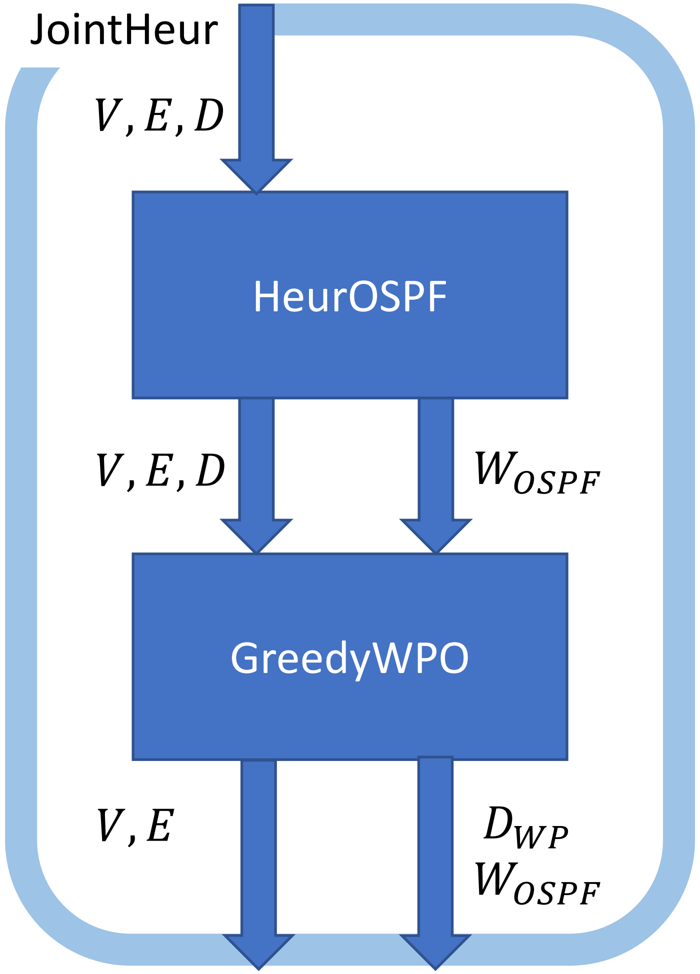
- Network Instance: $\aleph = \{V, E, W, D\}$
  - $V \coloneqq Nodes, E \coloneqq Edges(with\ Capacity)$, $W \coloneqq Weight\ Setting, D \coloneqq Demands$

- Base-Algorithm: JointHeur [1]
- Target : Minimize MLU

- Our Focus: Control amount of generated waypoints

Algorithm 1

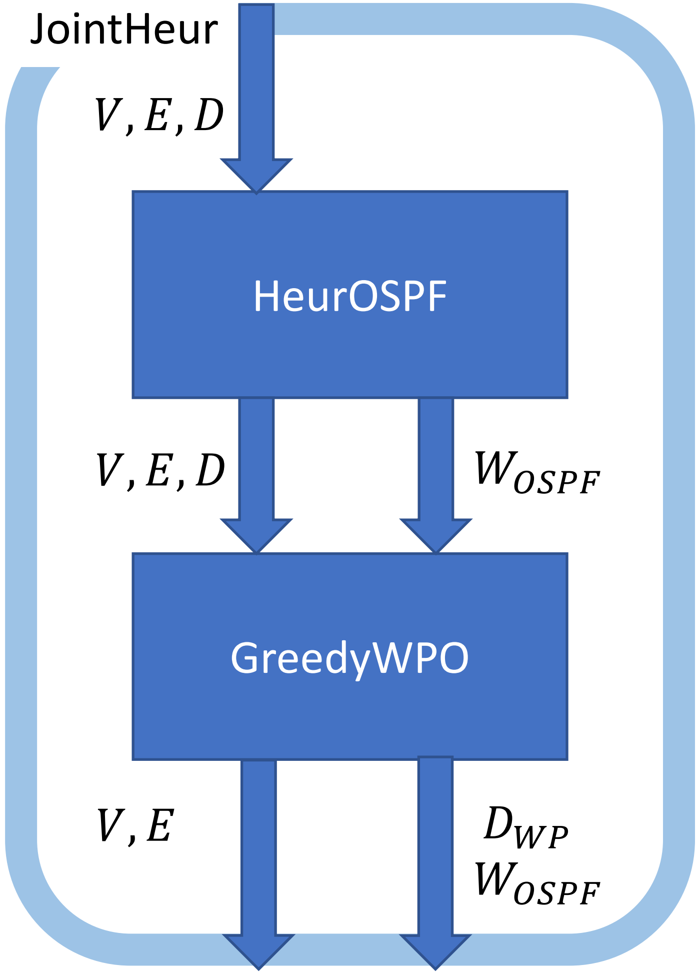
kWPO-JointHeur: Iterate GreedyWPO



[1]

Algorithm 1

kWPO-JointHeur: Iterate GreedyWPO



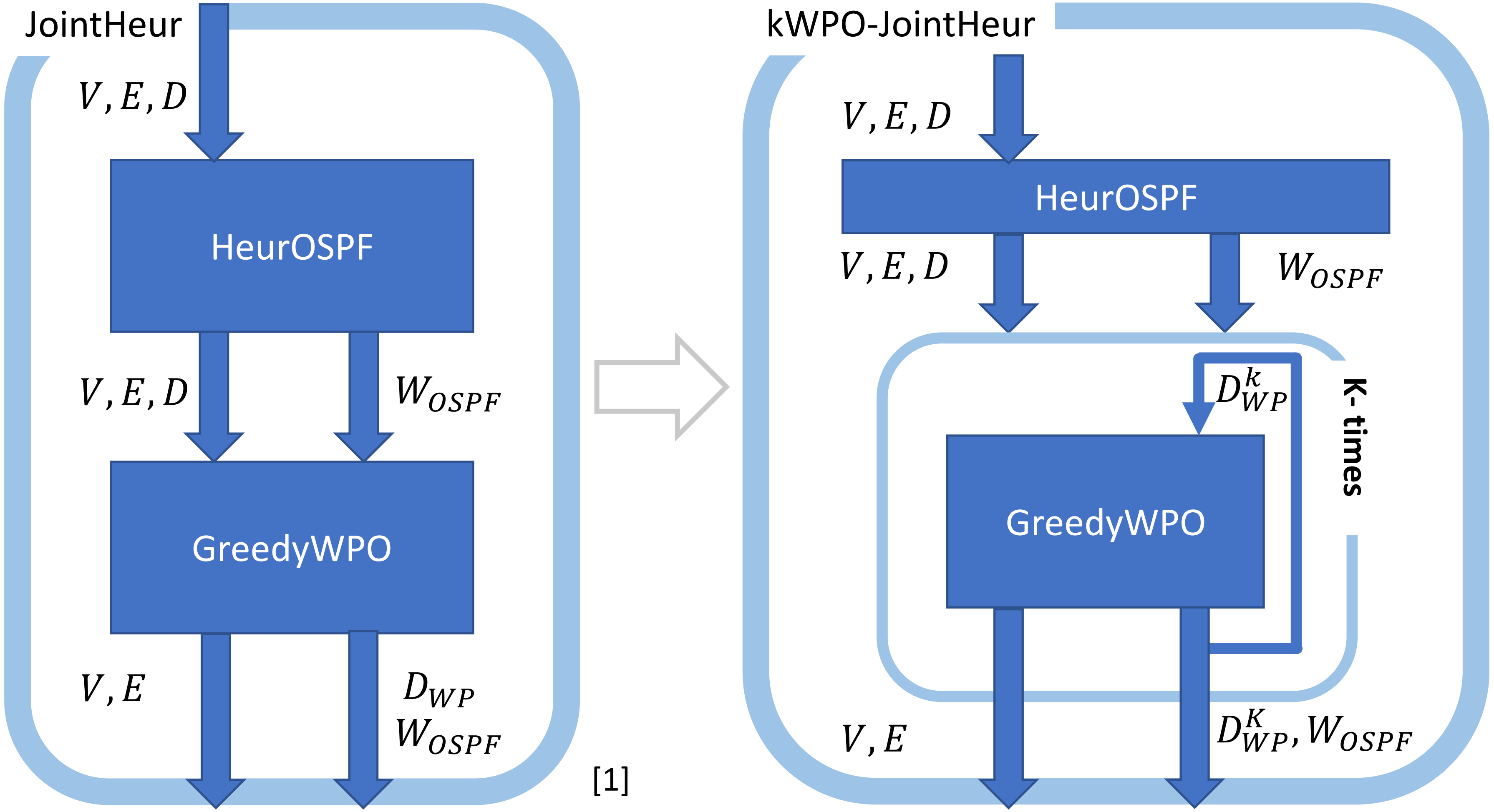
[1]

Motivation:

- Allow multiple waypoints per demand

Algorithm 1

kWPO-JointHeur: Iterate GreedyWPO



[1]

$K = 0$
$\rightarrow HeurOSPF$

$K = 1$
$\rightarrow JointHeur$

$K > 1$
$\rightarrow max.\ 2^{K} - 1\ WP$
$per\ demand$

Algorithm 1

kWPO-JointHeur: Sort by Capacity

GreedyWPO(V,E,D,W):

```
$U_{min} \leftarrow MLU(V,E,D,W)$
$\boldsymbol{D_{WP} \leftarrow D\ sorted\ by\ demand\ value}$
$for\ demand\ \psi = (s,t,d) \in D_{WP}\ do$
    $wp_{\psi} \leftarrow None$
    $for\ node\ w \in V\ do$
        $D' \leftarrow D_{WP} \setminus \psi \cup \{(s,w,d),(w,t,d)\}$
        $U \leftarrow MLU(V,E,D',W)$
        $if\ U < U_{min}$ then
            $wp_{\psi} \leftarrow w$
            $U_{min} \leftarrow U$
$for\ each\ wp_{\psi} \neq None\ do$
    $D_{WP} \leftarrow D_{WP} \setminus \psi \cup \{(s,wp_{\psi},d),(wp_{\psi},t,d)\}$
$return\ D_{WP}$
```

[2]

Motivation:

- Order of demands influences performance
- Use topology-dependent criteria

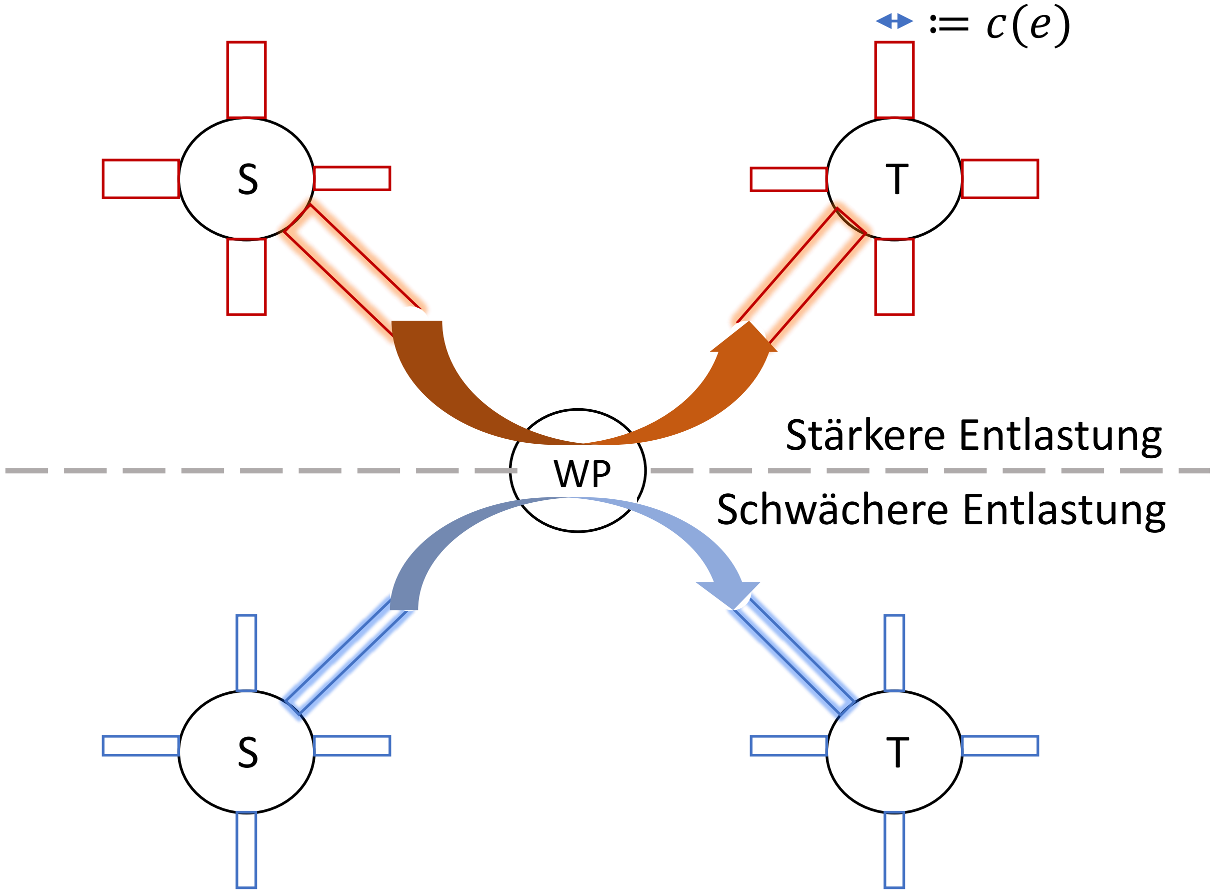
Algorithm 1

kWPO-JointHeur: Sort by Capacity

```
GreedyWPO(V,E,D,W):

$U_{min} \leftarrow MLU(V,E,D,W)$
$\boldsymbol{D_{WP} \leftarrow D\ sorted\ by\ demand\ value}$
$for\ demand\ \psi = (s,t,d) \in D_{WP}\ do$
    $wp_{\psi} \leftarrow None$
    $for\ node\ w \in V\ do$
        $D' \leftarrow D_{WP} \backslash \psi \cup \{(s,w,d),(w,t,d)\}$
        $U \leftarrow MLU(V,E,D',W)$
        $if\ U < U_{min}$ then
            $wp_{\psi} \leftarrow w$
            $U_{min} \leftarrow U$
$for\ each\ wp_{\psi} \neq None\ do$
    $D_{WP} \leftarrow D_{WP} \backslash \psi \cup \{(s,wp_{\psi},d),(wp_{\psi},t,d)\}$
$return\ D_{WP}$
```

[2]



kWPO-JointHeur: Sort by Capacity

```
GreedyWPO(V,E,D,W):

$U_{min} \leftarrow MLU(V,E,D,W)$
$\boldsymbol{D_{WP} \leftarrow D\ sorted\ by\ demand\ value}$
$for\ demand\ \psi = (s,t,d) \in D_{WP}\ do$
    $wp_{\psi} \leftarrow None$
    $for\ node\ w \in V\ do$
        $D' \leftarrow D_{WP} \setminus \psi \cup \{(s,w,d),(w,t,d)\}$
        $U \leftarrow MLU(V,E,D',W)$
        $if\ U < U_{min}$ then
            $wp_{\psi} \leftarrow w$
            $U_{min} \leftarrow U$
$for\ each\ wp_{\psi} \neq None\ do$
    $D_{WP} \leftarrow D_{WP} \setminus \psi \cup \{(s,wp_{\psi},d),(wp_{\psi},t,d)\}$
$return\ D_{WP}$
```

[2]

$\boldsymbol{D_{WP} \leftarrow D\ sorted\ by\ C(\psi) = (C_s + C_t)}$

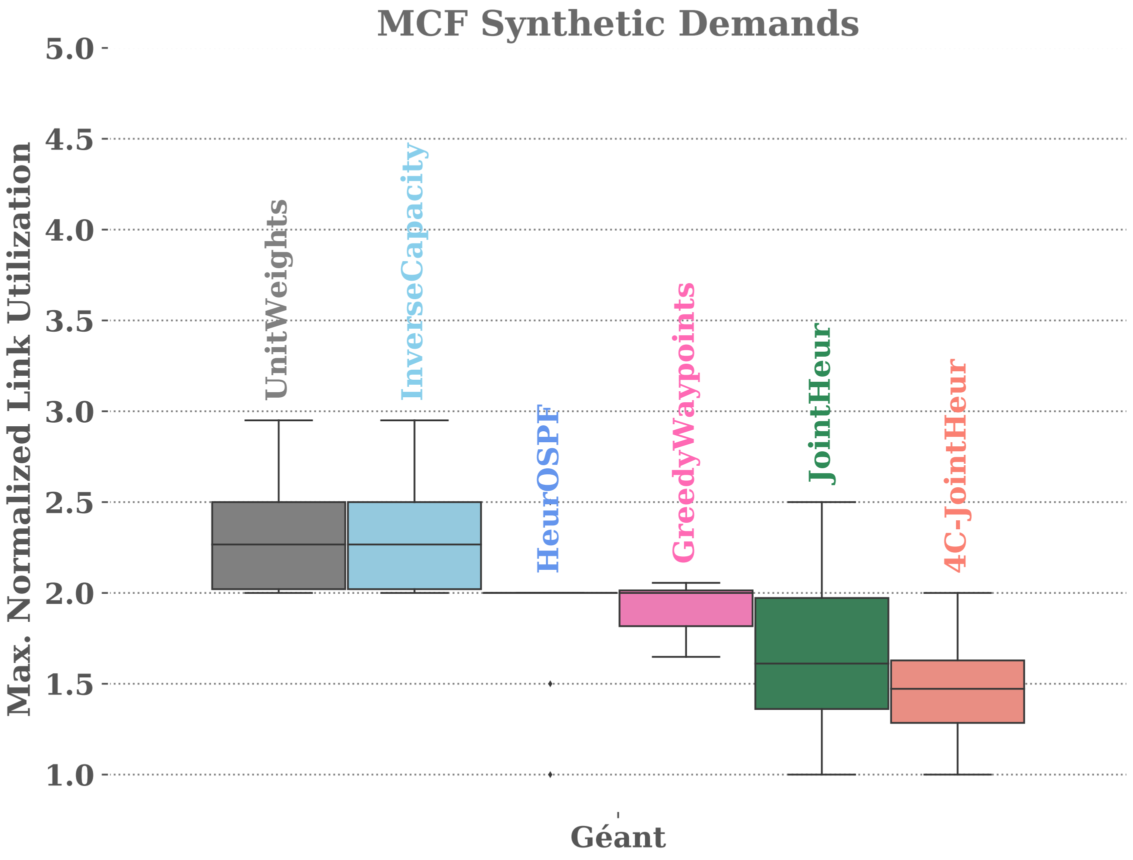
Node-Capacity C_v:

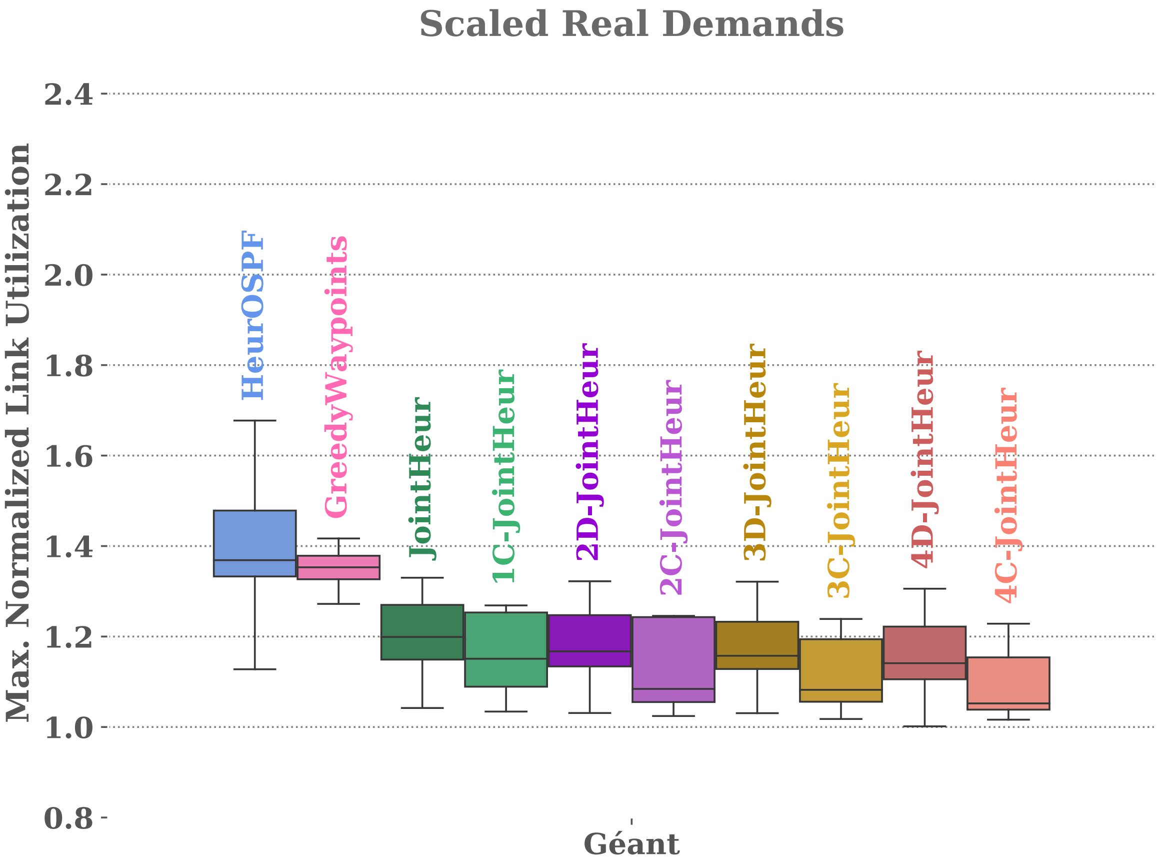
$$C_v = \sum_{e_{vi} \in E} c(e_{vi})$$

$$\coloneqq sum\ of\ connected\ edge - capacities$$

Algorithm 1

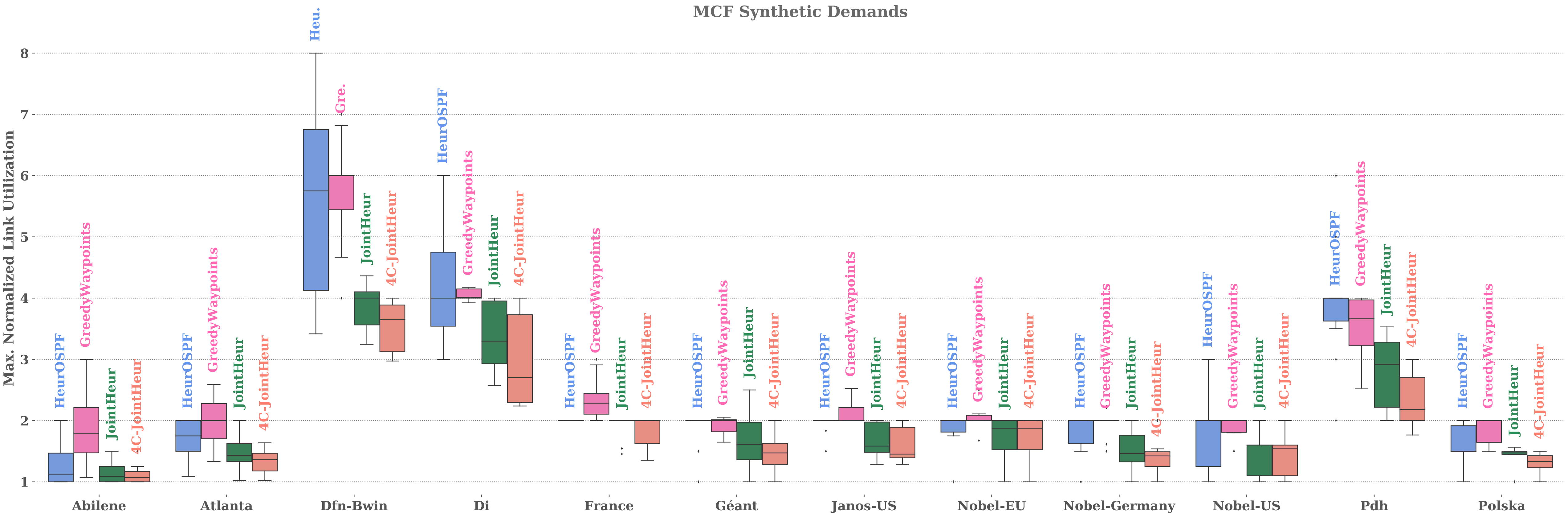
kWPO-JointHeur: Results





Algorithm 1

kWPO-JointHeur: Results



kWP per Topology: Concept

Idea: Limit the number of usable waypoints throughout the complete run of the algorithm for a topology to $k \in \mathbb{N}$

Actions:

- Implement parameter k as a counter in the algorithms
- Check out results for different values for k and different topologies

Why could this be interesting? - Answers for the questions:

- What number of waypoints are useful for such a restriction?
- What influence do waypoints actually have on the performance?

Algorithm 2

kWP per Topology: Prototype

```
GreedyWPO(V,E,D,W,K):
$k \leftarrow K$
$U_{min} \leftarrow MLU(V, E, D, W)$
$D_{WP} \leftarrow D$ sorted by demand value
for demand $\psi = (s, t, d) \in D_{WP}$ do
    $wp_{\psi} \leftarrow None$
    $\boldsymbol{if\ k \leq 0\ then\ break}$
    for node $w \in V$ do
        $D' \leftarrow D_{WP} \setminus \psi \cup \{(s, w, d), (w, t, d)\}$
        $U \leftarrow MLU(V, E, D', W)$
        if $U < U_{min}$ then
            $wp_{\psi} \leftarrow w$
            $U_{min} \leftarrow U$
    $\boldsymbol{if\ wp_{\psi} \neq None\ then\ k \leftarrow k - 1}$
for each $wp_{\psi} \neq None$ do
    $D_{WP} \leftarrow D_{WP} \setminus \psi \cup \{(s, wp_{\psi}, d), (wp_{\psi}, t, d)\}$
return $D_{WP}$
```

[2]

kWP per Topology: Results

Impressions:

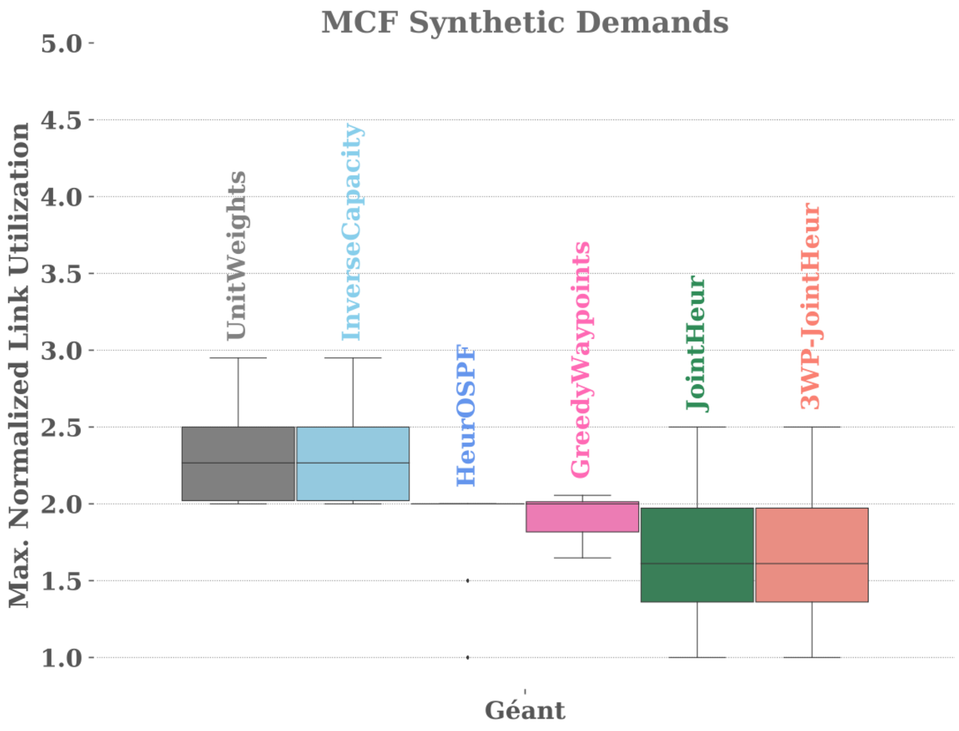
- The Limitation of the usable Waypoints cannot improve MLU
    - $MLU_k \geq MLU$

- The impact of WP-limit depends on the topology

- (Scaled-)Real demands show greater reaction to WP-restraints in JointHeur

Algorithm 2

kWP per Topology: Results

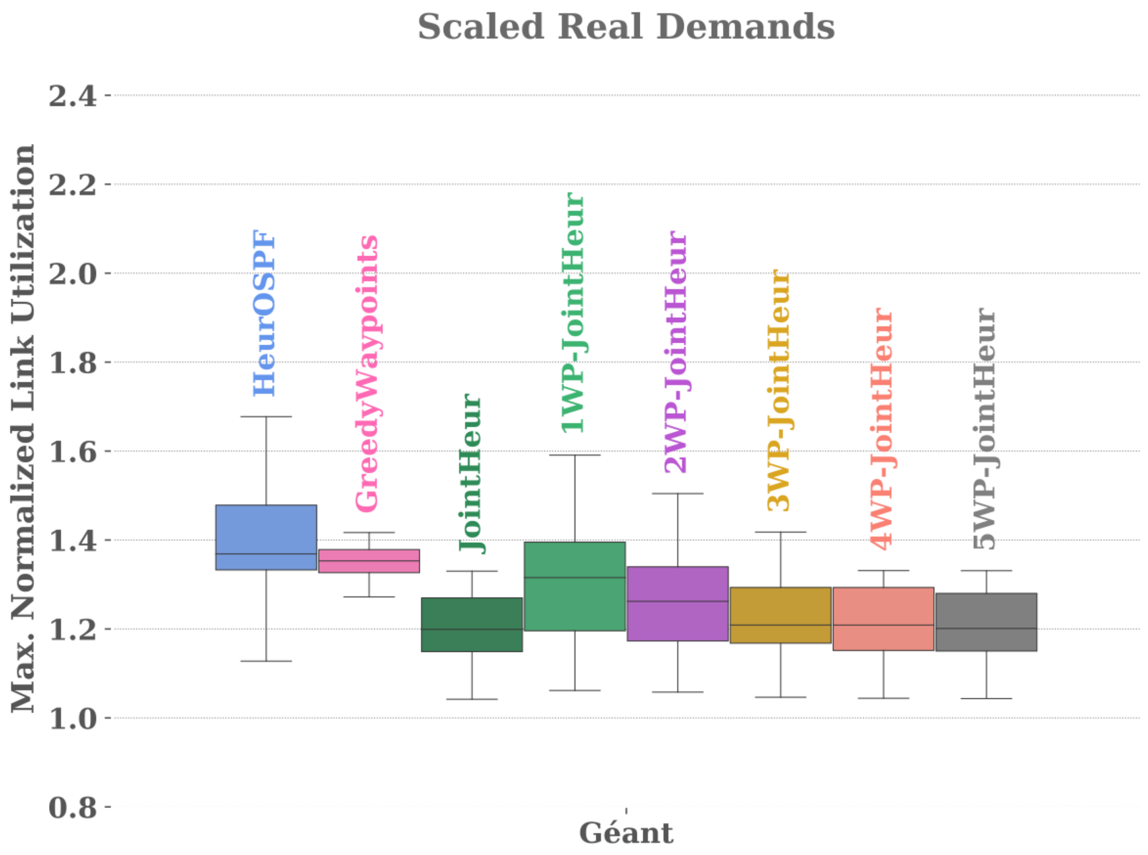


Synthetic Demands :

➡ Limitation seems to not affect MLU

Algorithm 2

kWP per Topology: Results

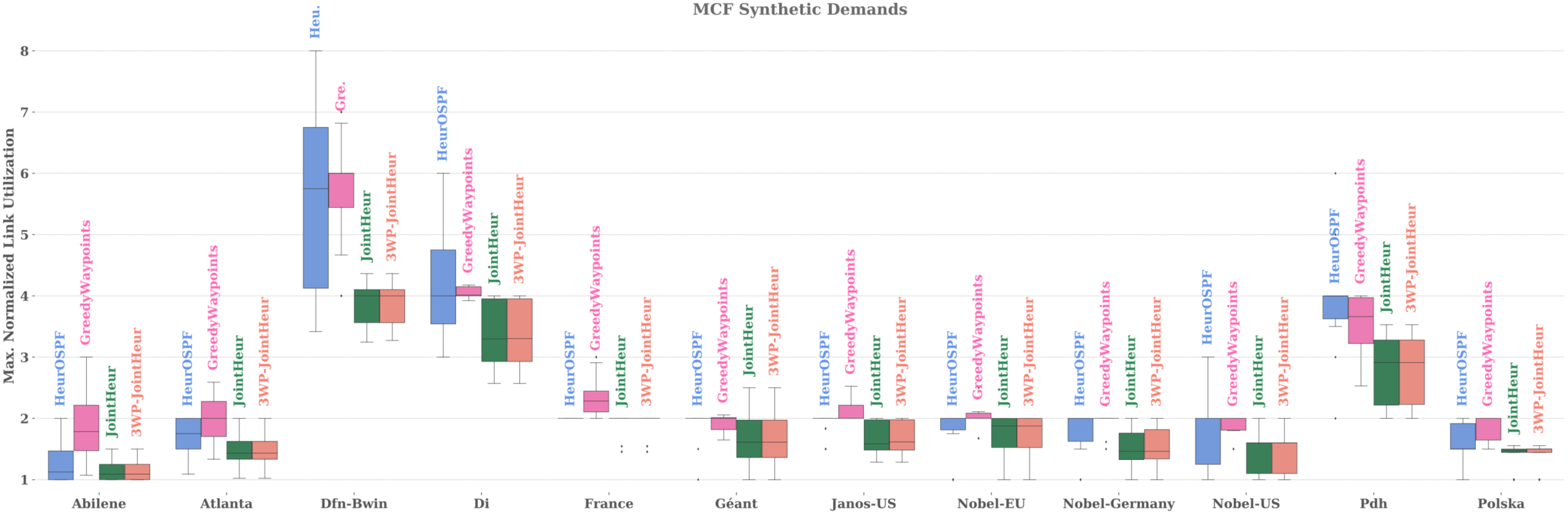


Real Demands :

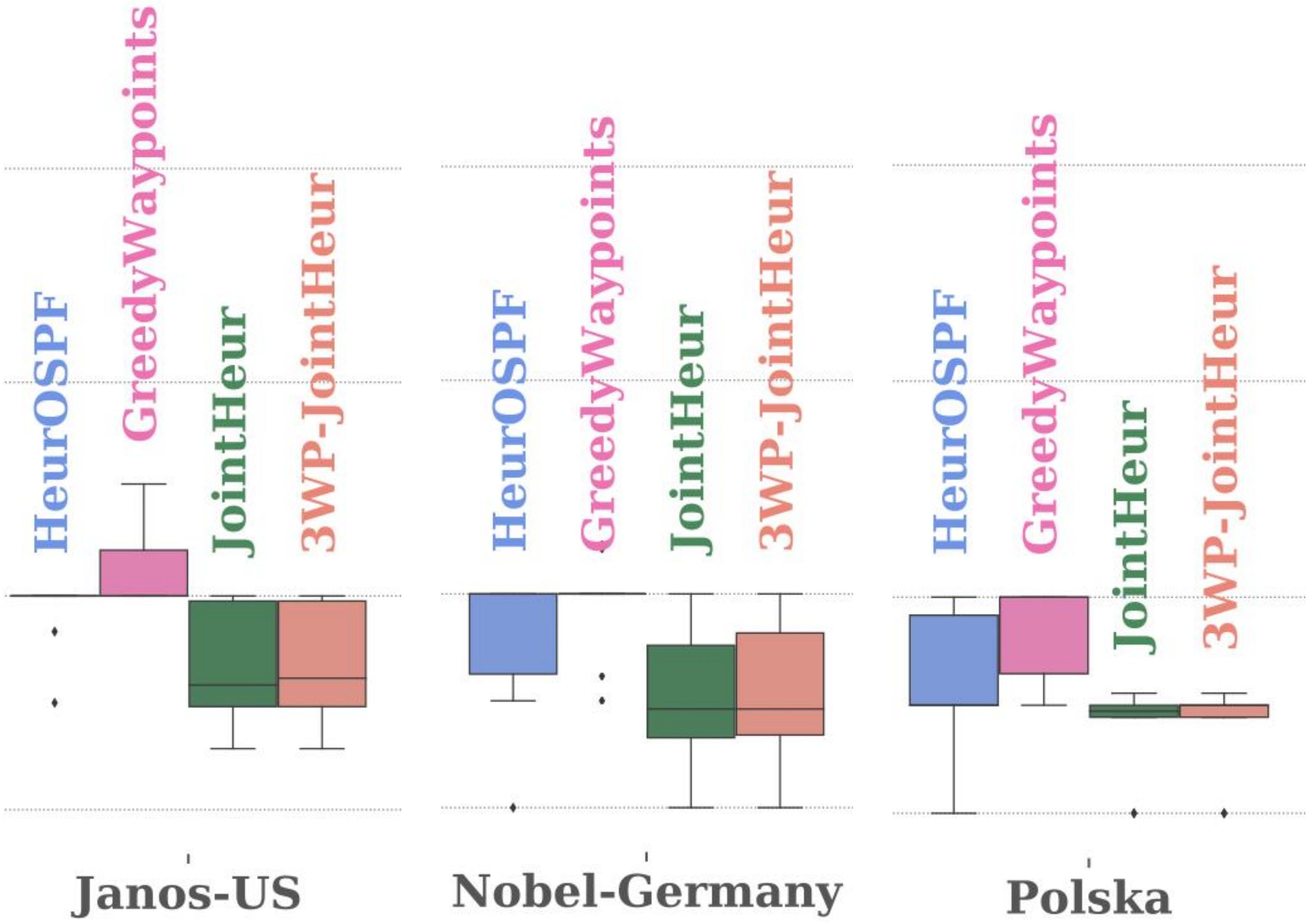
- MLU improves with each waypoint
- JointHeur shows measurable reaction to limit

Algorithm 2

kWP per Topology: Results



kWP per Topology: Results



Interpretation:

- ➡ most Topologies create $\leq 3\ WP$
- ➡ others are clearly missing useful WPs

kWP per Node: Concept

Idea: Each Node can only be used as a waypoint a finite number of times. Counter k for each Node can be defined homogene or heterogene

Strategy:

- Create $K = \{k_v \in \mathrm{N} | v \in V\ and k_v \coloneqq Counter\ of\ node\ v\}$
- Check out results for different distributions for K and different topologies

Why could this be interesting? - Answers for the questions:

- What influence do certain distributions of K have on performance?
- Can you enforce a certain distribution of waypoints using K?
    - Ban nodes based on reliability or security $(k = 0)$

Algorithm 3

kWP per Node: Practical

- Issue: How to define K for multiple Topologies in our experiment?

- Solution: Generate K using common rules for all Topologies

➡ Choose key-value $c_{gen} \in \mathbb{N}$

➡ Let $K = \{k_v \in \mathbb{N} \mid k_v = \begin{cases} \infty \;, \; if \;\; c_{gen} | v \\ 0 \;\;, \; else \end{cases} \}$

↔ Only every c_{gen}-th node is allowed as a waypoint

- Example: $c_{gen} = 2 \rightarrow$ 50% of the nodes are banned

Algorithm 3

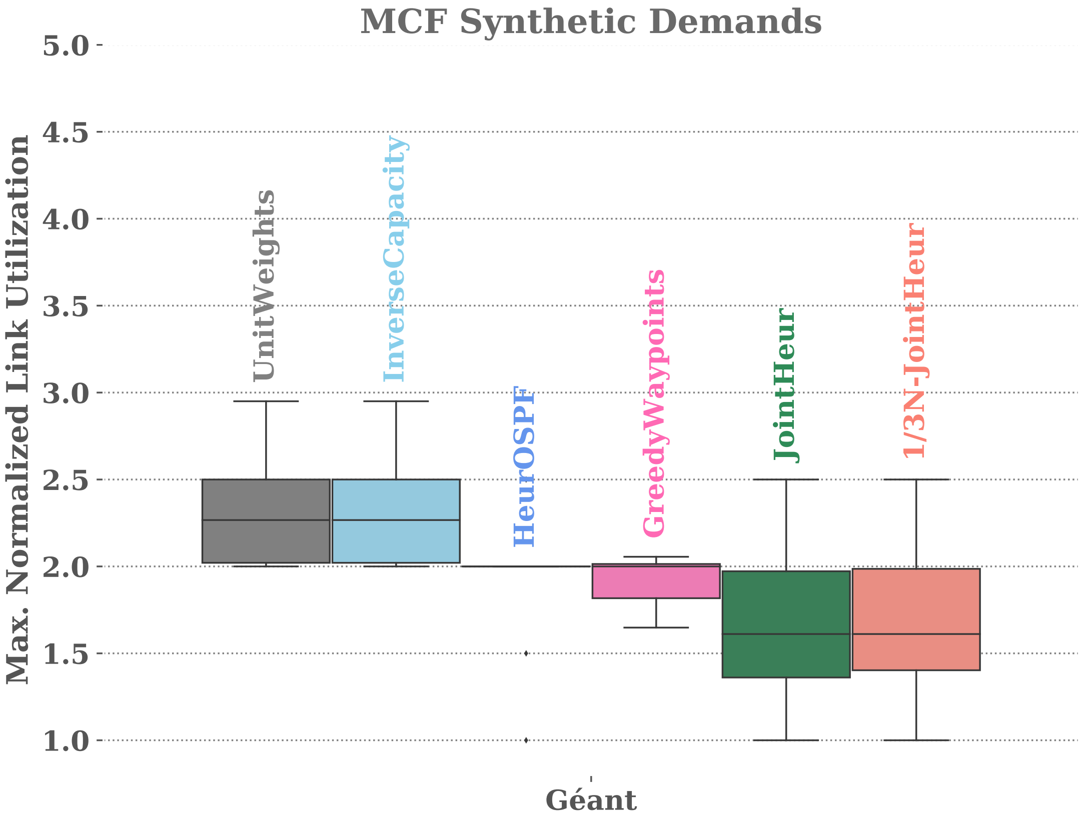
kWP per Node: Prototype

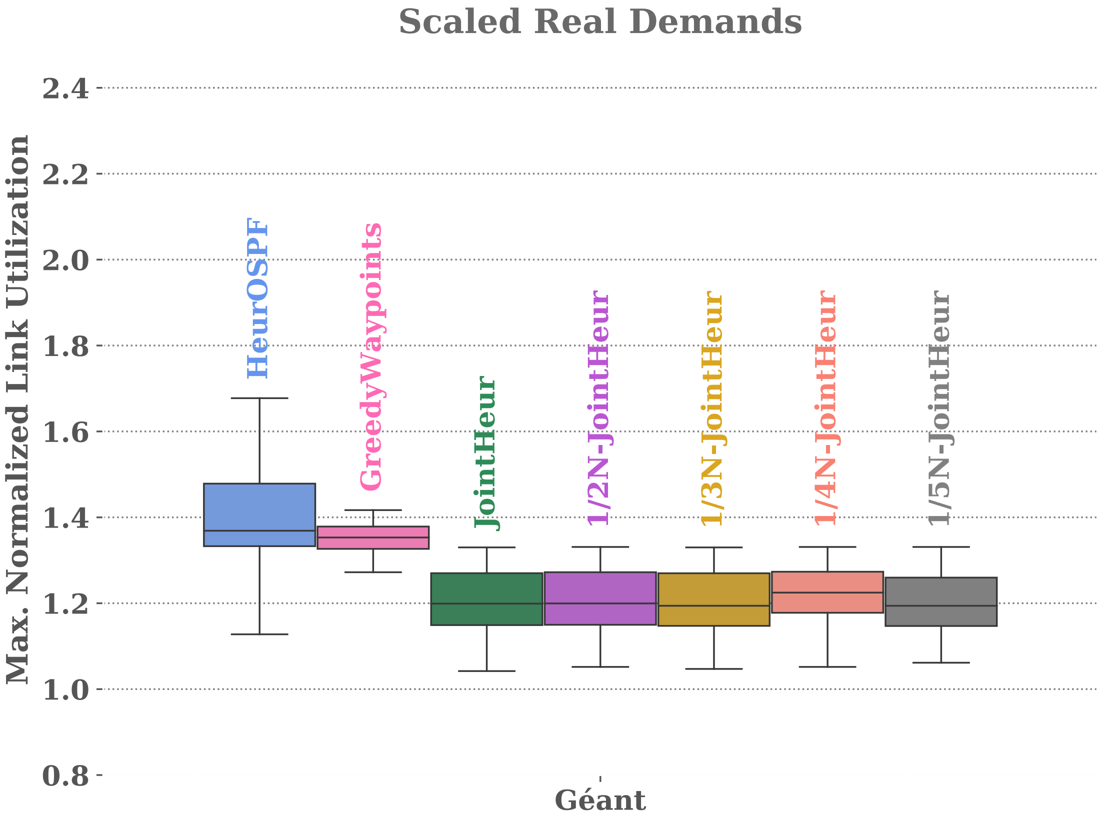
```
GreedyWPO(V,E,D,W,K):
$U_{min} \leftarrow MLU(V,E,D,W)$
$D_{WP} \leftarrow D\ sorted\ by\ demand\ value$
$for\ demand\ \psi = (s,t,d) \in D_{WP}\ do$
    $wp_{\psi} \leftarrow None$
    $for\ node\ w \in V\ do$
        $\boldsymbol{if\ K[w] \leq 0\ then\ continue}$
        $D' \leftarrow D_{WP} \backslash \psi \cup \{(s,w,d),(w,t,d)\}$
        $U \leftarrow MLU(V,E,D',W)$
        $if\ U < U_{min}$ then
            $wp_{\psi} \leftarrow w$
            $U_{min} \leftarrow U$
    $\boldsymbol{if\ wp_{\psi} \neq None\ then\ K[wp_{\psi}] \leftarrow K[wp_{\psi}] - 1}$
$for\ each\ wp_{\psi} \neq None\ do$
    $D_{WP} \leftarrow D_{WP} \backslash \psi \cup \{(s,wp_{\psi},d),(wp_{\psi},t,d)\}$
$return\ D_{WP}$
```

[2]

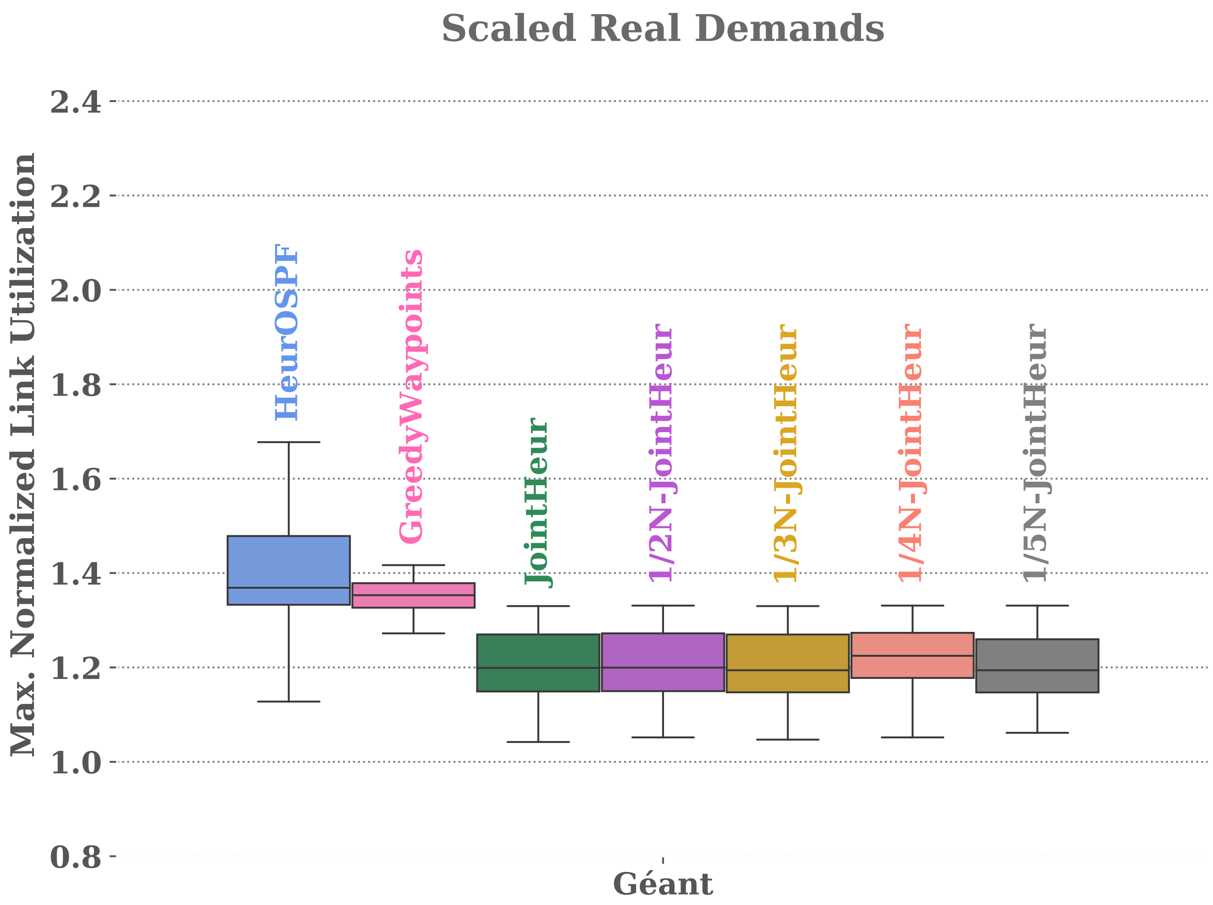
Algorithm 3

kWP per Node: Results





kWP per Node: Results

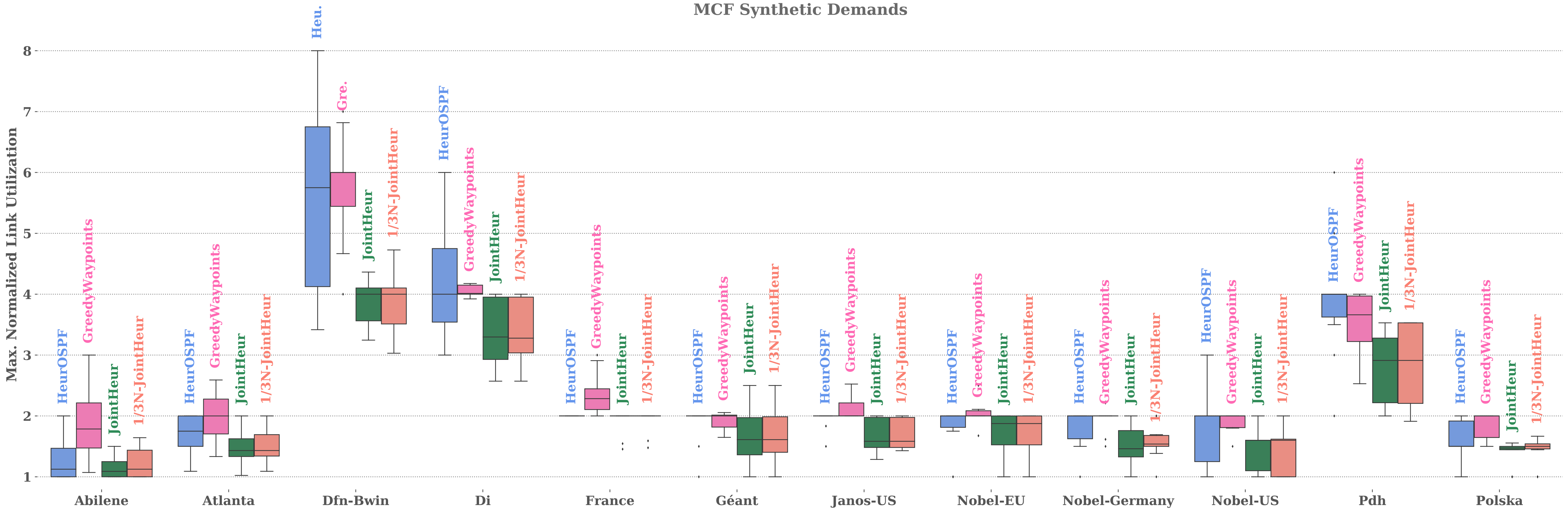


Interpretation:

- c_{gen} defines a subset of V as a WP-Pool
  - $\uparrow c_{gen} \Longrightarrow \downarrow |V_{WP-Pool}|$
- A smaller WP-Pool isn‘t necessarily worse, because the elements are different
  - $\uparrow c_{gen} \not\Rightarrow \uparrow MLU$

Algorithm 3

kWP per Node: Results



Conclusion

- Topology‘s susceptibility towards waypoints determines whether our constraints have a measurable influence
  - Some topologies didn‘t react to our algorithms at all

- Future Idea
  - Combine all algorithms and constraints

Future Idea - Combination

ALG1: kWPO

- Benefit: Eliminate upper limit of possibly generated waypoints
- Problem: Number of waypoints per iteration not predictable → How many iterations?

+

ALG2: Topo-kWP

- Solution: Keep iteration high, but cap actually generated waypoints
- Problem: Risk of cutting off „desirable“ waypoints

+

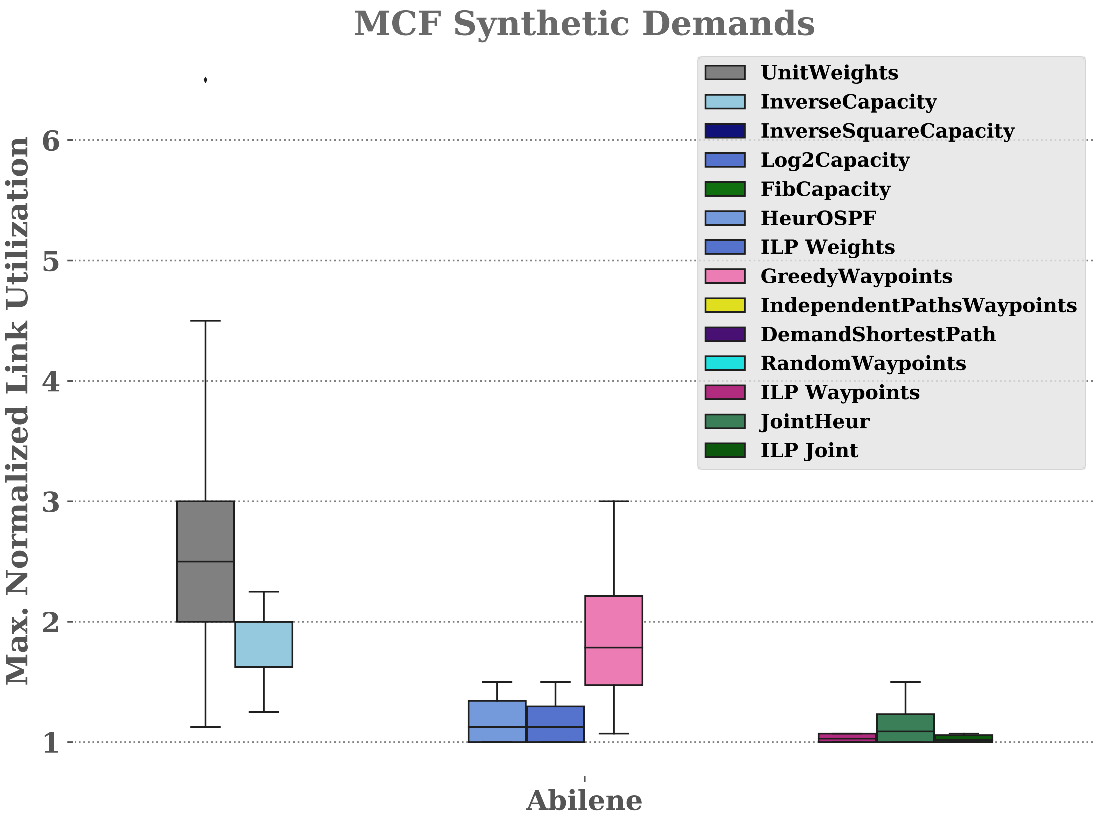
ALG3: Nodes-kWP

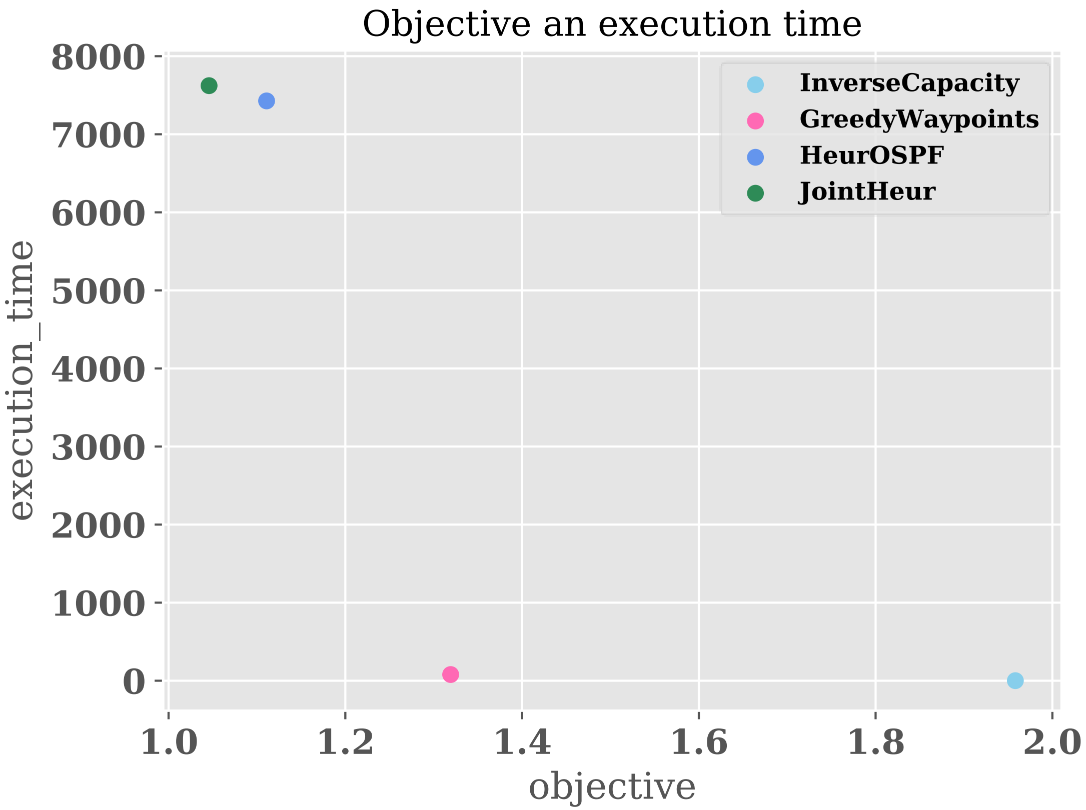
- Solution: Etch prioritisation into waypoint generation
- Problem: Which criterias to consider when choosing a distribution?

Replication – Code

- Distribution
  - Public Repository on Github
  - Additional preconfigured VM-Image
- Execution
  - No errors
  - high(to be expected) runtime
- Results
  - Additional algorithms were added to existing plots
  - Plots illustrating processing time

Replication - Plots


MCF Synthetic Demands


Objective an execution time

Sources

- [1]: Thomas Fenz, Klaus-Tycho Förster, Mahmoud Parham, Stefan Schmid, Nikolaus Süß:

  Traffic Engineering with Joint Link Weight and Segment Opitmization: Algorithm 2
- [2]: Thomas Fenz, Klaus-Tycho Förster, Mahmoud Parham, Stefan Schmid, Nikolaus Süß:

  Traffic Engineering with Joint Link Weight and Segment Opitmization: Algorithm 3