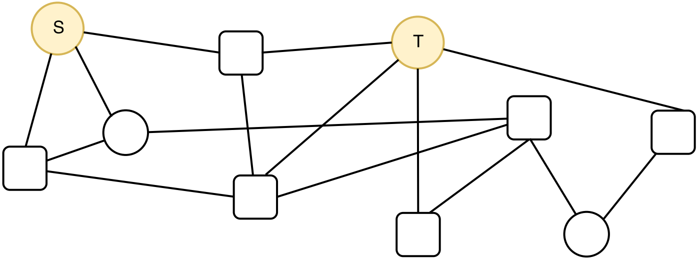
# Walking through the Cloud: Routing of Virtualized Network Functions

Klaus-Tycho Foerster Mahmoud Parham Stefan Schmid

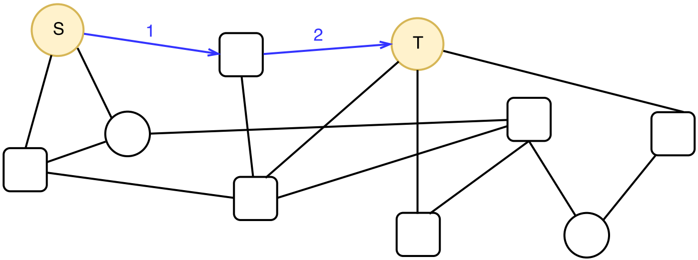
Department of Computer Science
Aalborg University, Denmark

ALGOCLOUD, 2017

# A Routing Problem



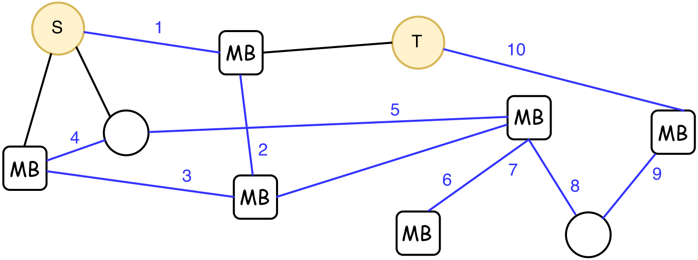
# A Routing Problem
## the usual routing



# A Routing Problem

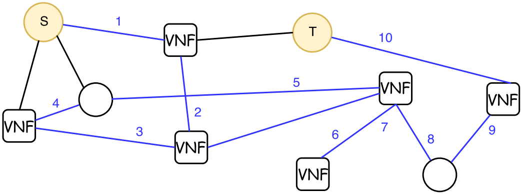
new: middleboxes

- Middlebox: Firewall, NAT, proxies, DPI etc.



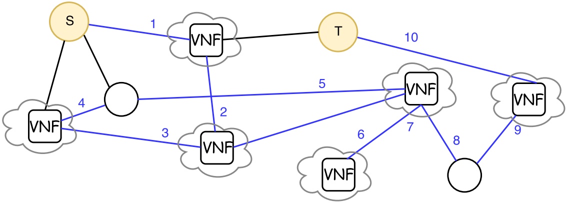
# A Routing Problem
## moving to clouds

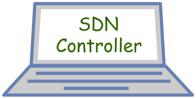
- VNFs brings flexibility, are cheaper



# A Routing Problem
## moving to clouds

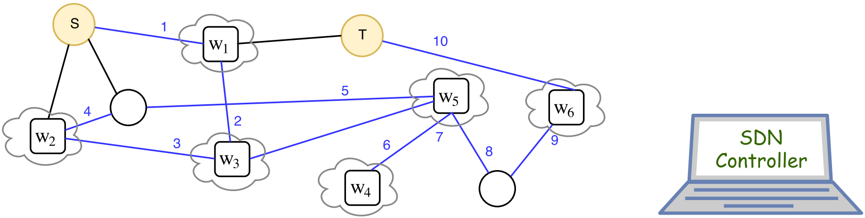
- VNFs brings flexibility, are cheaper
- A lot of them, in clouds



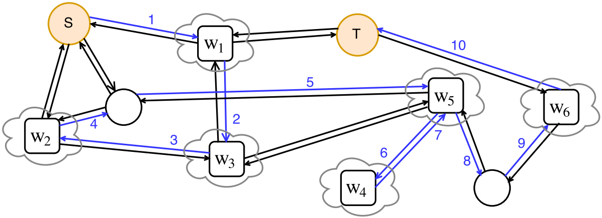
# A Routing Problem
## the waypoints

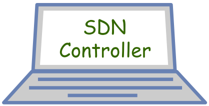
- The task: find the shortest $S$–$T$ walk through *waypoints*
- Capacities must be respected



# A Routing Problem
## the network

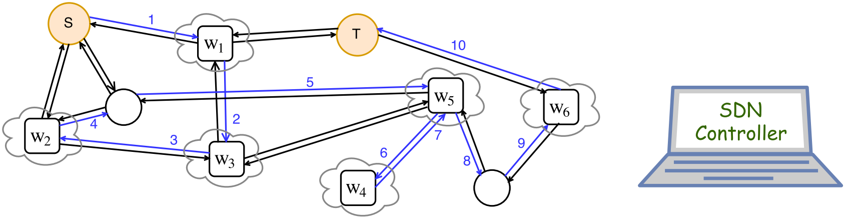
- Real Networks Are Bidirected



# A Routing Problem
## in two flavors

- Real Networks Are Bidirected
- Two Flavors: Ordered vs Unordered



# Outline

# Model

- Bidirected graph $G(V, E)$: $\forall (x, y) \in E \implies (y, x) \in E$
- $n$ nodes, $k$ of which are waypoints
- Arbitrary capacities, unit demand for $(S, T)$

# Model

- Bidirected graph $G(V, E)$: $\forall (x, y) \in E \implies (y, x) \in E$
- $n$ nodes, $k$ of which are waypoints
- Arbitrary capacities, unit demand for $(S, T)$
- The shortest *feasible* $S$-$T$ walk visiting *all* waypoints
- A feasible walk respects link capacities

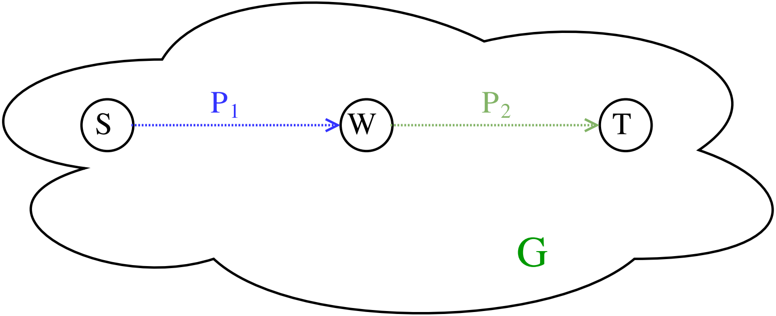
# Model

- Bidirected graph $G(V, E)$: $\forall (x, y) \in E \implies (y, x) \in E$
- $n$ nodes, $k$ of which are waypoints
- Arbitrary capacities, unit demand for $(S, T)$
- The shortest *feasible* $S$-$T$ walk visiting *all* waypoints
- A feasible walk respects link capacities
- Ordered and Unordered

# Outline

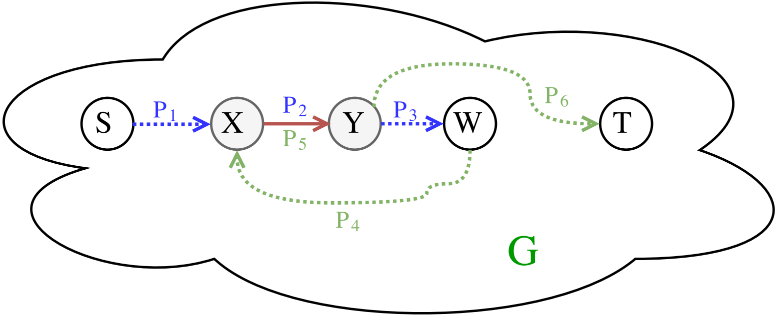
# One waypoint: greedy is optimal

Two shortest paths?



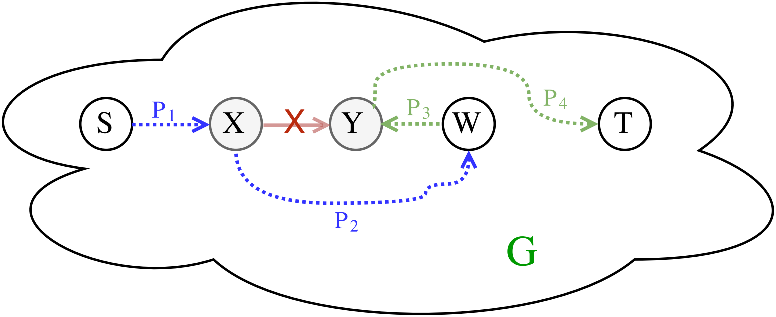
# One waypoint: greedy is optimal

Assume both shortest paths chose $(X, Y)$



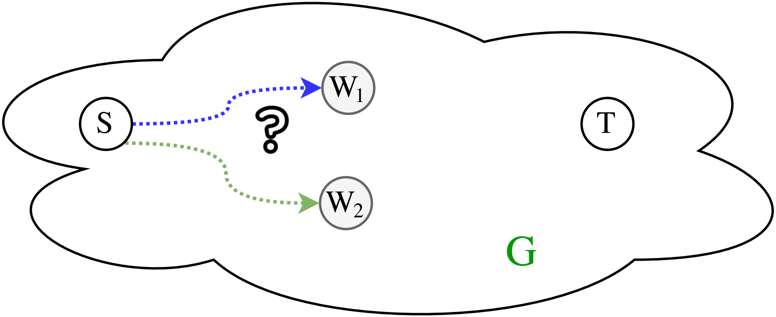
# One waypoint: greedy is optimal

$P_1P_2P_3P_4$ is shorter $\implies$ Contradiction!



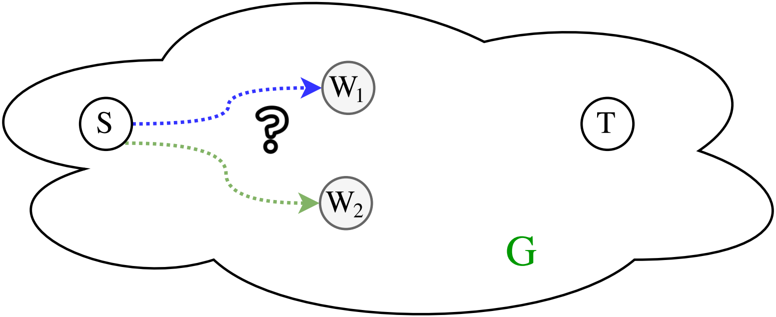
# More waypoints

✓ The optimal order + shortest paths $\Longrightarrow$ it works!



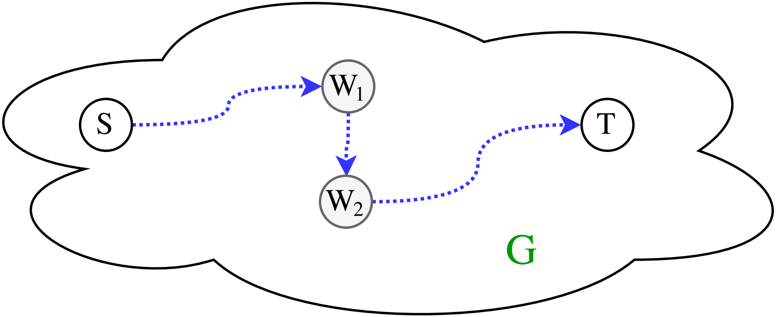
# More waypoints

- ✓ The optimal order + shortest paths $\implies$ it works!
- ✓ Try all permutations: $SW_1W_2T$ or $SW_2W_1T$?



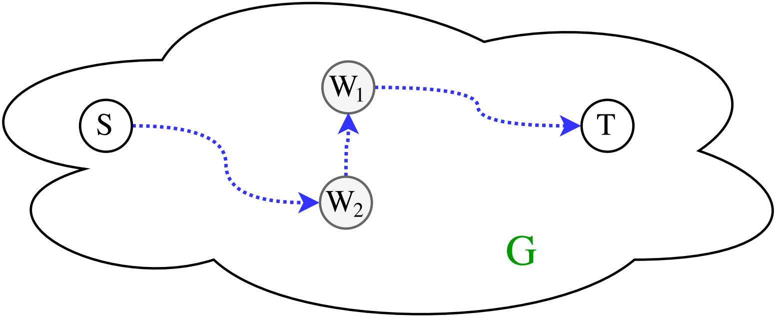
# More waypoints

- ✓ The optimal order + shortest paths $\implies$ it works!
- ✓ Try all permutations: $SW_1W_2T$ or $SW_2W_1T$?



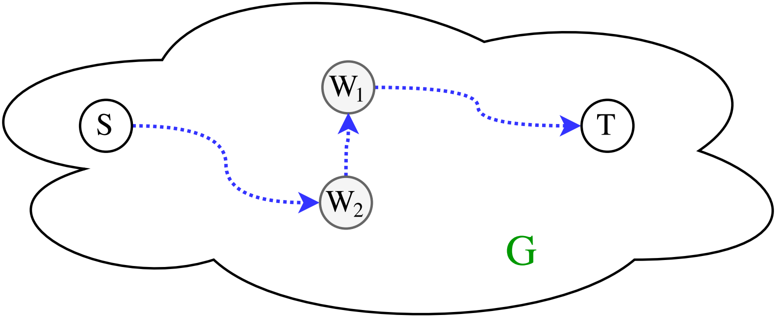
# More waypoints

- ✓ The optimal order + shortest paths $\implies$ it works!
- ✓ Try all permutations: $SW_1W_2T$ or $SW_2W_1T$?



# More waypoints

- ✓ The optimal order + shortest paths $\implies$ it works!
- ✓ Try all permutations: $SW_1W_2T$ or $SW_2W_1T$?
- ✓ Polynomial time for $k = \mathcal{O}\left(\frac{\log n}{\log \log n}\right)$



# Outline

# Hardness

- Feasibility via spanning tree $\implies$ always feasible

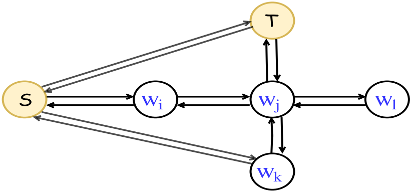



Figure: Bidirected

# Hardness

- Feasibility via spanning tree $\Longrightarrow$ always feasible

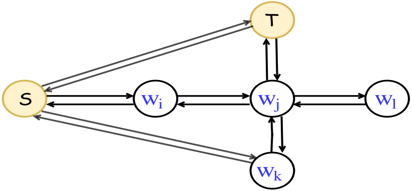



Figure: Bidirected

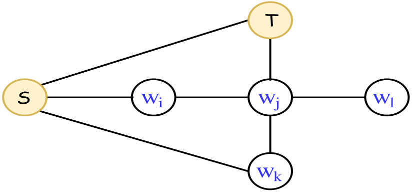



Figure: Undirected

# Hardness

- Feasibility via spanning tree $\implies$ always feasible

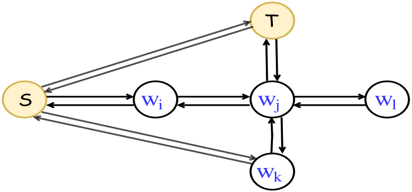



Figure: Bidirected

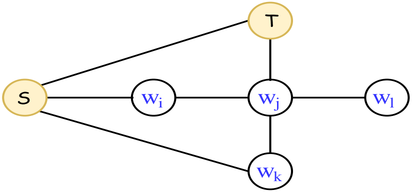



Figure: Undirected

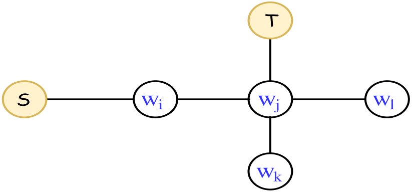



Figure: Spanning Tree

# Hardness

- Feasibility via spanning tree $\Longrightarrow$ always feasible

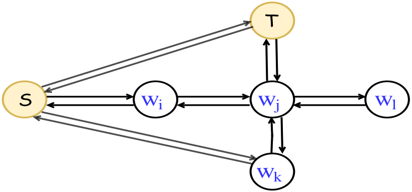



Figure: Bidirected

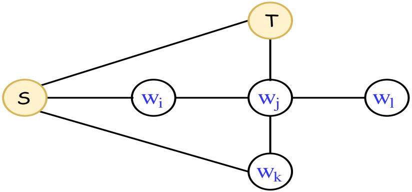



Figure: Undirected

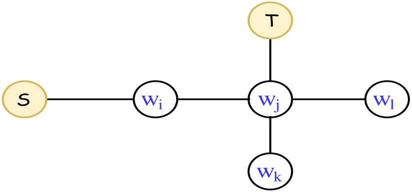



Figure: Spanning Tree

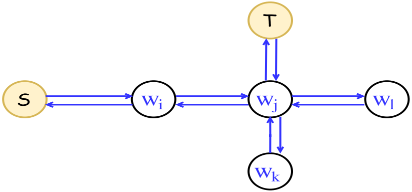



Figure: Bidirected again

# Hardness

- Feasibility via spanning tree $\implies$ always feasible

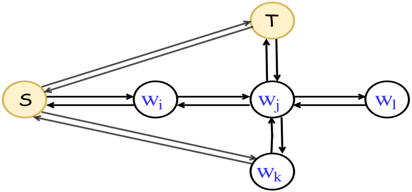



Figure: Bidirected

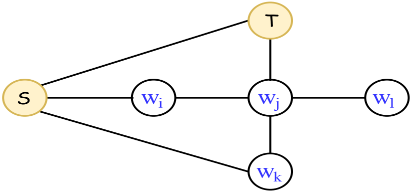



Figure: Undirected

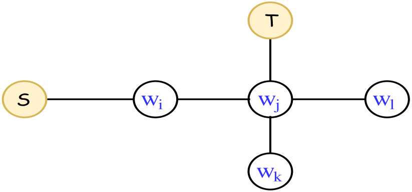



Figure: Spanning Tree

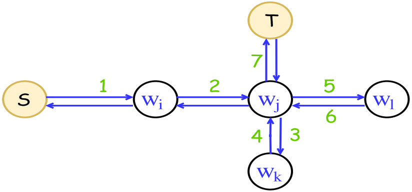



Figure: $S$-$T$ tour

# Hardness

- Feasibility via spanning tree $\Longrightarrow$ always feasible
- Approximation via metric TSP $\Longrightarrow$ L: ≈1.008[1], U: ≈1.53[2]

[1] Karpinski et al. J. Comput. Syst. Sci., 2015
[2] Andr as Seb o and Anke van Zuylen, FOCS 2016

# Hardness

- Feasibility via spanning tree $\implies$ always feasible
- Approximation via metric TSP $\implies$ L: $\approx 1.008$[1], U: $\approx 1.53$[2]
- FPT via subset TSP $\implies 2^k \cdot n^{\mathcal{O}(1)}$ (Klein and Marx, 2014)

[1] Karpinski et al. J. Comput. Syst. Sci., 2015
[2] Andr as Seb o and Anke van Zuylen, FOCS 2016

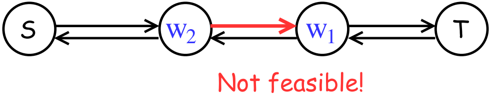
# Outline

# Ordered waypoint routing
## The problem

- A permutation is given, e.g. $w_1 w_2 \ldots w_k$
- Find the shortest route visiting every $w_i$, satisfying the permutation

# Ordered waypoint routing
## The problem

- A permutation is given, e.g. $w_1 w_2 \ldots w_k$
- Find the shortest route visiting every $w_i$, satisfying the permutation
- Not always feasible



# Ordered waypoint routing
## The problem

- A permutation is given, e.g. $w_1 w_2 \dots w_k$
- Find the shortest route visiting every $w_i$, satisfying the permutation
- Not always feasible
- Related to **E**dge **D**isjoint **P**ath **P**roblem

# Ordered waypoint routing

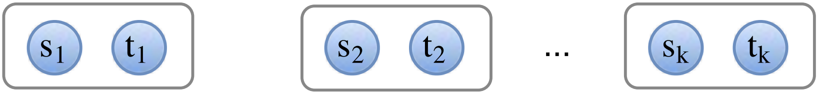
## The problem

- A permutation is given, e.g. $w_1 w_2 \ldots w_k$
- Find the shortest route visiting every $w_i$, satisfying the permutation
- Not always feasible
- Related to **E**dge **D**isjoint **P**ath **P**roblem
- NP-Hardness and feasibility via EDPP

# Edge Disjoint Path Problem
EDPP

**Edge Disjoint Path Problem $\in$ NP-Complete**

Find a set of pairwise edge-disjoint paths connecting every pair $(s_i, t_i), i = 1 \ldots k$

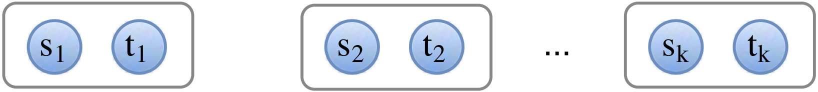


# Reduction
from EDPP

## Building the OWRP instance

The waypoints:

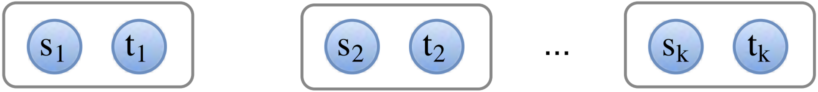
$s_1, t_1 \ldots s_i, t_i, s_{i+1}, t_{i+1} \ldots s_k, t_k$



# Reduction
from EDPP

## Building the OWRP instance

The waypoints:
$S = s_1, t_1 \dots s_i, t_i, s_{i+1}, t_{i+1} \dots s_k, t_k = T$



# Reduction
## from EDPP

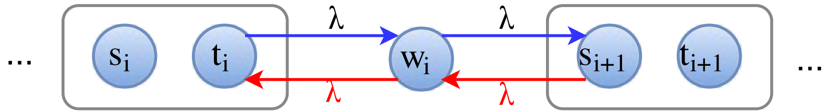
### Building the OWRP instance

The waypoints:
$S = s_1, t_1 ... s_i, t_i, s_{i+1}, t_{i+1} ... s_k, t_k = T$

# Reduction
from EDPP

## Building the OWRP instance

The waypoints:
$S = s_1, t_1 \ldots s_i, t_i, w_i, s_{i+1}, t_{i+1} \ldots s_k, t_k = T$

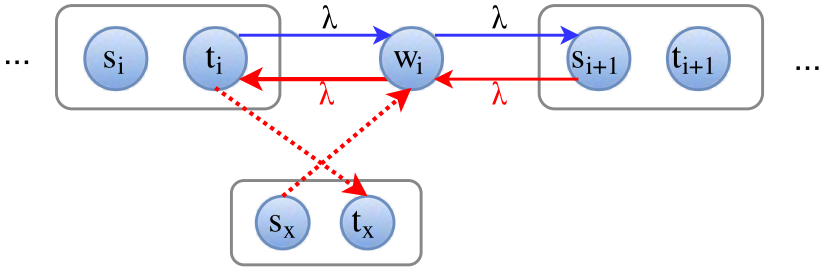


# Reduction
## from EDPP

### Building the OWRP instance

The waypoints:

$S = s_1, t_1 \ldots s_i, t_i, w_i, s_{i+1}, t_{i+1} \ldots s_k, t_k = T$

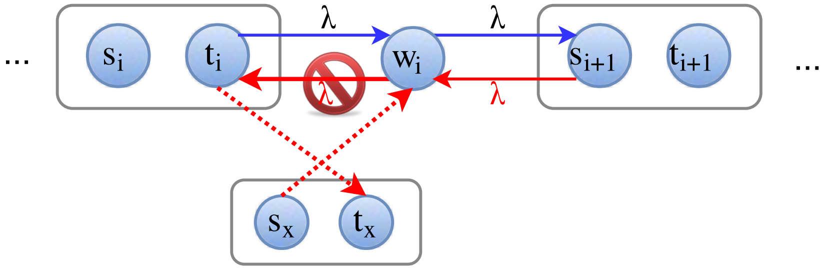


# Reduction
## from EDPP

### Building the OWRP instance

The waypoints:

$S = s_1, t_1 \ldots s_i, t_i, w_i, s_{i+1}, t_{i+1} \ldots s_k, t_k = T$

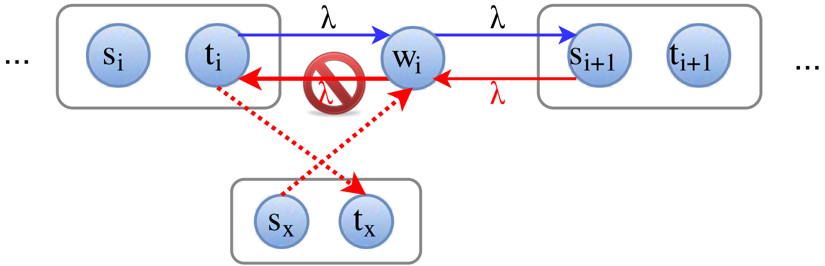


# Reduction
from EDPP

## Building the OWRP instance

The waypoints:

$S = s_1, t_1 \ldots s_i, t_i, w_i, s_{i+1}, t_{i+1} \ldots s_k, t_k = T$



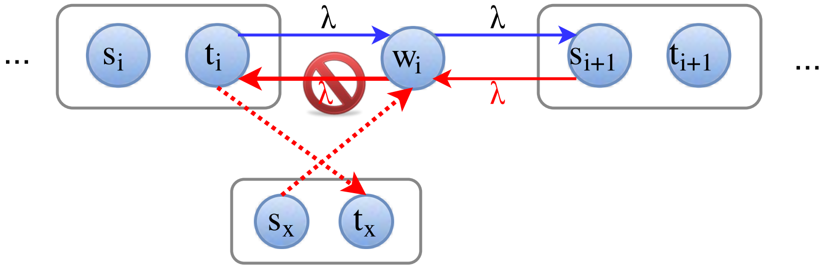

- Set $\lambda$ large enough

# Reduction
from EDPP

## Building the OWRP instance

The waypoints:

$S = s_1, t_1 \ldots s_i, t_i, w_i, s_{i+1}, t_{i+1} \ldots s_k, t_k = T$




- Set $\lambda$ large enough
- OWRP chooses a backward edge $\iff$ EDPP is not feasible

# Ordered waypoint routing
## Results

- General graphs: $k \in \mathcal{O}(1) \implies$ feasibility $\in$ P, via EDPP (A. Jarry et al., 2009)

# Ordered waypoint routing
## Results

- General graphs: $k \in \mathcal{O}(1) \implies$ feasibility $\in$ P, via EDPP (A. Jarry et al., 2009)
- Trivial on trees

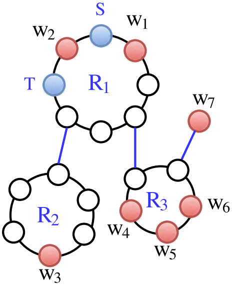
# Ordered waypoint routing
## Results

- General graphs: $k \in \mathcal{O}(1) \implies$ feasibility $\in$ P, via EDPP (A. Jarry et al., 2009)
- Trivial on trees
- On ring: $\min_e c_e \in \mathcal{O}(1) \implies$ dynamic programming $\in$ P

# Ordered waypoint routing

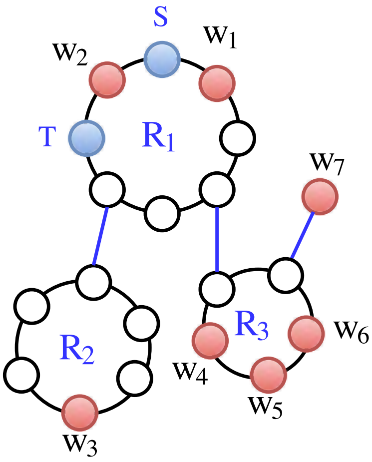
## Results

- General graphs: $k \in \mathcal{O}(1) \implies$ feasibility $\in$ P, via EDPP (A. Jarry et al., 2009)
- Trivial on trees
- On ring: $\min_e c_e \in \mathcal{O}(1) \implies$ dynamic programming $\in$ P
- *Cactus graph*: tree of rings



# Ordered waypoint routing

## Cactus: a tree of rings

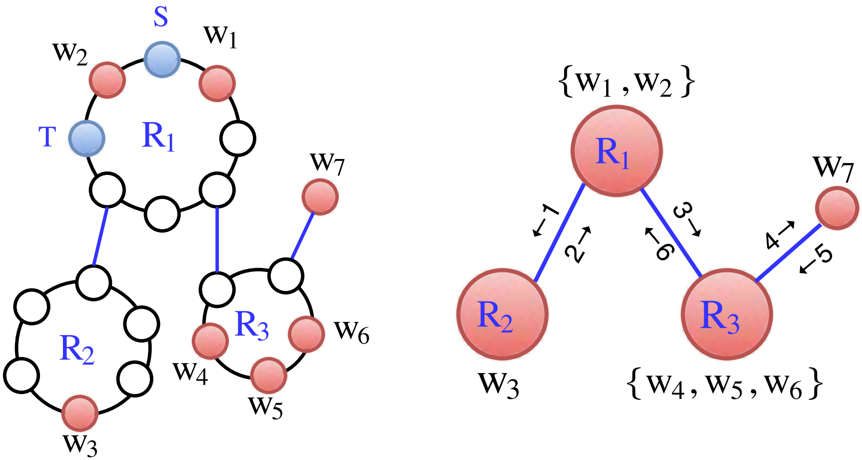



Step 1: solve the tree contraction given

$(S = R_1), R_2, R_3, W_7, (R_1 = T)$

# Ordered waypoint routing

## Cactus: a tree of rings

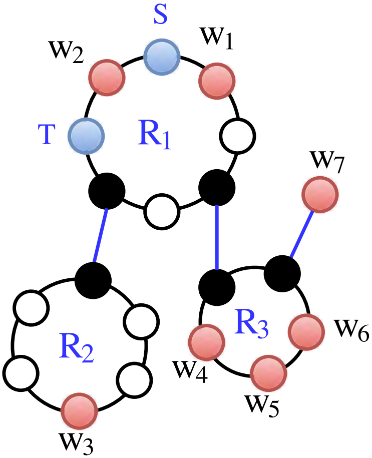


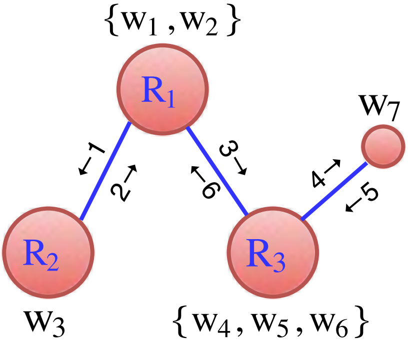

Step 1: solve the tree contraction given

$(S = R_1), R_2, R_3, W_7, (R_1 = T)$

# Ordered waypoint routing

## Cactus: a tree of rings




- Step 2: mark the port nodes (shown in back) as new waypoints

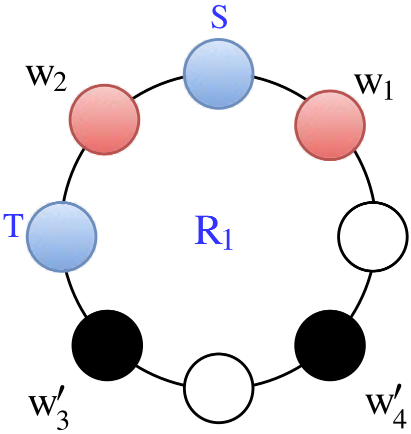
# Ordered waypoint routing

## Cactus: a tree of rings

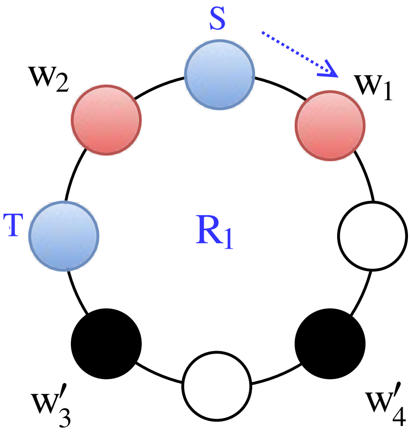
- Step 2: mark the port nodes (shown in back) as new waypoints
- Step 3: solve OWRP on each ring separately

# Ordered waypoint routing

## Cactus: a tree of rings

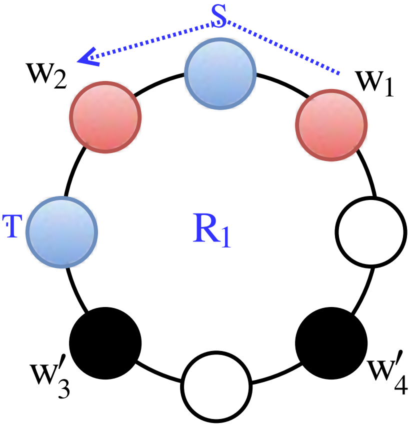


# Ordered waypoint routing
## Cactus: a tree of rings



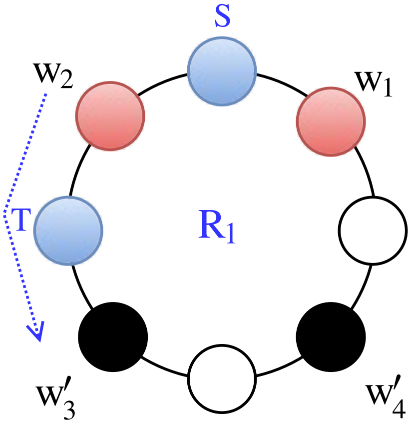
# Ordered waypoint routing

## Cactus: a tree of rings



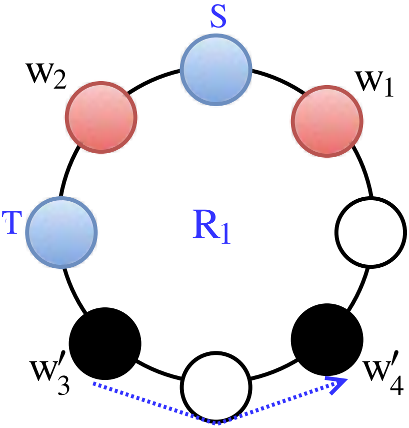
# Ordered waypoint routing

## Cactus: a tree of rings



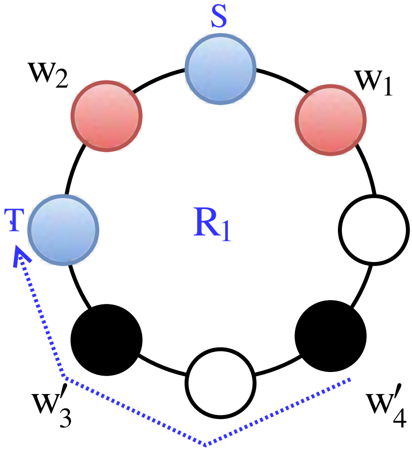
# Ordered waypoint routing

## Cactus: a tree of rings



# Ordered waypoint routing

## Cactus: a tree of rings



# Summary

Table: Ordered WRP

| | General | $k \in \mathcal{O}(1)$ | Tree | $c_e \in \mathcal{O}(1)$ |
|---|---|---|---|---|
| Feasibility | open | P | P | Ring$\in$ P |
| Optimality | NP-Hard | open | | |

Table: Unordered WRP

| | General | $k \in \mathcal{O}(\frac{\log n}{\log \log n})$ |
|---|---|---|
| Feasibility | P | |
| Optimality | NPH,APX,FPT | P |

# Open Questions

- Other special graph classes, e.g.: bidirected planar graphs
- Feasibility hardness for the ordered variant (we gave the optimality hardness)