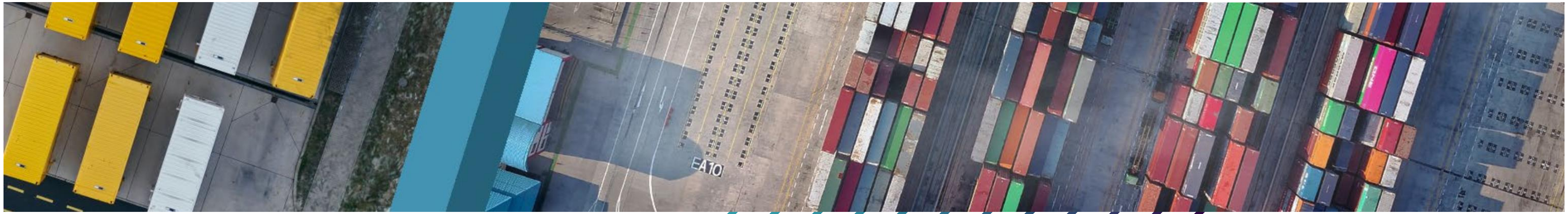
AUSTRIAN INSTITUTE OF TECHNOLOGY
TOMORROW TODAY

On-demand automated guided vehicles in yard logistics

Ulrike Ritzinger, Bin Hu, Martin Reinthaler
AIT Austrian Institute of Technology, Center for Energy

Eurocast 2024, Las Palmas de Gran Canaria, 2024-02-26

AWARD
Scaling autonomous logistics



AWARD

H2020 Project

- **A**ll **W**eather **A**utonomous **R**eal logistics operations and **D**emonstrations
- Efficient and safe connected and automated heavy-duty vehicles in real logistics operations
- Potential to address key issues in commercial transportation:
  - Lack of qualified drivers in Europe
  - Number of accidents with trucks unacceptably high (due to driver failures)
  - Utilization of freight transport capacity is below 50%

AWARD has received funding from the European Union's Horizon 2020 research and innovation program under Grant Agreement No 101006817

AWARD
Scaling autonomous logistics

AWARD
H2020 Project

- Driving autonomously **on all roads at all speeds** with all sorts of **known and unknown** hazards certainly is and will be a challenge

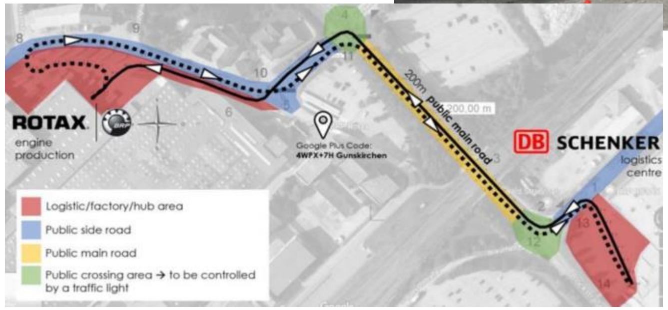
- Developing and operating **safe autonomous transportation systems**
  - In a wide range of real-life logistic use cases in a variety of different scenarios.
  - Solution will be based on multiple sensor modalities and an embedded teleoperation system **to address 24/7 availability**
  - Vehicles will be deployed, integrated and operated in a **variety of real-life use cases** to validate their value

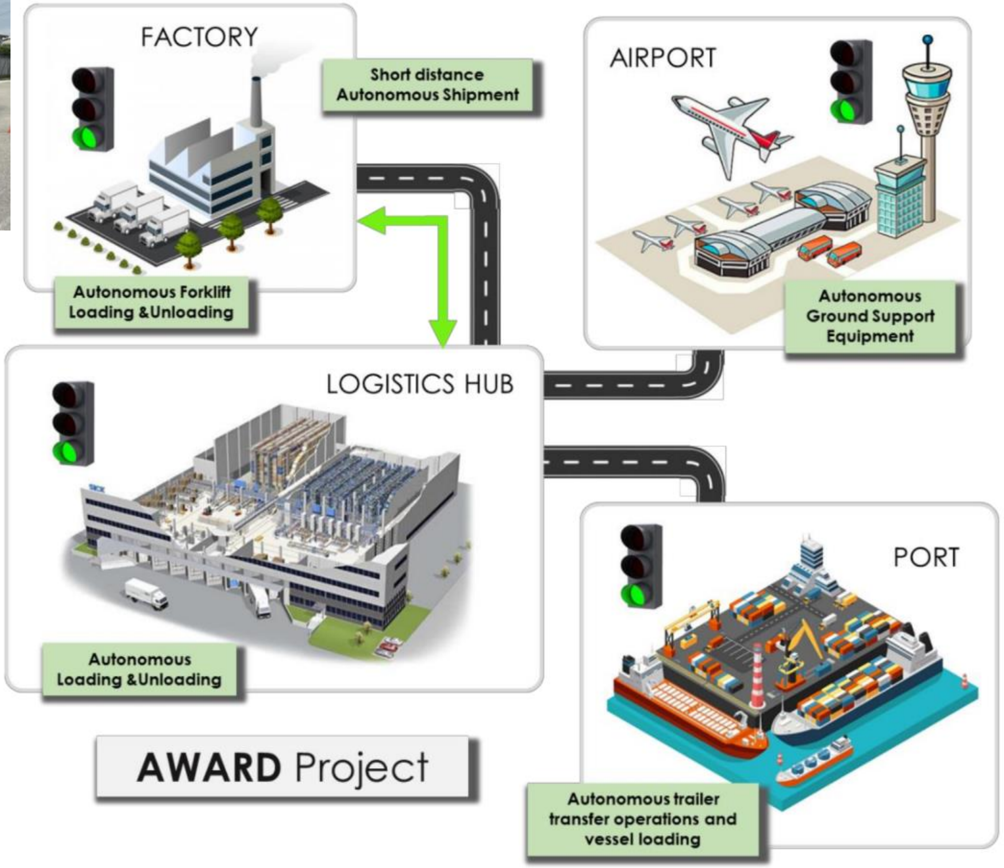
AWARD has received funding from the European Union's Horizon 2020 research and innovation program under Grant Agreement No 101006817

AWARD
Scaling autonomous logistics

AWARD

4 Use Cases in Europe





AWARD
Consortium

AWARD
Scaling autonomous logistics

Sensors	Autonomous Driving System	Heavy-Duty Vehicles manufacturer	Fleet Management & Supervision	End-users Industrial sites
Continental The Future in Motion	EASY MILE	KAMAG	APPLIED AUTONOMY	DFDS
NAVTECH RADAR	Continental The Future in Motion	KION GROUP	DFDS	AVINOR
ADASKY	AIT AUSTRIAN INSTITUTE OF TECHNOLOGY	SAS SMART AIRPORT SYSTEMS	ottopia	ROTAX
FORESIGHT	VTT	TERBERG BENSCHOP	EASY MILE	DB SCHENKER
EASY MILE	LCM LINZ CENTER OF MECHATRONICS		AIT AUSTRIAN INSTITUTE OF TECHNOLOGY	KION GROUP
	ITS Norway			
	UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA			
	RÉPUBLIQUE FRANÇAISE Cerema			
	ottopia			

Certification and proving grounds

CERTX · Digitrans · EASY MILE · VTT · RÉPUBLIQUE FRANÇAISE Cerema

Impact assessment, business modelling and regulatory frameworks

ITS Norway · CARA EUROPEAN CLUSTER FOR MOBILITY SOLUTIONS · business upper austria · RU · UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA · austriatech Mobility in motion · enide · VTT · FRANCE AVIATION CIVILE SERVICES

AWARD

Agenda

- **Fleet Management**
  - Generate a schedule for vehicles
  - Consider different weather and road conditions
  - Non-blocking vehicles

- Time-dependent Shortest Path Problem
- Time-dependent Pickup and Delivery Problem with Time Window
- Conflict-free Large Neighboorhood Search

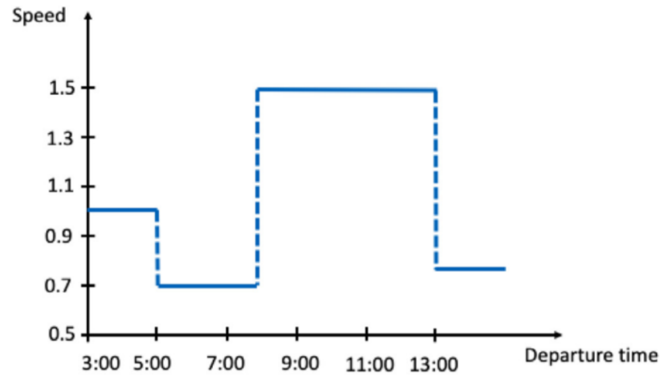
PROBLEM DESCRIPTION

Time-dependent road network

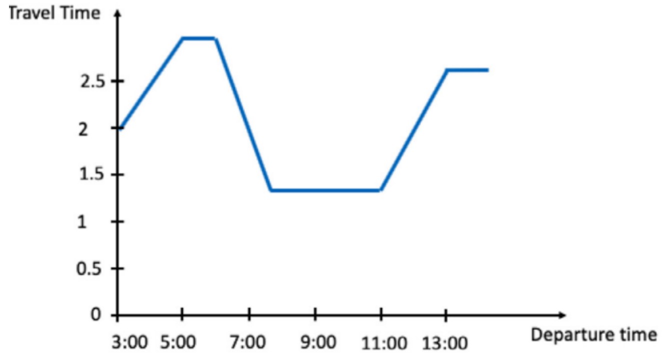
- Road network is a directed graph $G = (V, A)$
  - Set of vertices $V = \{0, \dots, n\}$
  - Set of arcs A, road segments between a pair of vertices

- Each arc $(i, j) \in A$ is associated with a
  - Distance d_{ij}
  - Time-dependent speed function $\mathrm{v_{ij}}$: $t \rightarrow R^+$
  - Time-dependent cost function: $c_{ij}: t \rightarrow R^+$ (= travel time)

- Each speed function is a stepwise function from which a piecewise travel time function can be derived



(a) Speed function



(b) Travel time function

(Gmira et al., EJOR 2021)

PROBLEM DESCRIPTION

Time-dependent Pickup and Delivery Problem with Time Windows

- Set of n requests r each with:
  - Pickup and delivery node pair $\{i, n+i\} \in V$
  - Demand for pickup q_{i+} and delivery q_{i-}
  - Service time for pickup s_{i+} and delivery s_{i-}
  - Time window $tw_r = [a_i, b_i]$ (either at pickup or delivery, soft)

- Set of m vehicles K starting at parking positions $z_k \in V$ with capacity C_k

- TDPDPTW is to generate routes for all vehicles:
  - Feasibility: Conflict-free, time windows, capacity constraint
  - Objective: Maximize the number of served requests
    Minimize the total lateness and travel time of the routes

SOLUTION APPROACH I

Time-dependent Dijkstra‘s algorithm with new weighting function

- Time-dependent variant of Dijkstra‘s algorithm
  - Minimum cost path between a source and destination node at a departure time t
  - Forward: travel time for departure time t at source
  - Backward: travel time for arrival time t at destination

- Situations defined by operators
  - Time period with a start and an end time $p = [a_p, b_p]$
  - Speed defined over this interval $v_{ij}^{+}(p)$: $v_{ij}(t)$ for all $t \in [a_p, b_p]$
  - Consider special case when situation is “road closed“
    - Either waiting until b_p or detour

SOLUTION APPROACH I

Time-dependent Dijkstra with new weighting function

- Award weighting function
  - Extension of procedure in Ichoua et al., 2003
  - From current node to successor through arc (i, j) at time t
  - Special case road closed
    - No distance covered when waiting

Algorithm 1: Travel costs to a successor of a node.

Input : arc (i, j), arrival time t, $p = 0$

Output: costs c_{ij}

while $t \geq a_p$ **do**

  $p = p + 1$

$c' = b_p - t$

$d' = c_{ij} = 0$

$d = c' \cdot v^+_{ij}(p)$ //if $v^+_{ij}(p)$ is road closed $\rightarrow d = 0$

while $d \leq d_{ij}$ **do**

  $c_{ij} = c_{ij} + c'$

  $p = p + 1$

  $d' = d$ //if $v^+_{ij}(p)$ is road closed $\rightarrow d' = 0$

  $c' = b_p - a_p$

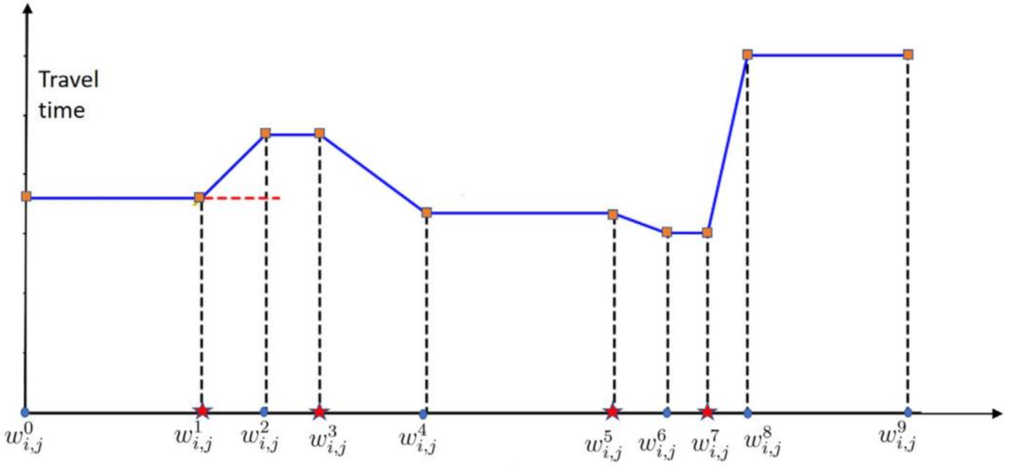
  $d = d' + (c' \cdots v^+_{ij}(p))$

return $c_{ij} + ((d_{ij} - d')/v^+_{ij}(p))$

SOLUTION APPROACH II

Time-dependent distance matrix

- Efficiently store travel times between two nodes for a given time
  - Entry for every possible departure → high time resolution (ms)
- Changes in travel time depend on given situations (periods)
  - Calculate the times when the speed changes on arc (ij)
    → Set of breakpoints for arc (ij): $W_{(ij)} = \{w^0_{(ij),}\dots, w^c_{(ij)}\}$
- Entry in matrix for not changing travel times
  - Travel cost $c_{ij}^{w^0_{(ij)},w^1_{(ij)}}$ between i and j
    for all time points $t \in [w^0_{(ij)}, w^1_{(ij)}]$
- Travel costs of other instants stored in cache



(Pan et al., COR 2021)

SOLUTION APPROACH II

Compute time-dependent distance matrix

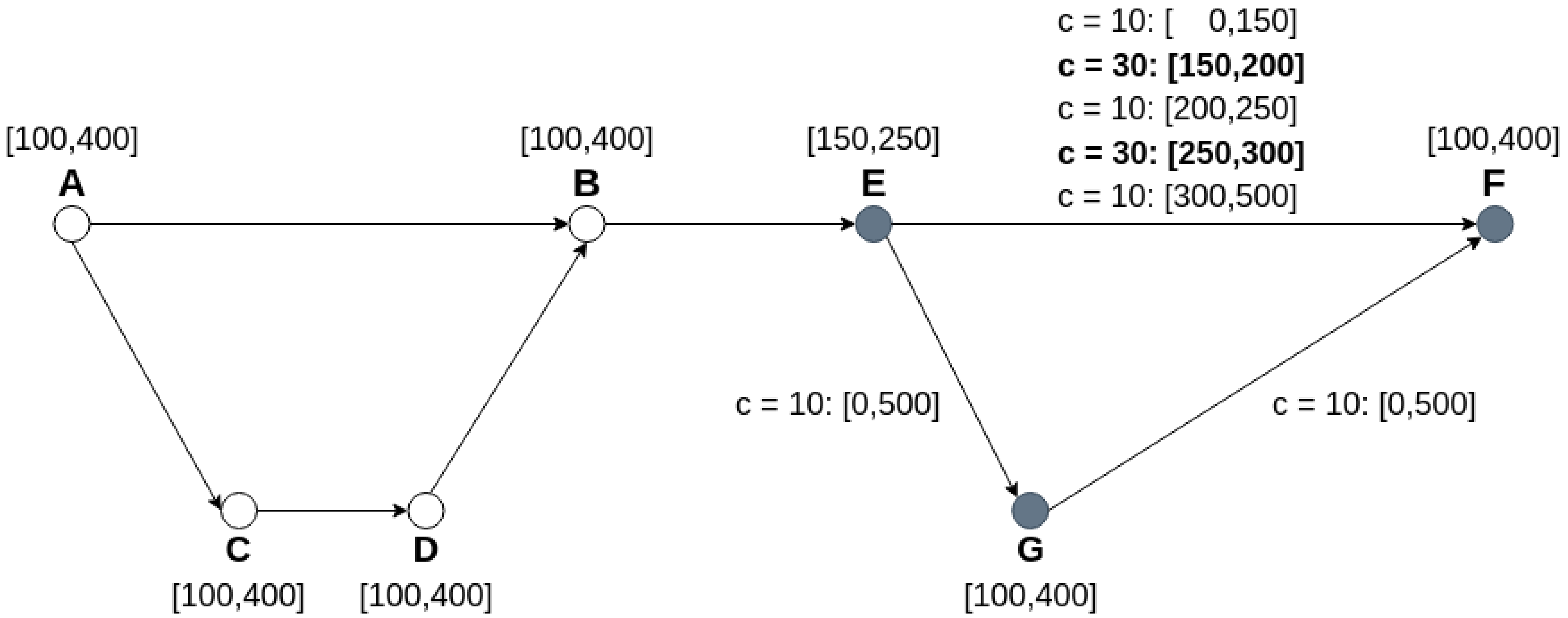
- Calculation of distance matrix entry between node x and node y
  - Determine all departure times $e \in E$ in x and their resulting paths p_{xy}^{e+} (increasing)
  - If there is no change (travel cost <u>and</u> path) $\rightarrow$ entry to matrix

- Determine all departure times $e \in E$ in x:
  - Compute set of paths P with arrival in y at a_y and b_y (backward Dijkstra)
  - For all arcs (i, j) on the paths plus for all incoming arcs $(i', j), i' \neq i$
    $\rightarrow$ compute set of breakpoints $W_{(ij)}$
  - For all breakpoints calculate departure in x and path from x to i, i' respectively
    - Departure e in x: backward Dijkstra with arrival in i at $w_{(ij)}^{c}$
    - Calculate path from x to i at departure e with forward Dijkstra

SOLUTION APPROACH II

Time-dependent distance matrix

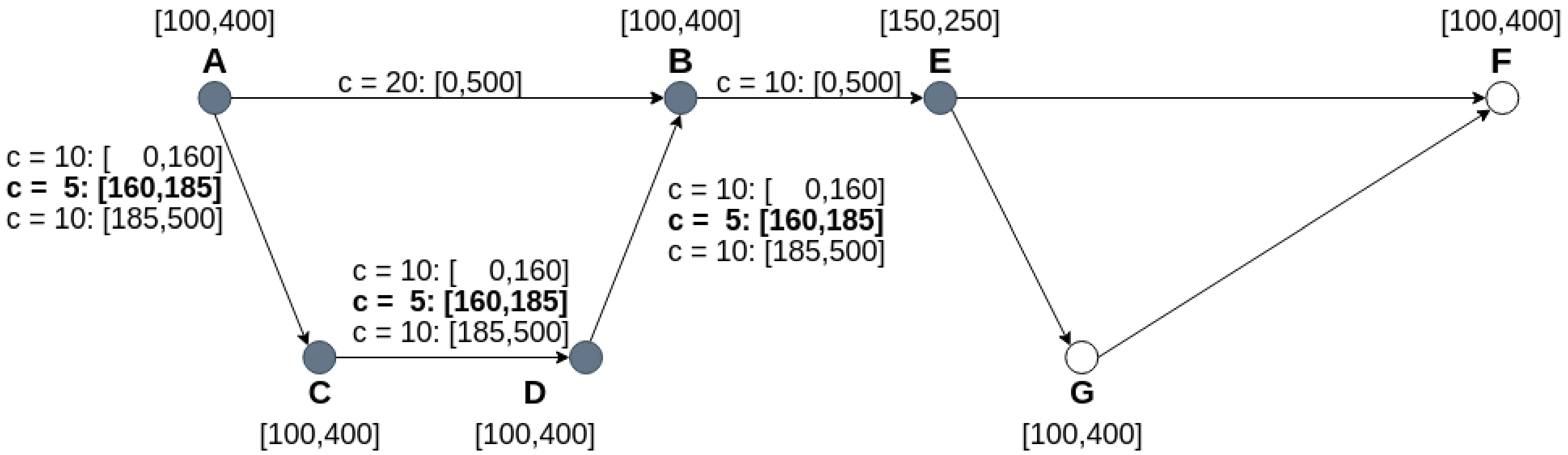
- Calculate matrix entry for **route from node E to node F**
  - Departure at 140 → path is EF with costs 10
  - Departure at 150 → path is EG.GF with costs 20 (detour)



SOLUTION APPROACH II

Time-dependent distance matrix

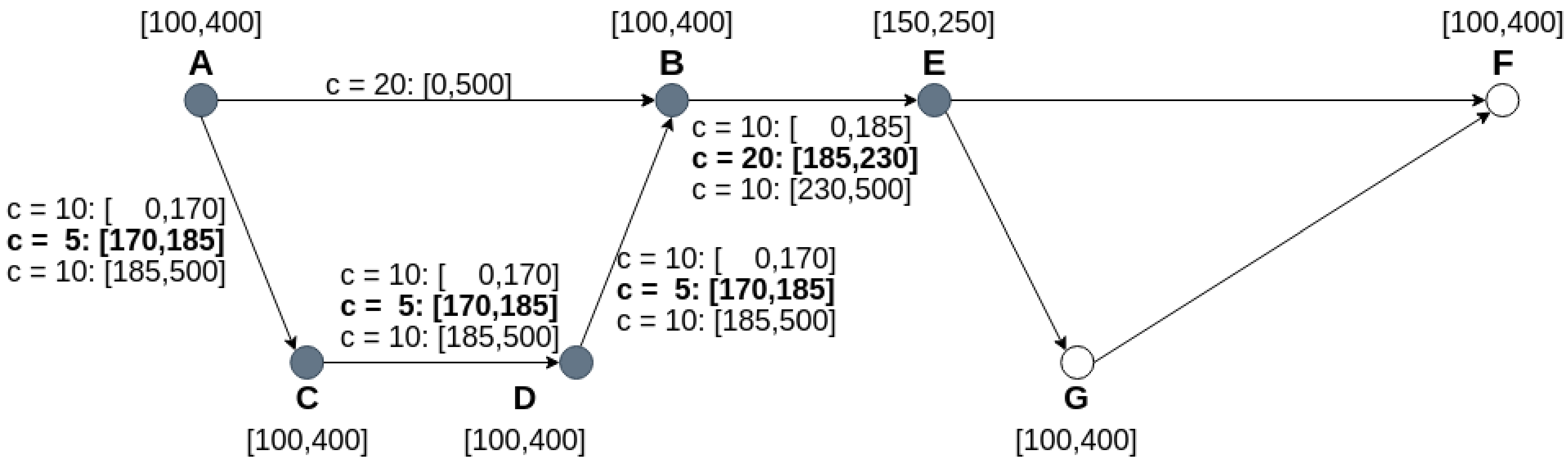
- Calculate matrix entry for **route from node A to node E**
  - Example that considering incoming arcs is necessary:
    - Departure at 120 and 220 → path is AB-BE with costs 30 (**no situations on path!**)
    - Departure at 160 → path is AC-CD-DB-BE with costs 25 (detour)



SOLUTION APPROACH II

Time-dependent distance matrix

- Calculate matrix entry for **route from node A to node E**
  - Example that considering path not only departure in A is necessary:
    - Departure at 160 → path is AB-BE with costs 35
    - Departure at 170 → path is AC-CD-DB-BE with costs 35 (**same costs put different path!**)



SOLUTION APPROACH III

TW Calculator TDPDPTW (soft)

- Improvement method to calculate time windows for the VRPTW
- Introduced by Vidal et al., 2013
- Efficiently calculate penalties according to time window constraints
  - Forward and backward
  - Concatenation

Algorithm 1 : calculateTimeWindowFeasibility(Sequence σ)

$i = first(\sigma);\ TW = 0;\ E = e_0;\ L = l_0;\ D = s_0;$
for $\forall j$ in $\sigma \setminus 0$ **do**
  $\Delta = D - TW + t_{ij};$
  $\Delta^{WT} = max\{0,\ e_j - \Delta - L\};$
  $\Delta^{TW} = max\{0,\ E + \Delta - l_j\};$
  $D = D + s_j + \Delta^{WT} + t_{i,j};$
  $TW = TW + \Delta^{TW};$
  $E = max\{e_j - \Delta, E\} - \Delta^{WT};$
  $L = min\{l_j - \Delta, L\} + \Delta^{TW};$
  $i = j;$
end for

SOLUTION APPROACH III

TW Calculator TDPDPTW (soft)

- Adapt the sequence-based TW calculation from Vidal
- Store values (duration, lateness) for earliest (E) **and** latest departure (L)
- Calculate departure times for E and L
  - When $E = L \longrightarrow$ use forward extension of Vidal
  - Otherwise, if E and/or L changes:
    - Update values and check if there are changes in travel costs
    - Worst case: Update until the beginning of the sequence
- Determine best departure in depot for the node sequence

SOLUTION APPROACH IV

Conflict-free Large Neighborhood Search (LNS)

- (Adaptive) Large Neighborhood Search as in Ropke and Pisinger, 2006
  - Remove requests from solution (up to 60%) and insert them again
    - Destroy operators: Random and Worst
    - Repair operators: Greedy Repair, Regret Repair

- Only conflict-free solutions are allowed
  - Conflict: if vehicles blocks another vehicle at a stop
  - Whenever feasibility of solution is calculated:
    - Check for each vehicle route if it passes a stop where another vehicle stays
      → Solution is infeasible

PRELIMINARY RESULTS

Time-dependent PDPTW with no conflicts

- Instances on road network generated by an operator
  - Road network consists of 173 nodes and 252 edges
  - 30, 40, 50 customer requests, 5 or 10 situations, and 3 vehicles
- First results show benefit of implementing a time-deent distance matrix
  - Percentage of runtime for travel time calculation to total runtime

# requests	5 situations		10 situations	
	No matrix	Matrix	No matrix	Matrix
30	3.08	1.67	3.61	1.46
40	2.34	1.01	2.66	0.79
50	4.20	2.01	4.09	1.34

PRELIMINARY RESULTS

Time-dependent PDPTW with no conflicts

- Efficient time window calculation with time-dependent travel times
  - Implemented and tested for forward method (many possibilities)

    → Extension to backward calculation (allowing concatention in constant time)

    → Compare to other time window calculations

- Conflict-free vehicle routes are generated for all instances
  - Dealing with larger instances (more vehicles):

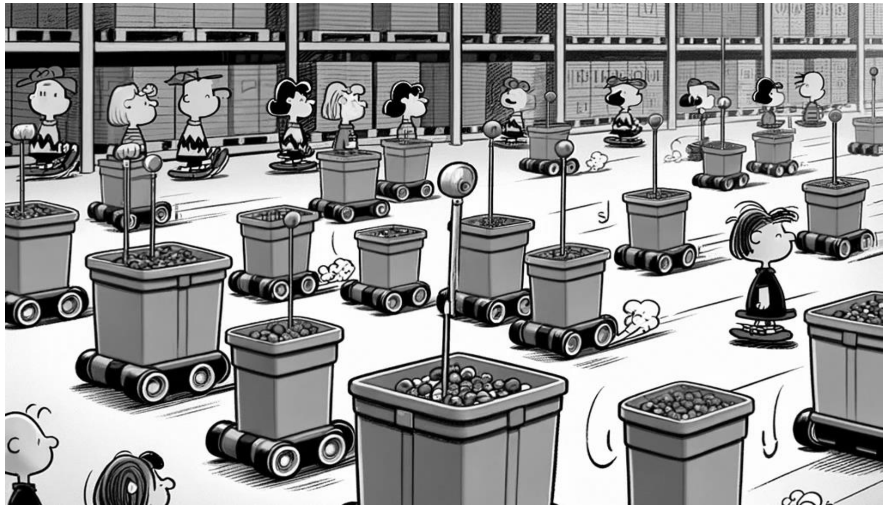
    → Operator: Relocate vehicle to parking position

THANK YOU!

Ulrike Ritzinger

Eurocast2024, 2024-02-26

ulrike.ritzinger@ait.ac.at



AWARD
Scaling autonomous logistics

AUSTRIAN INSTITUTE OF TECHNOLOGY
TOMORROW TODAY