technische universität
dortmund

department of
computer science

Bienen-inspiriertes Straßenverkehrsrouting

Sebastian Lehnhoff
Sebastian Senge
Anca M. Lazarescu

http://ls3-www.cs.tu-dortmund.de

Lehrstuhl für Betriebssysteme
und Rechnerarchitektur

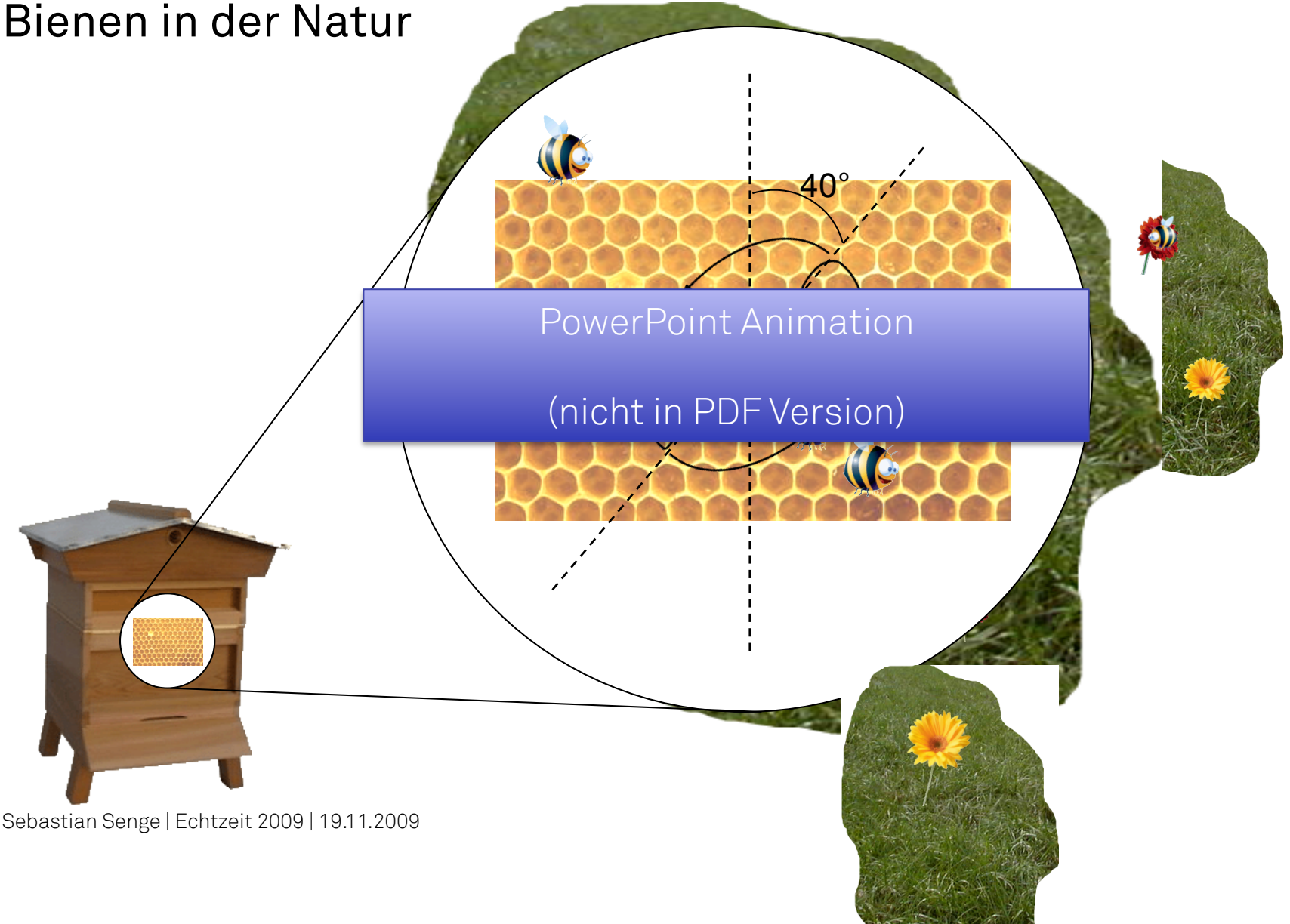
Agenda

- Einleitung
- Hintergrund
  - Bienen in der Natur
  - BeeHive-Algorithmus in Computernetzwerken
- BeeJamA: Verteiltes Verkehrsmodell
  - Infrastruktur und Architektur
  - Ebenenmodell
  - Scheduler
- Ausblick
- Fazit

Einleitung

- Verkehrsstau – ein hoch dynamisches und verteiltes Problem
- Schwarmintelligenz:
  - Kollektives Verhalten von einfachen Agenten
  - „Verteilte künstliche Intelligenz“
- Ansatz:
  - Futtersuche der Bienen Vorbild zur Minimierung von Reisezeiten durch Stauvermeidung
  - Hop-to-Hop-Routing bei dynamischen Deadlines

Bienen in der Natur



BeeHive Routing in Computernetzwerken

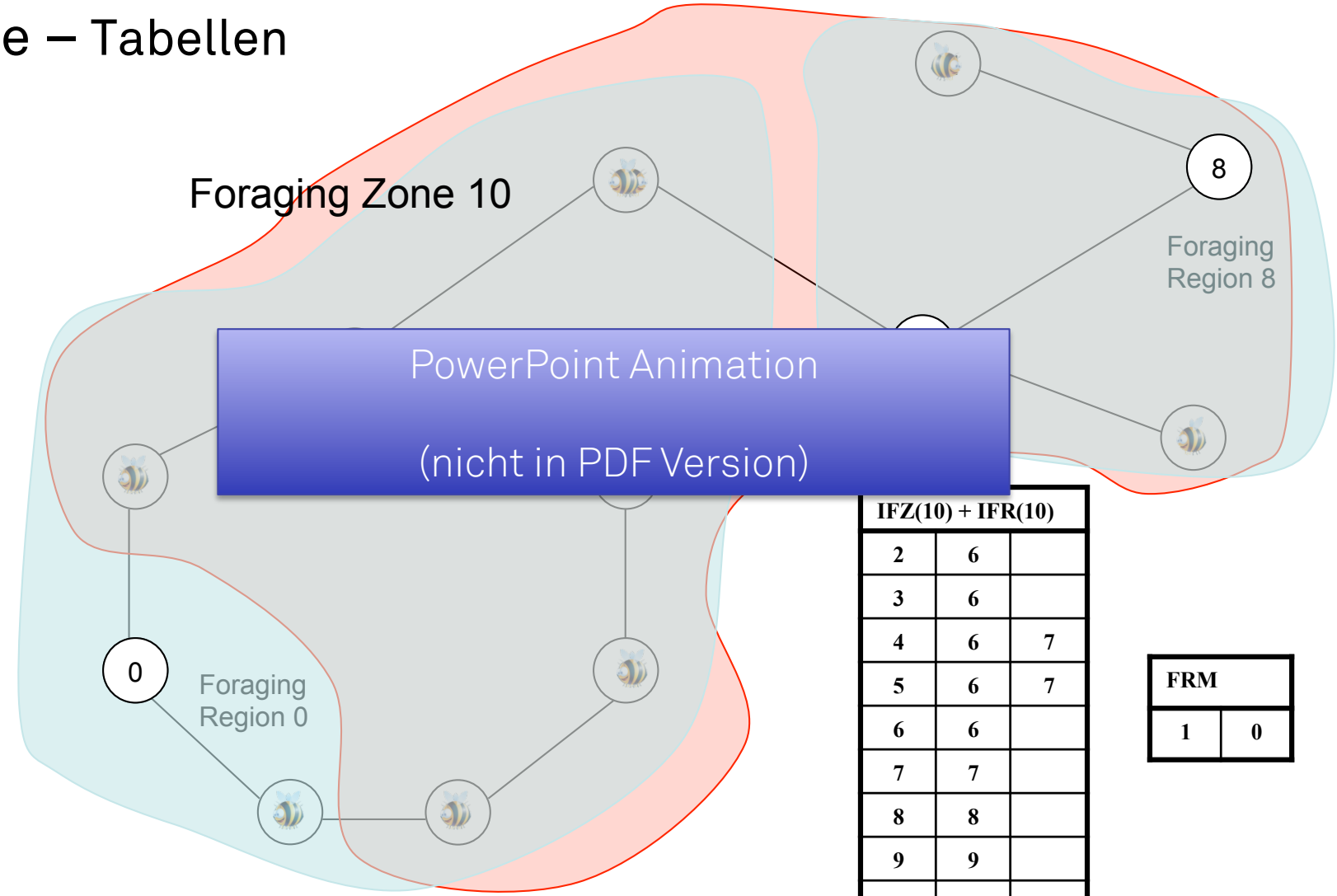
- Bienen-inspiriertes Multiagentensystem (MAS)
- Dynamischer Multipath-Routingalgorithmus
- Dezentrale Kontrolle
- Kleine Routingtabellen
- Hohe Skalierbarkeit, Flexibilität und hoher Durchsatz

- Grundlage für unser verteiltes Verkehrsroutingsystem BeeJamA

BeeHive – Abbildung der Konzepte (M. Farooq, 2006)

Nature	Computer network packet routing
Scouts	Bee Agents
Foragers	Data Packets
Dance Floor	Routing Table
Hive	Node
Dance	Propagation Delay (Distance) Queuing Delay (Direction)
Vicinity of the Hive	Foraging Zone of a source node
Vicinity of the Food Source	Foraging Region of a destination node (extended search structure)

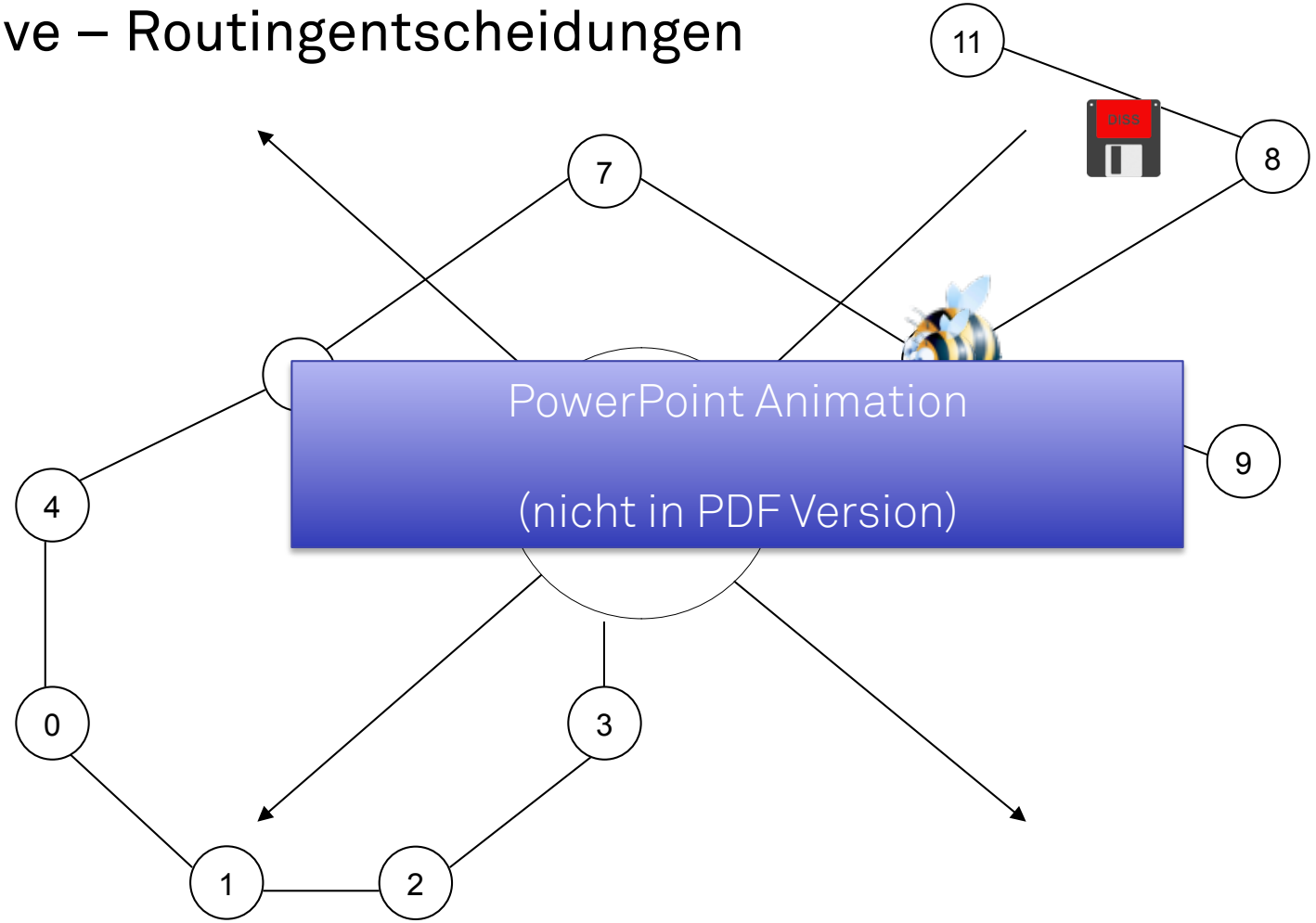
BeeHive – Tabellen



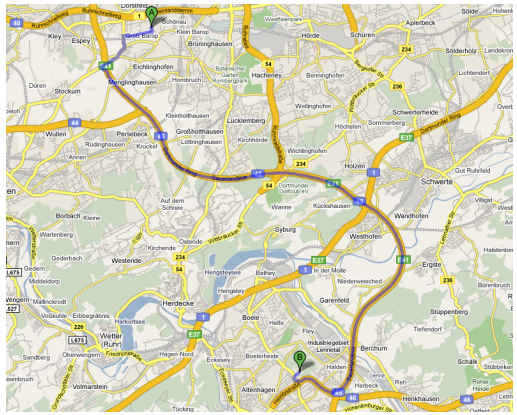
IFZ(10) + IFR(10)		
2	6	
3	6	
4	6	7
5	6	7
6	6	
7	7	
8	8	
9	9	
11	8	
0	6	7

FRM	
1	0

BeeHive – Routingentscheidungen

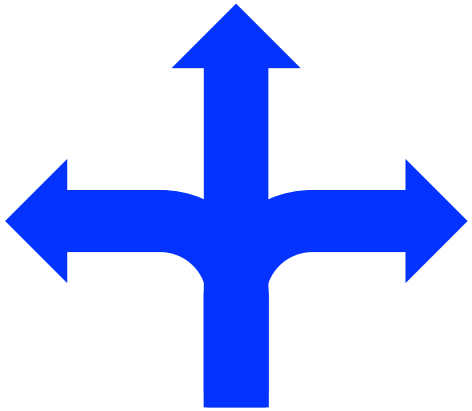


Ziele des Routing im Straßenverkehr



1. Durchschnittliche Reisezeit wird minimiert
   - Unerwartete Staus → flexible Umleitungsentscheidungen
   - Keine Stauverschiebung auf untergeordnete Straßen
2. (Weiche) Echtzeitbedingungen:
   - Auslastungen werden schnell propagiert
   - Individuelle Routinganweisung kommt rechtzeitig vor jeder Kreuzung
3. Skalierbar auf beliebig große Gebiete

→ Adaptives, vollständig verteiltes Management nötig



BeeJamA – System

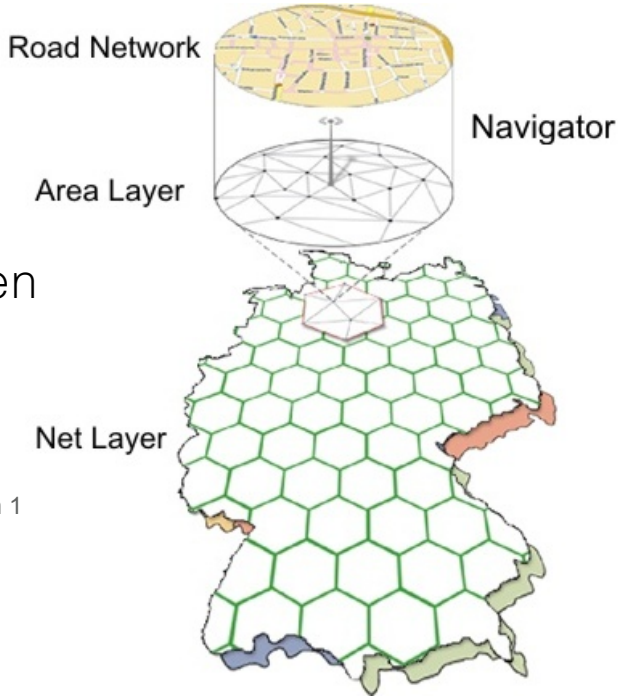
- Verteiltes Verkehrsmanagement
  - Vehicle-2-Infrastructure (V2I) Architektur
  - Floating Car Data zur Kantenbewertung
  - MAS zum Propagieren der Bewertungen
  - Probabilistisches Hop-to-Hop-Routing
  - Ebenenmodell → Hierarchie

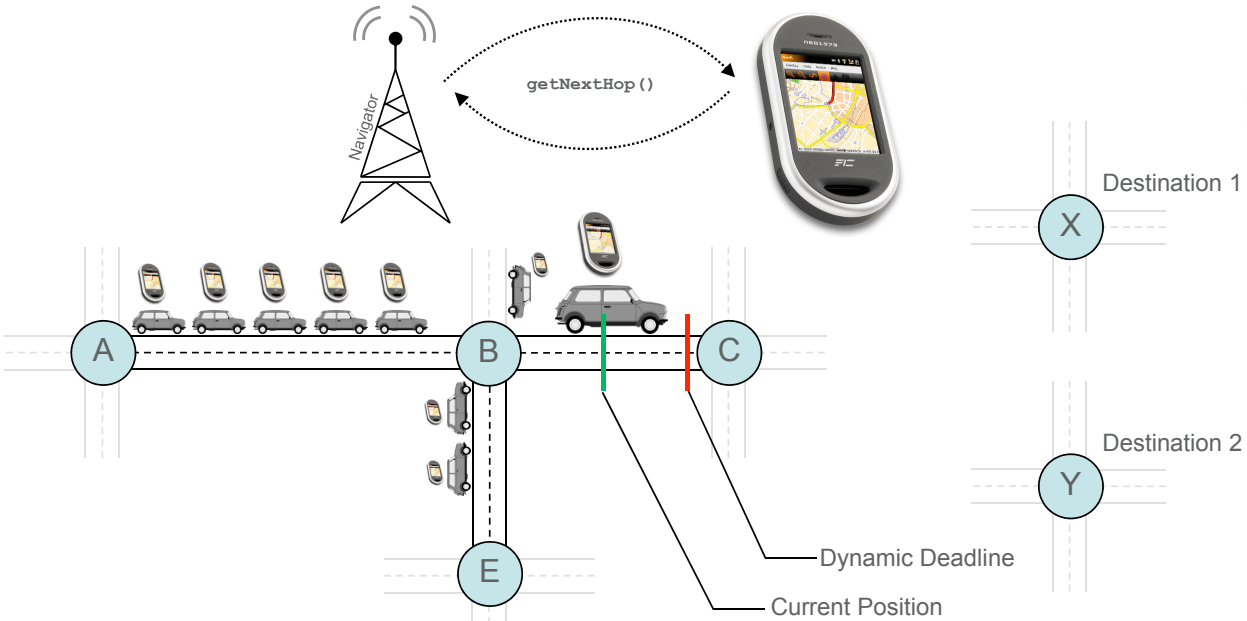
→ Keine globalen Informationen nötig

- Echtzeitherausforderung: Dynamische Deadlines

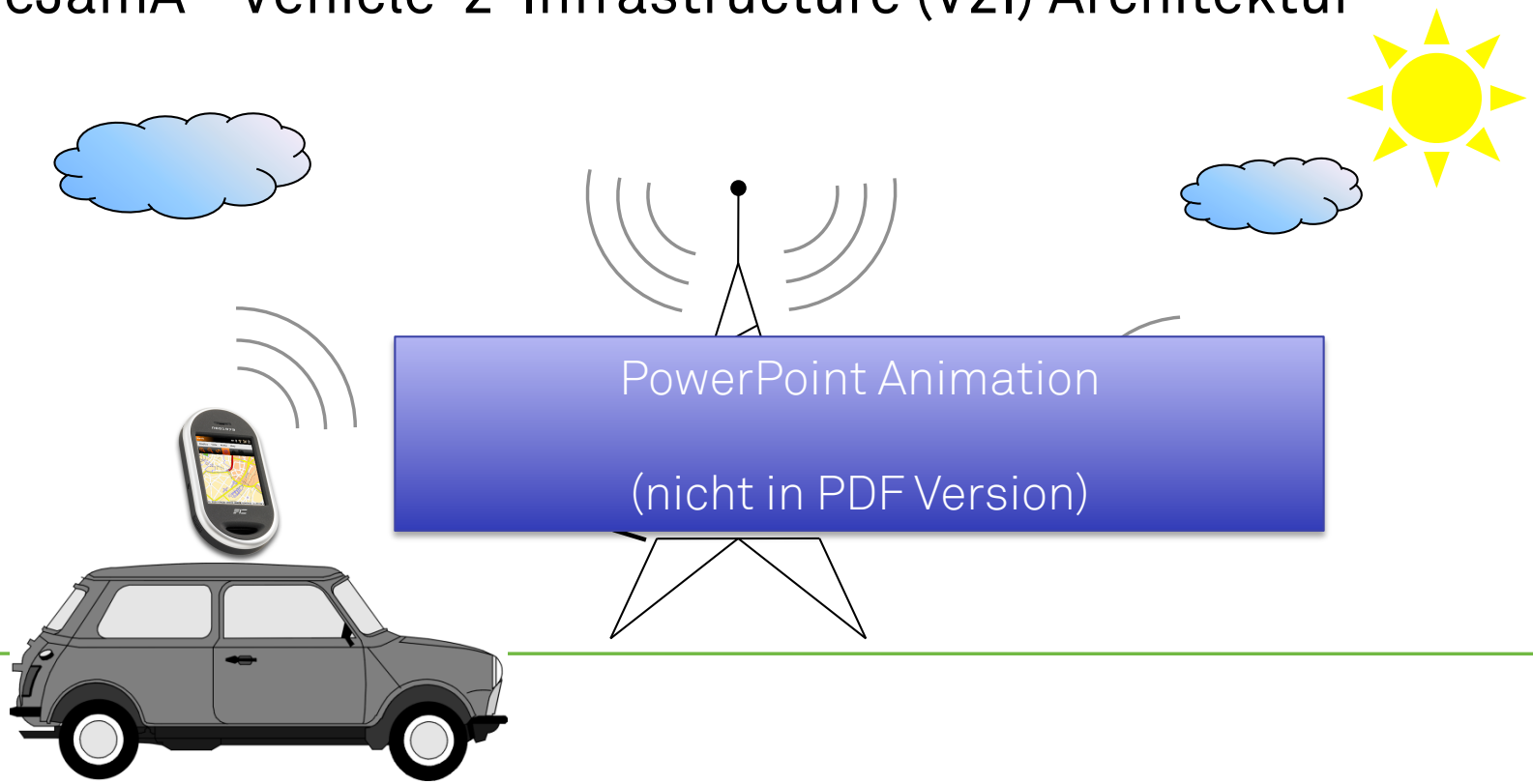
BeeJamA – Regional verantwortliche Navigatoren

- Ein Navigator pro Bereich (area)
- Sammelt alle Informationen eines Bereichs
- Verwaltet die Routingtabellen eines Bereichs
- Kontinuierliche Kommunikation mit den Fahrzeugen



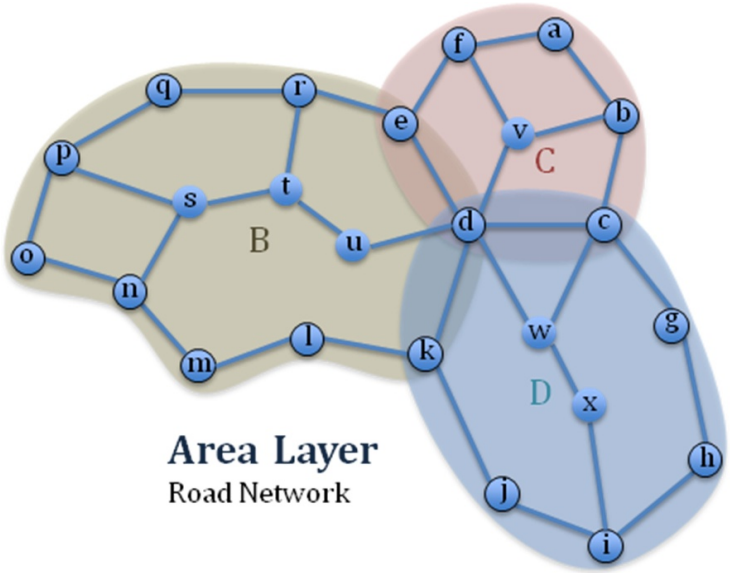


BeeJamA - Vehicle-2-Infrastructure (V2I) Architektur



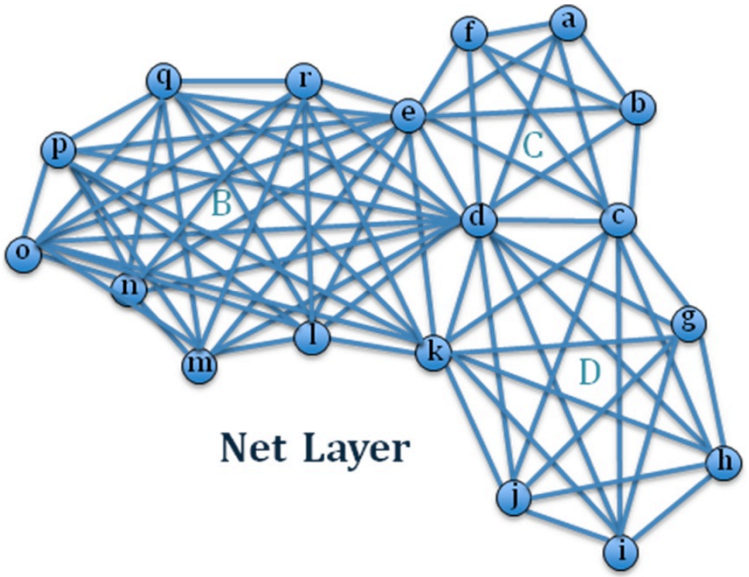
- Fahrzeuge als mobile Stausensoren → Floating Car Data (FCD)
- Kantenbewertung basierend auf aktuellen Verkehrsinformationen
  - Propagiert durch Bienenagenten

BeeJamA – Ebenenmodell



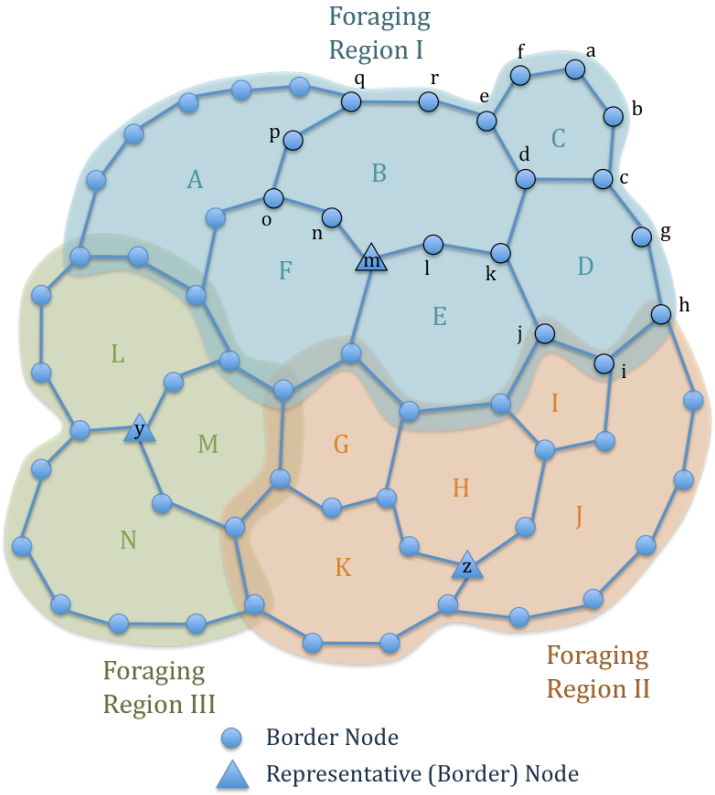
- Knoten = Kreuzungen, Kanten = Straßen
- Ein Navigator pro Bereich
- Navigatoren können kommunizieren
- IFZ_{area} Routingtabelle pro Knoten

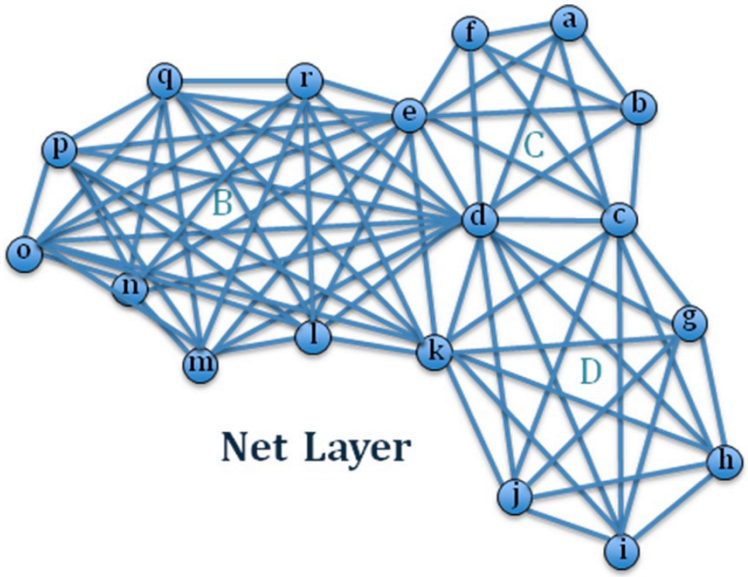
BeeJamA - Layered routing



- Knoten = Grenzknoten eines Bereichs der Bereichsebene
  - Grenzknoten eines Bereichs sind vollständig verbunden
- Kanten = Pfade auf der Bereichsebene
- Keine zusätzliche Hardware
- BeeHive als Routingverfahren

BeeJamA – Layered routing

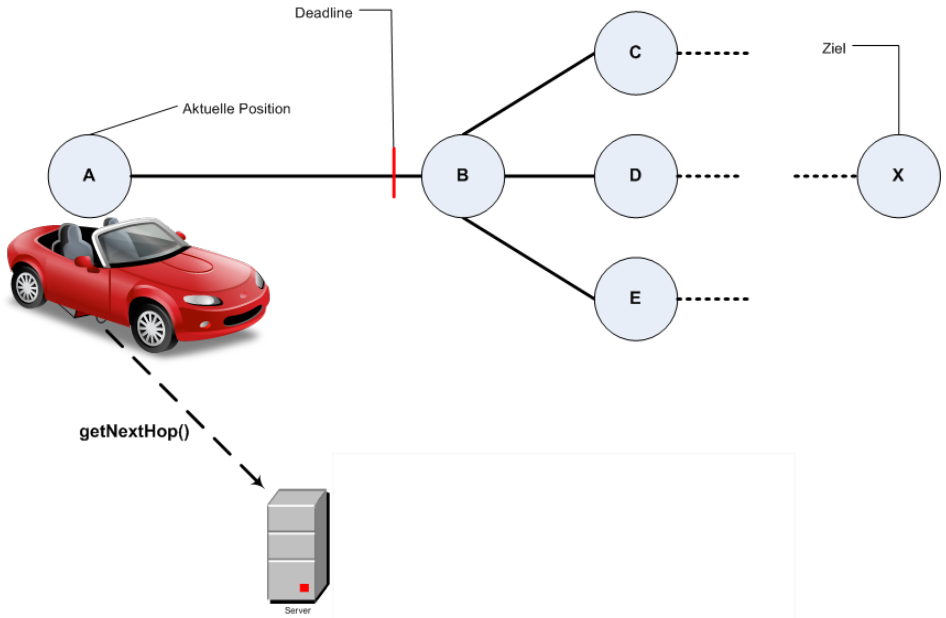




- Foraging Regions werden durch Representative Nodes vertreten
- Routing in die “grobe” Richtung

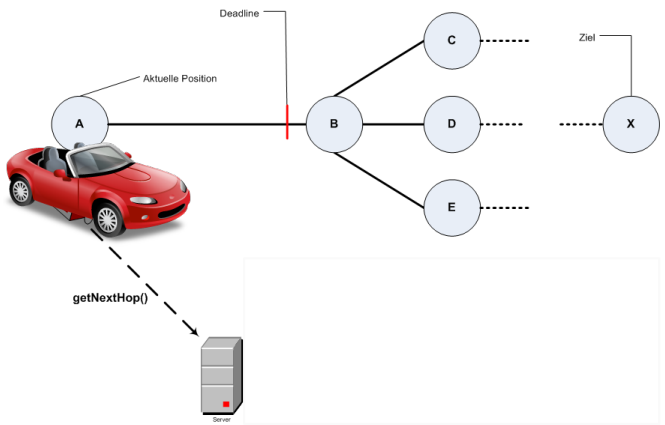
Dynamische Deadlines: BeeJamA Scheduler

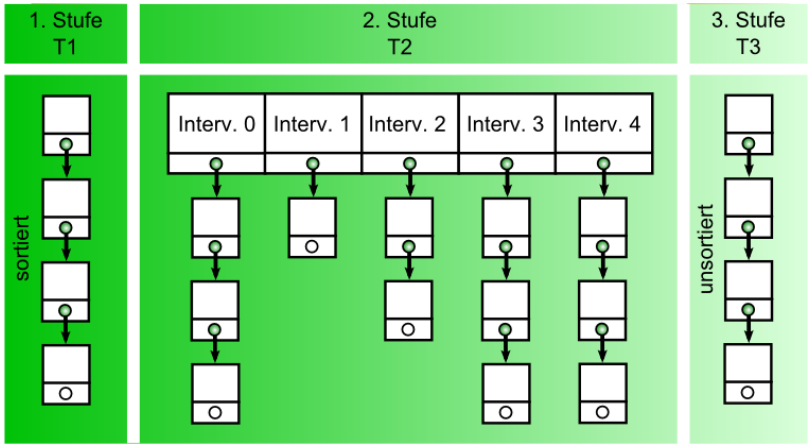
- Routinganweisungen müssen rechtzeitig kommen
  - (Aber: so spät wie möglich)
  - Weiche Echtzeitbedingung: Bei Miss wird Kürzester-Weg genommen



Dynamische Deadlines: BeeJamA Scheduler

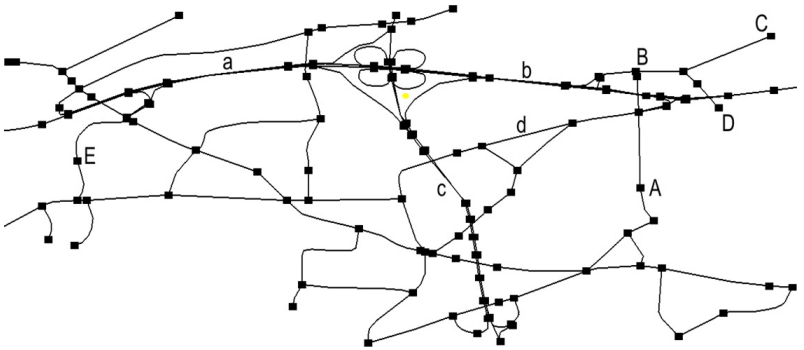
- Routinganweisungen müssen rechtzeitig kommen
  - (Aber: so spät wie möglich)
  - Weiche Echtzeitbedingung: Bei Miss wird Kürzester-Weg genommen
- Nichtpreemptiven, dynamischen (in Bezug auf Deadlines) Scheduler auf Basis von EDF für hohe Task-Anzahl und vielen Deadlineänderungen
- Experimente:
  - Auf Listen-basierende EDF Scheduler langsam
  - Liste durch M-List (3-stufige Priority Queue) ersetzt
    - Min. um Faktor 30-50 schneller

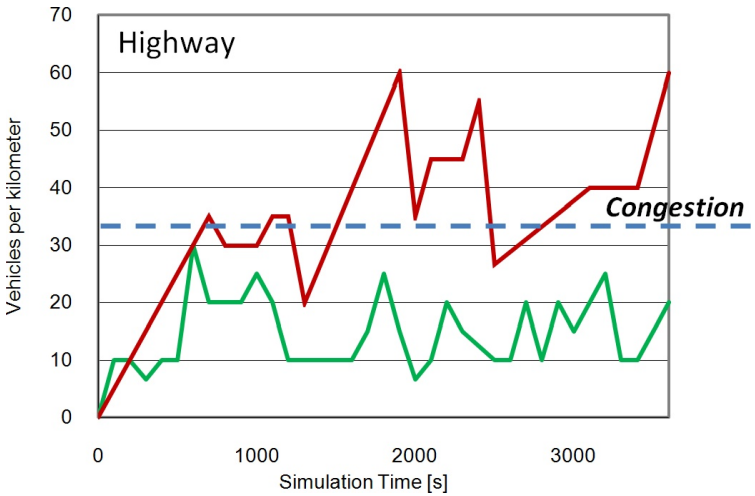


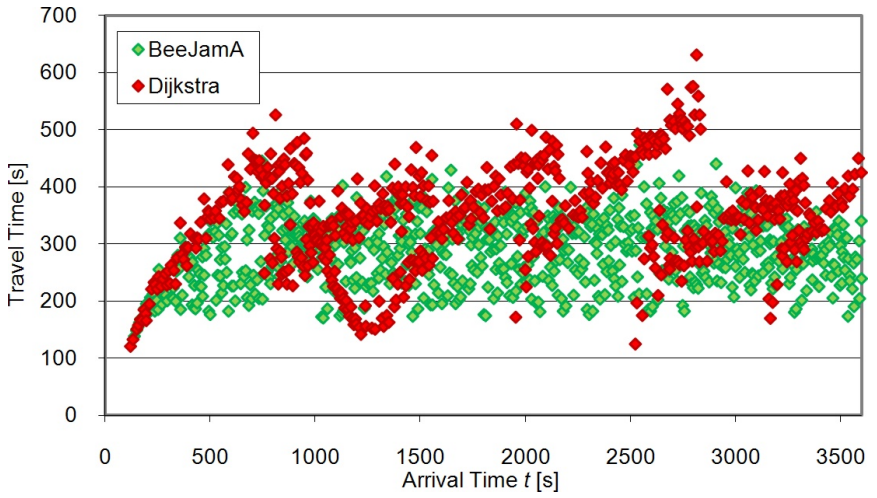


Erste Simulationsergebnisse

- Verkehrssimulator
  - Zellularautomat (Nagel/Schreckenberg)
  - Ausschnitts des Ruhrgebiets





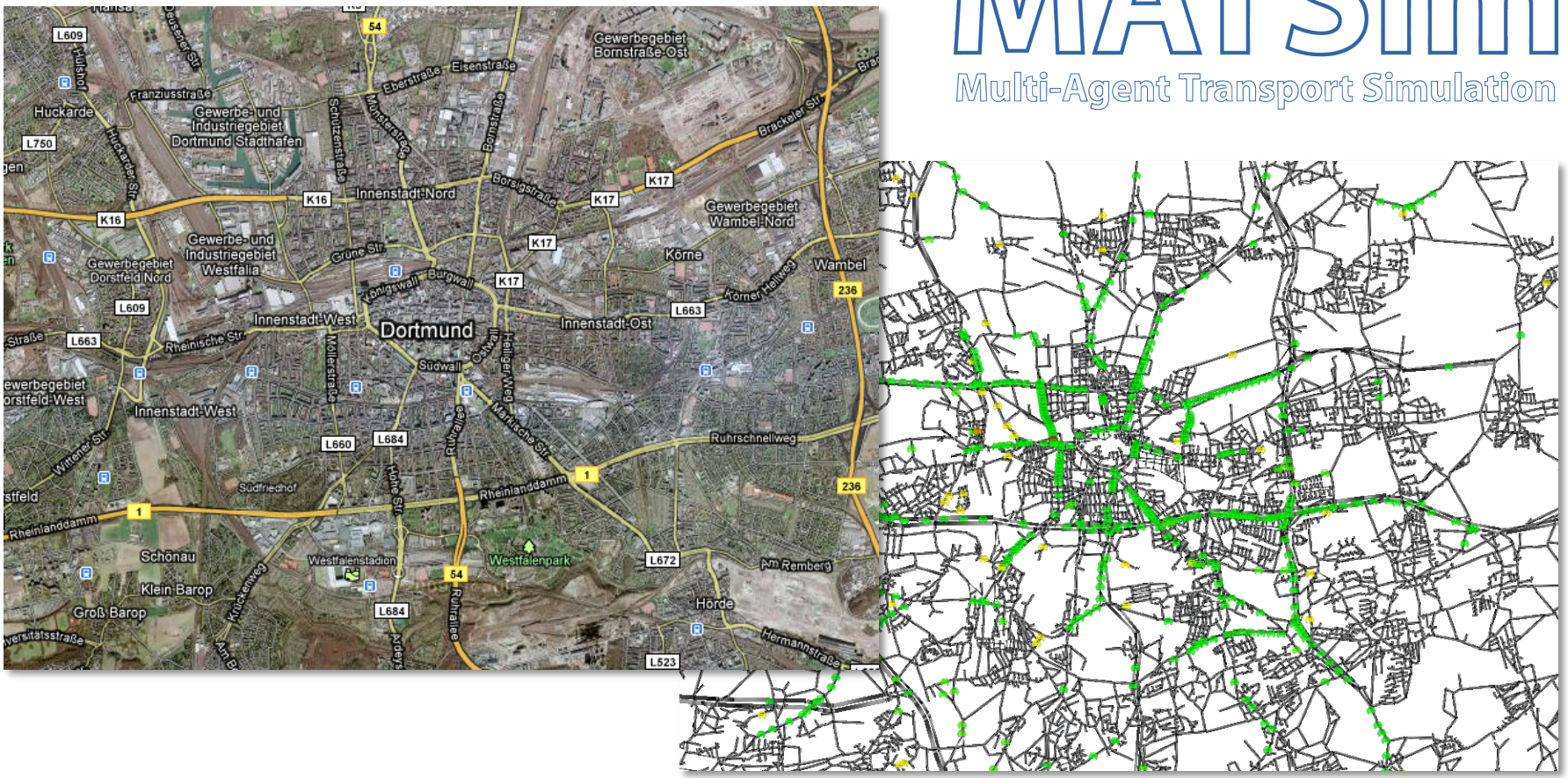


Ausblick

1. Umfangreiche Simulationsstudien für MATSim
   - Ruhrgebiet
   - Durchdringungen
   - Dimensionierung von Hardware und Aufwandsabschätzung
2. Bereitstellung eines Generischen Routing Frameworks (GRF) für MATSim
3. Simulation verschiedener Kommunikationsmodell (z.B. mit OmNet++)
4. Hybridexperimente (Simulation und reale Navigation)
5. Lazy Scheduling

MATSim

Multi-Agent Transport Simulation



Fazit

- BeeJamA
  - Verteiltes Verkehrsmanagement
  - Vehicle-2-Infrastructure Architektur
  - Floating Car Data zur Kantenbewertung
  - MAS zum Propagieren der Bewertungen
  - Probabilistisches Hop-to-Hop-Routing

→ Keine globalen Informationen nötig

→ Vorherige Resultate: Reisezeitverkürzung durch Stauvermeidung

- Echtzeitherausforderung: Dynamische Deadlines

Fragen?

Vielen Dank für Ihre Aufmerksamkeit!