Shortest Path Algorithms: Traditional vs. Innovative Approach

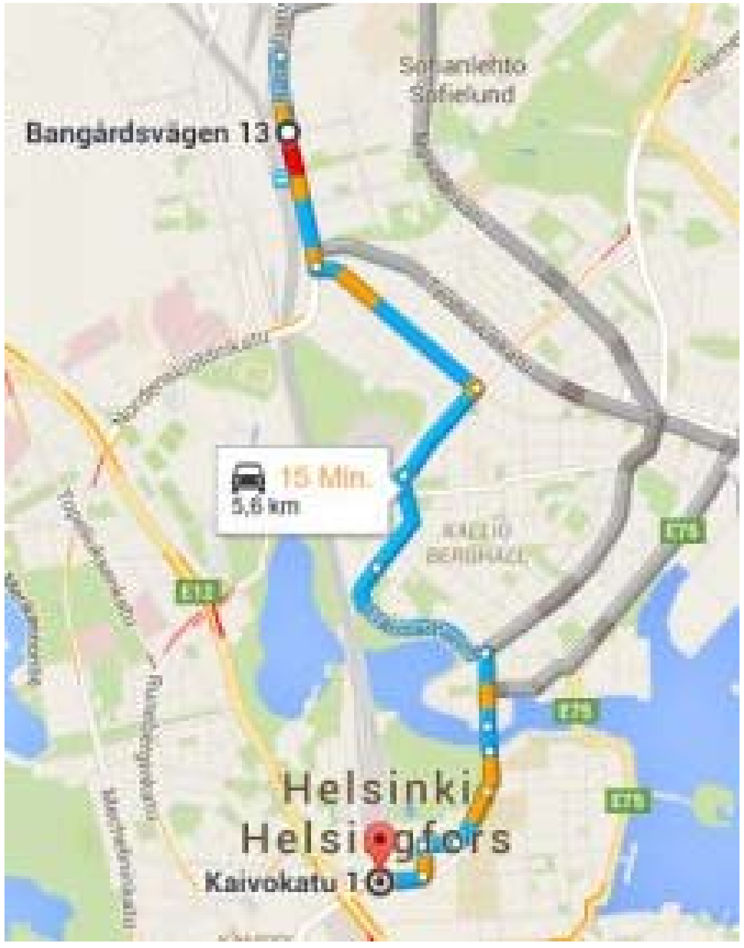
Prof. Dr. Sebastian Iwanowski

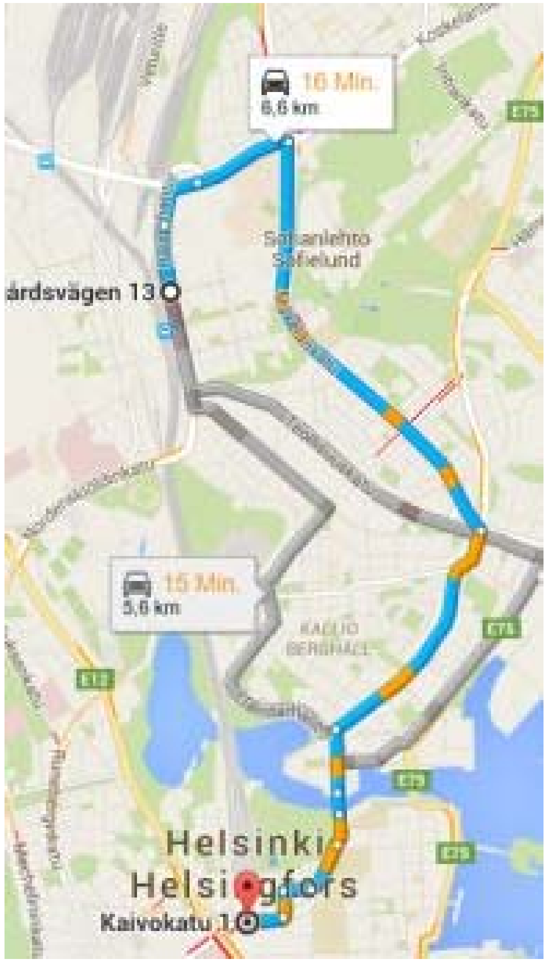
FH Wedel, University of Applied Systems, Germany
Guest Lecturer at Haaga-Helia Ammattikorkeakoulu
Class Innovation Topics, Wednesday
16. September 2015

Shortest Path Problem

Navigation Problem:

Given a map with nodes and connecting edges, each edge beiing assigned a number (for travel time/cost/etc.):
For a certain source and destination, find the shortest path.

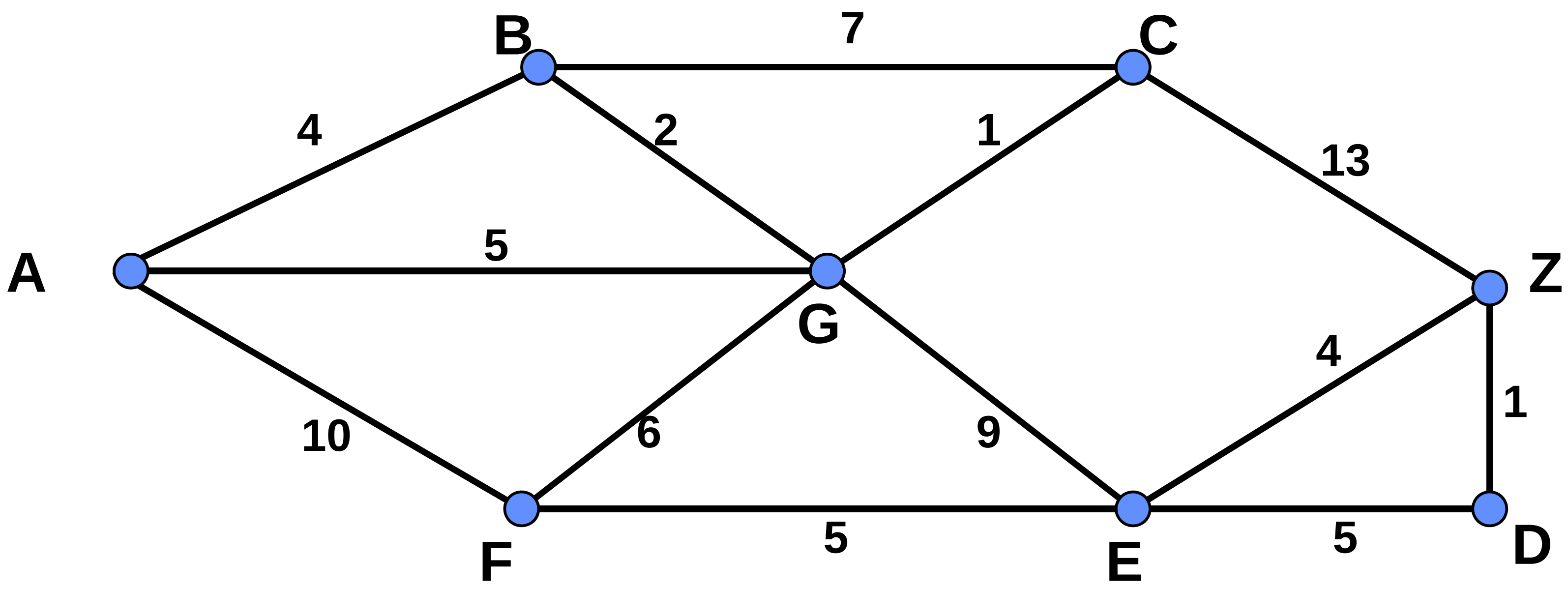




Shortest Path Problem

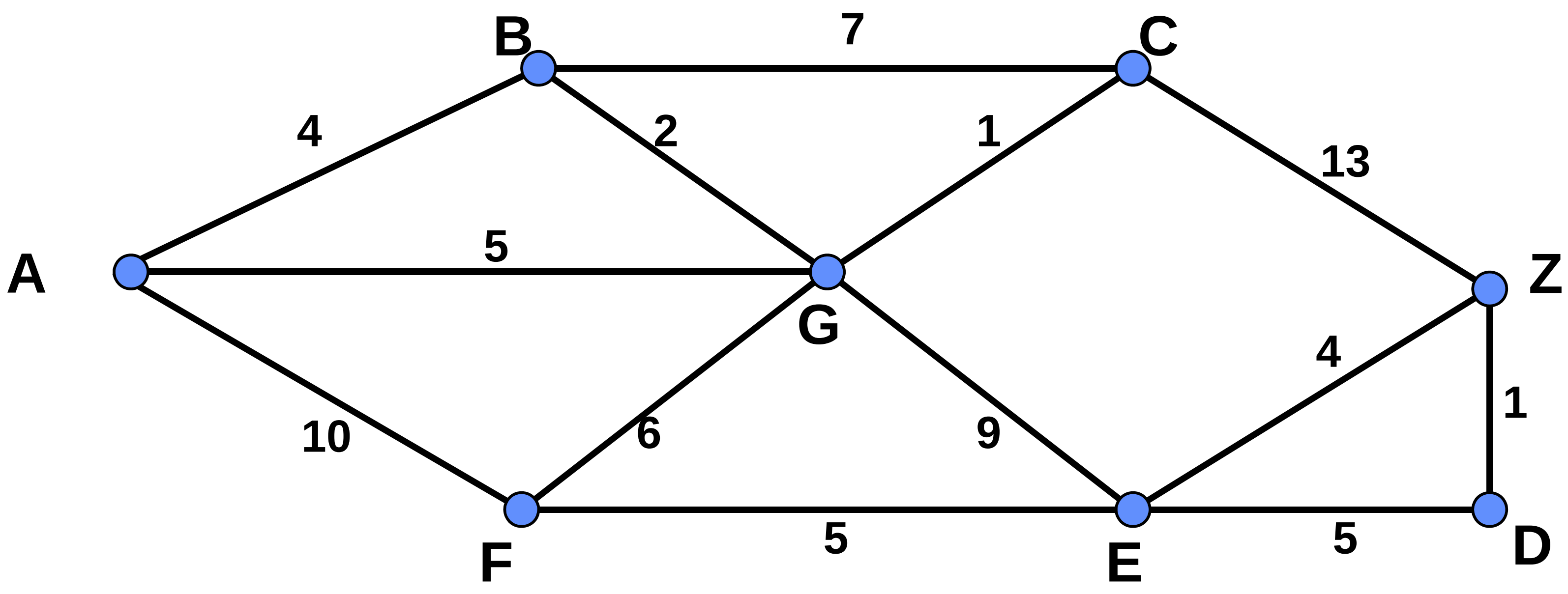
Abstract Graph Problem:

A graph (V,E) is a construct made of *vertices* and *edges*:
An edge always connects two vertices. These vertices are the *endpoints* of the edge.



Find the shortest path from A to Z.

Example for Dijkstra's algorithm



Shortest path from A to Z:

Node (distance from G, direct predecessor):

Result: A → G → E → Z (18 units)

B(4,A)										
C(∞)		C(11,B)		C(6,G)						
D(∞)		D(∞)		D(∞)		D(∞)		D(∞)		D(19,E)
E(∞)	→	E(∞)	→	E(14,G)	→	E(14,G)	→	E(14,G)	→	
F(10,A)		F(10,A)		F(10,A)		F(10,A)				
G(5,A)		G(5,A)								
Z(∞)		Z(∞)		Z(∞)		Z(19,Z)		Z(19,Z)		Z(18,E)

Pseudocode for Dijkstra‘s algorithm

<u>Dijkstra‘s algorithm for weighted graphs</u>

(special case of best first search)

For all edges (u,v) there is a weight function:
length (u,v) := length of an edge from node u to node v

Requirement for edge weights: All lengths have to be nonnegative.

Algorithm for the search of a path from A to B having minimal total edge length:

- Put A into the set **`Done`**. Label A by *distance*(A) := 0.
  Put all other nodes into the set **`YetToCompute`**.
  Label all neighbors N of A by *distance* (N) := *length* (A,N)
  and all othe nodes by *distance* (V) := ∞.
- Repeat:
  Choose node V from **`YetToCompute`** with minimum *distance* (V)
  and shift V to the set **`Done`**.
  Update all neighbors N of V that are still in **`YetToCompute`**:
  distance (N) := min {*distance* (N), *distance* (V) + *length* (V,N)}.
  until V = B

Dynamic Routing

Navigation considering the current quality of road segments

Prerequesites for the system:

- Continuous provision of latest infos about any road segment

not a topic of this work!

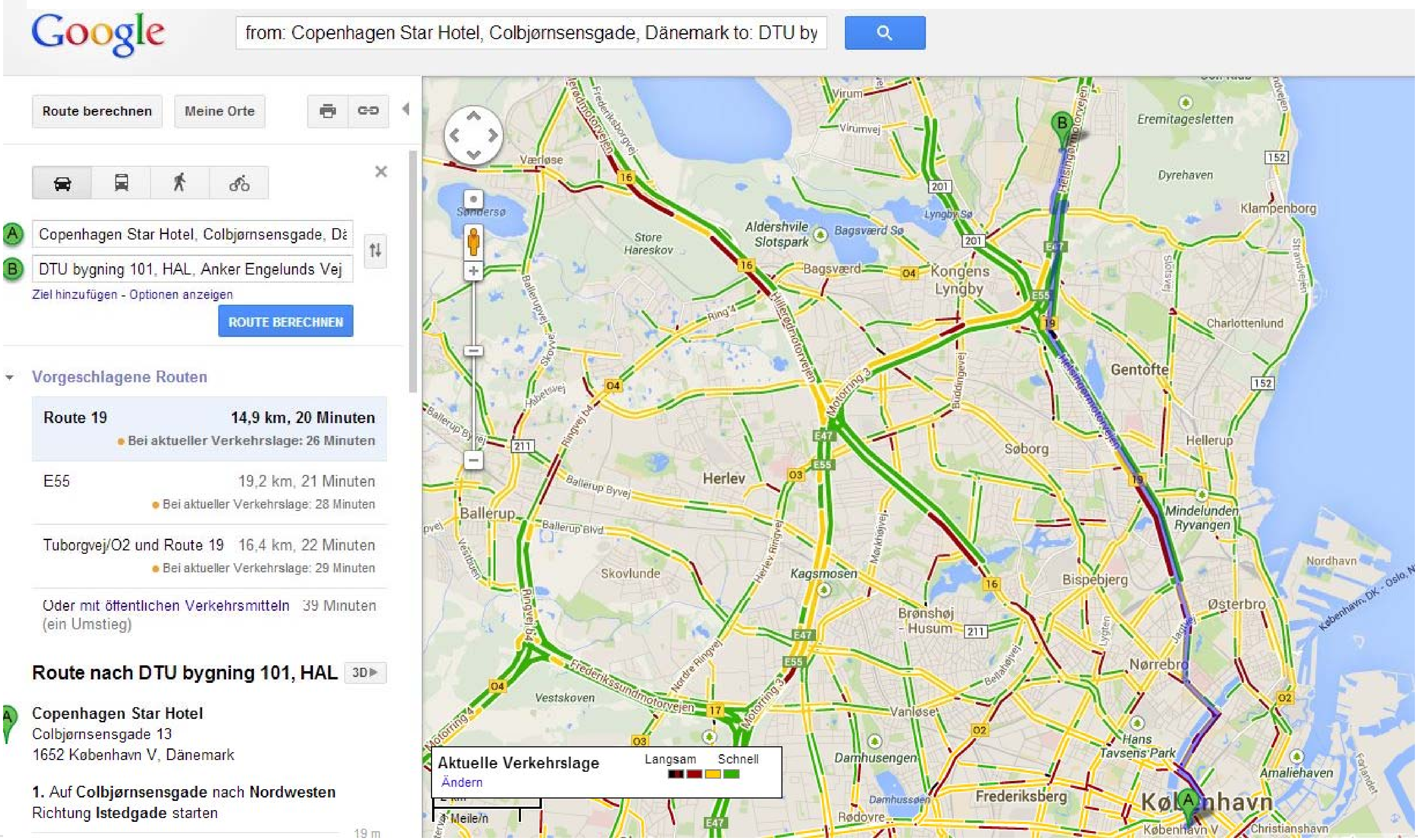
Scenario for future navigation systems:

- All info is available for all road segments at any time.
- Individual request asks for the best road from the present position to a chosen destination **considering all infos at the time of query**.

This makes an on-board computation of the route unfeasible!

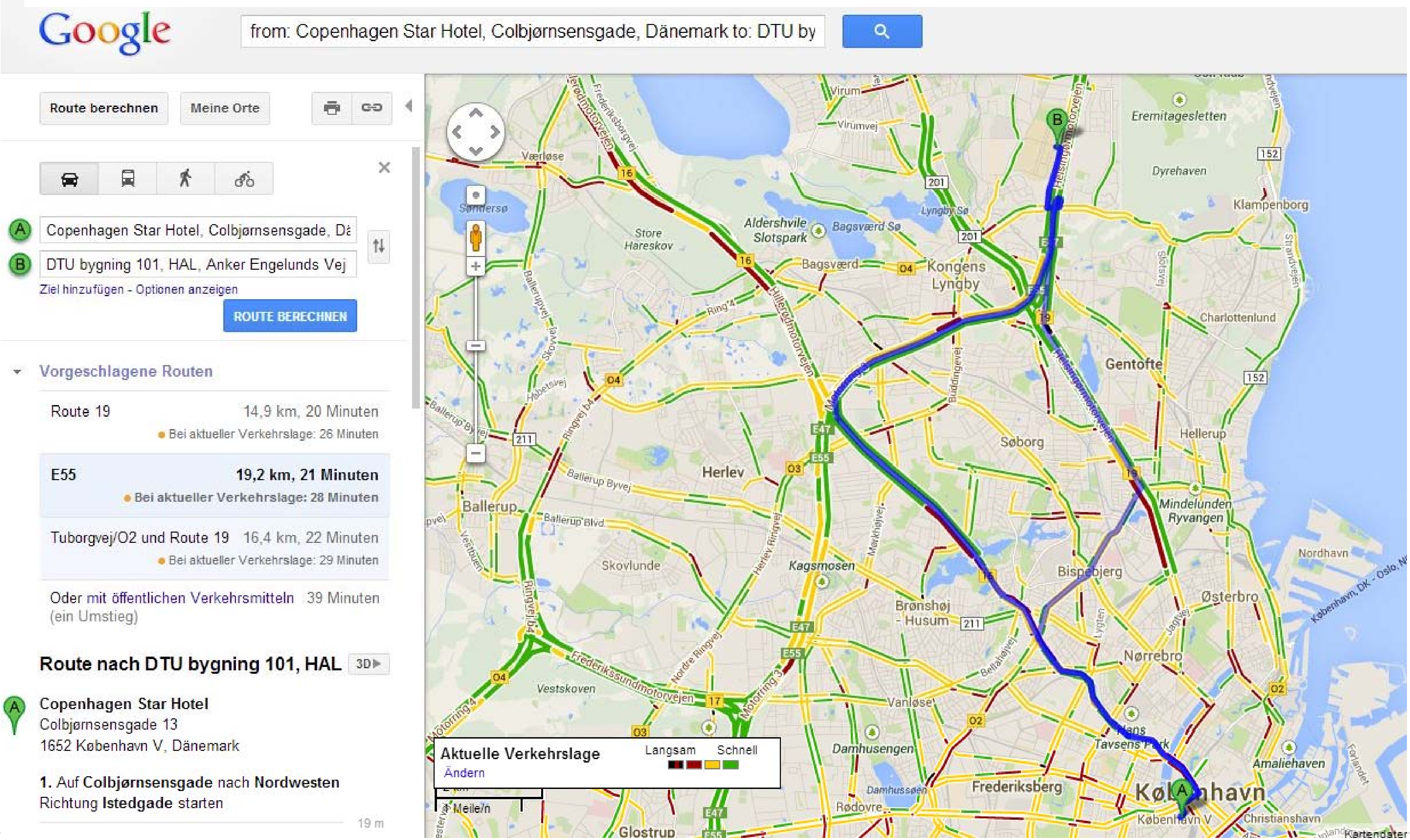
Dynamic Routing

Navigation considering the current quality of road segments



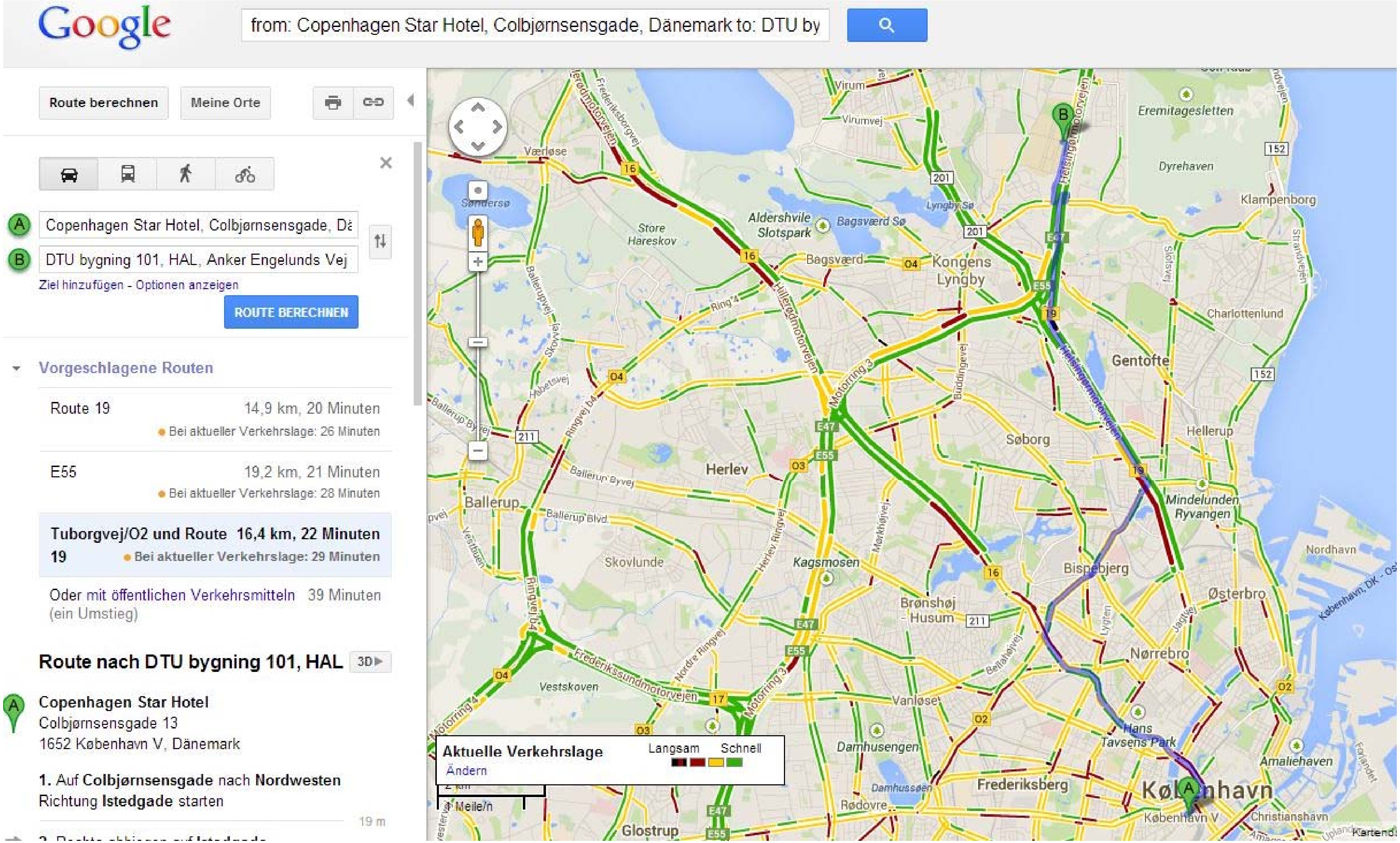
Dynamic Routing

Navigation considering the current quality of road segments



Dynamic Routing

Navigation considering the current quality of road segments



Dynamic Routing

Navigation considering the current quality of road segments

GoogleMaps as a state-of-the-art provider using current infos:

- Google does off-board computation
- Google gives you the three routes with the expected driving time considering the current situation at time of query

Our problem:

- We do not know how Google computes the best routes so fast.

This would already be a motivation to investigate how to do this on our own

But tests show:

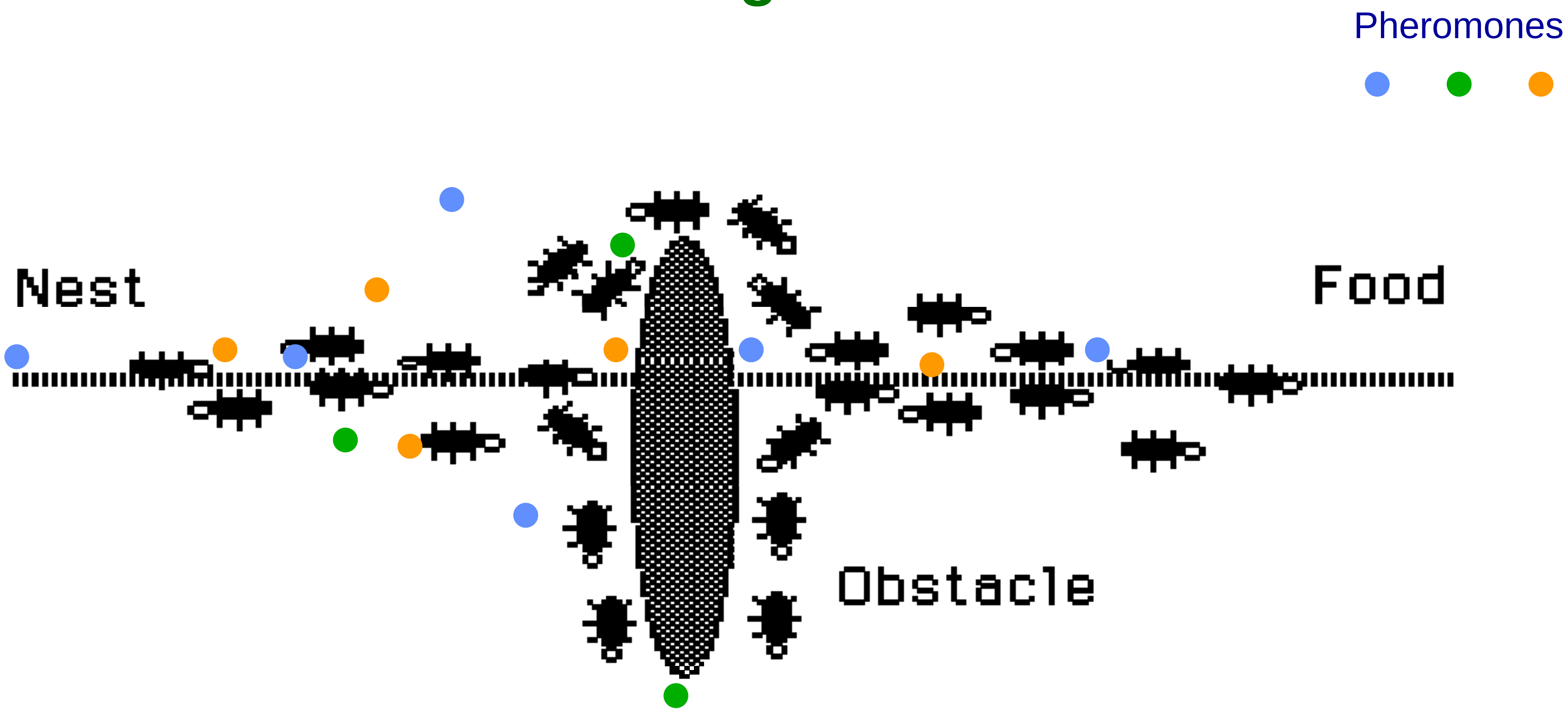
- Google does **not** give you **the best** routes considering the **current situation**
- Google rather computes the best routes for the normal situation and adapts the time forecast for these routes considering the best situation.

Open problem:

- How to compute the best routes considering the current situation?

How ant colonies solve dynamic routing

Ants searching for food



How ant colonies solve dynamic routing

Principal concept (nature and simulation)

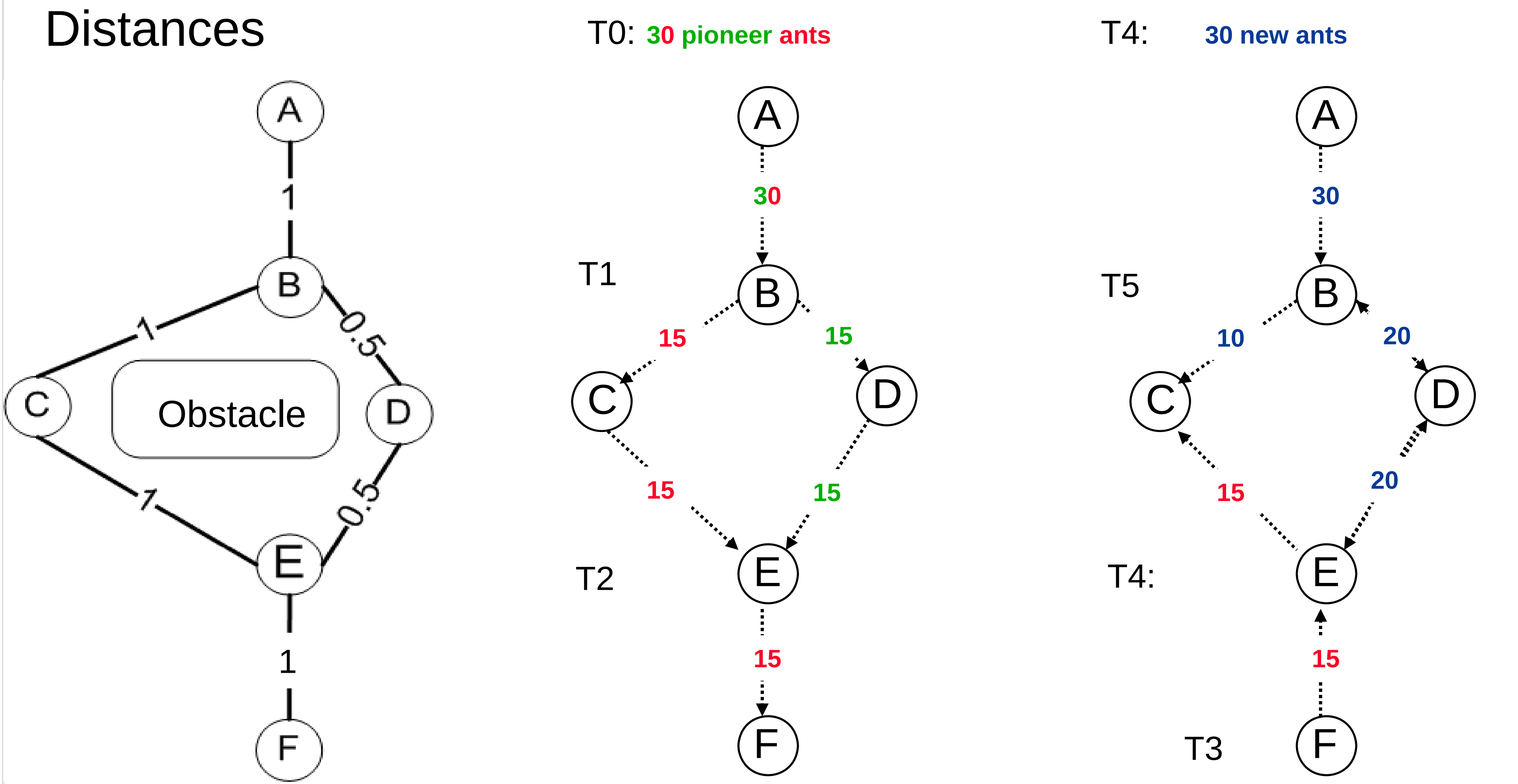
- Each ant sets pheromones continuously walking on its path.
- At junctions, the probability that an ant decides for a certain direction is proportional to the pheromone concentration towards this direction.
- It makes a difference if an ant is on the search for food or on its return path:

a) Each ant returns the same path back as it came there (as soon as it found food). ***Simulation***

b) For either direction different pheromone types are used. ***Nature***

How ant colonies solve dynamic routing

Advantages of probabilistic decision making: Example (alt. a)



Artificial Ant Systems

How do we simulate ant behaviour for the routing problem?

Different pheromones for different destinations

- Each node has got a routing table
- This looks exactly like routing tables in a computer network

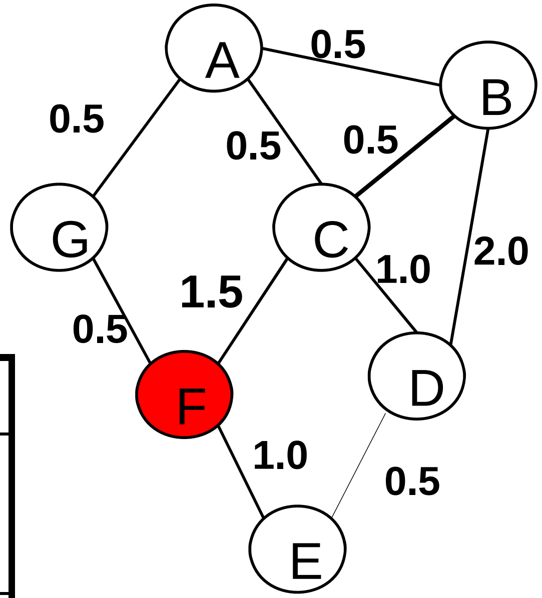


table F			
Dest \ Next	C	G	E
A	0.3	0.65	0.05
B	0.5	0.35	0.15
C	0.9	0.05	0.05
D	0.9	0.05	0.05
E	0.05	0.05	0.9
G	0.6	0.35	0.05

table C				
Dest \ Next	A	B	D	F
A	0.7	0.1	0.1	0.1
B	0.05	0.85	0.05	0.05
D	0.05	0.05	0.85	0.05
E	0.25	0.05	0.05	0.65
F	0.15	0.05	0.05	0.75
G	0.6	0.05	0.05	0.3

This need not necessarily correspond to the current traffic situation !

Artificial Ant Systems

Algorithmic processing

Alternating phases:
Construction of a route and update of pheromone values

Continuously, ants are generated from each source to each destination

Tasks of an ant running from ist source to its destination (forward ant phase):

- At each intersection, choose next edge probabilistically (according to current table entries)
- Collect and store the encountered information (edge lengths, etc.)
- Start the individual pheromone update phase for this ant when destination is reached

Tasks of the pheromone update for a single ant (backward ant phase):

- Trace back the path the corresponding ant used
- Update node infos according to the real-time information the forward ant collected

A simple strategy for pheromone update

$$\Delta P_{s,d} = \frac{c_1}{t_{s,d}} + c_2$$

Evaporation coefficient: This number is used to confirm the path, the ant has really used, and – simultaneously – to deminish the paths, the ant has NOT used.

Evaporation of pheromones for edges not used

$$P_{d,i} = \frac{P_{d,i}}{1 + \Delta P_{s,d}} \forall i \neq f$$

Confirmation of pheromones for edges used

$$P_{d,f} = \frac{P_{d,f} + \Delta P_{s,d}}{1 + \Delta P_{s,d}}$$

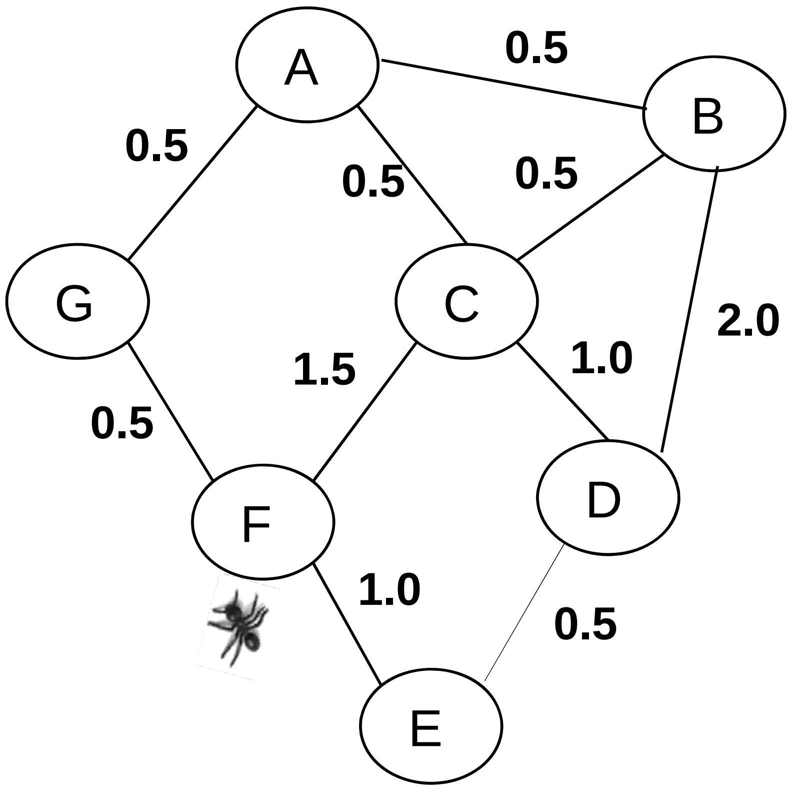
s ... source of ant

d … destination of ant

F … node which was next for ant in order to reach destination

Simple example for pheromone update

Constructing the route (forward ant phase)



memory

s = F d = B

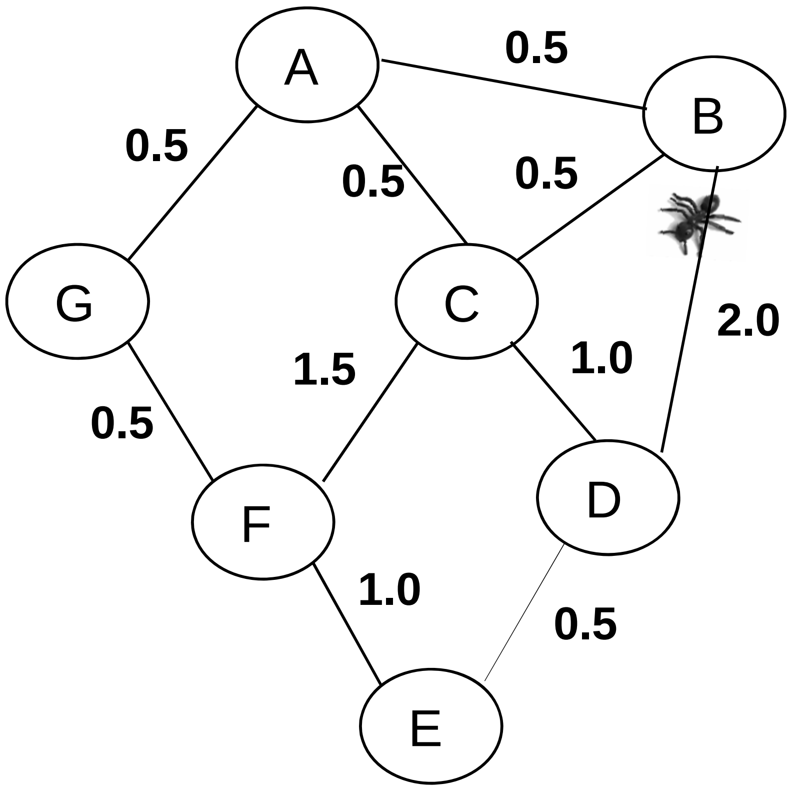
$t_{F,C}=1.5$ $t_{C,B}=0.5$

Table for F (extract)			
Dest \ Next	C	G	E
B	0.5	0.35	0.15

Table for C (extract)				
Dest \ Next	A	B	D	F
B	0.05	0.85	0.05	0.05

Simple example for pheromone update

Updating the pheromones (backward ant phase):



memory

s = F d = B

$t_{F,C}$=1.5 $t_{C,B}$=0.5

Choice for evaporation formula: ***c1=2, c2=1***

$t_{C,B} = 0.5$

Old Table for C (extract)				
Dest \ Next	A	B	D	F
B	0.05	0.85	0.05	0.05

New Table for C (extract)				
Dest \ Next	A	B	D	F
B	0.01	0.97	0.01	0.01

$$\Delta P = \frac{2}{0.5} + 1 = 5$$

$$P_{new,A} = \frac{0.05}{1+5} = 0.01$$

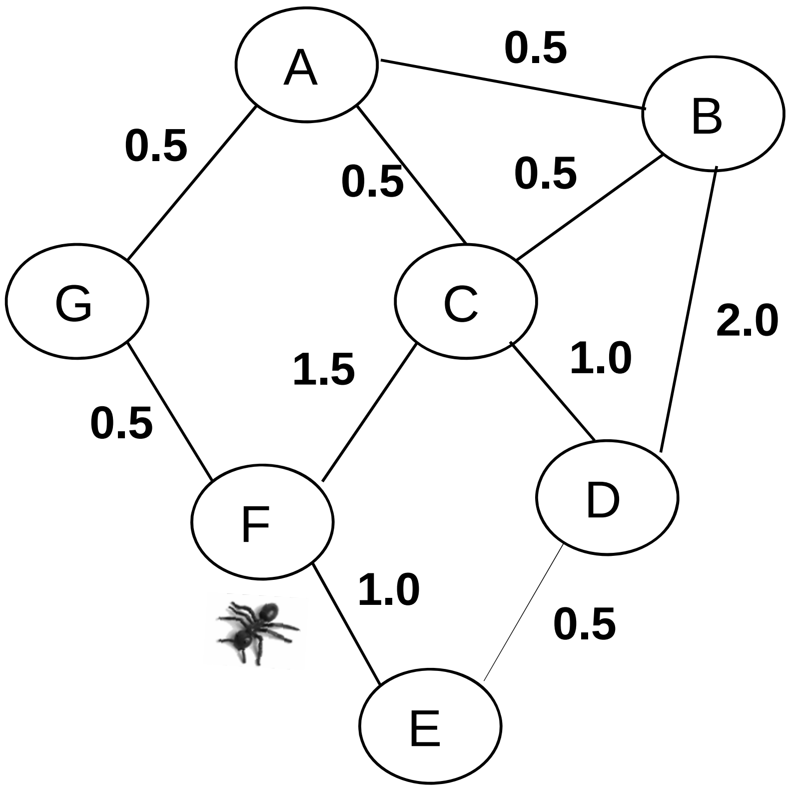
$$P_{new,B} = \frac{0.85+5}{1+5} = 0.97$$

$$P_{new,D} = \frac{0.05}{1+5} = 0.01$$

$$P_{new,F} = \frac{0.05}{1+5} = 0.01$$

Simple example for pheromone update

Updating the pheromones (backward ant phase):



<u>memory</u>

$s = F \quad d = B$

$t_{F,C} = 1.5 \quad t_{C,B} = 0.5$

Choice for evaporation formula: ***c1=2, c2=1***

$t_{F,B} = 2$

Old Table for F (extract)			
Next / Dest	C	G	E
B	0.5	0.35	0.15

New Table for F (extract)			
Next / Dest	C	G	E
B	0.83	0.12	0.05

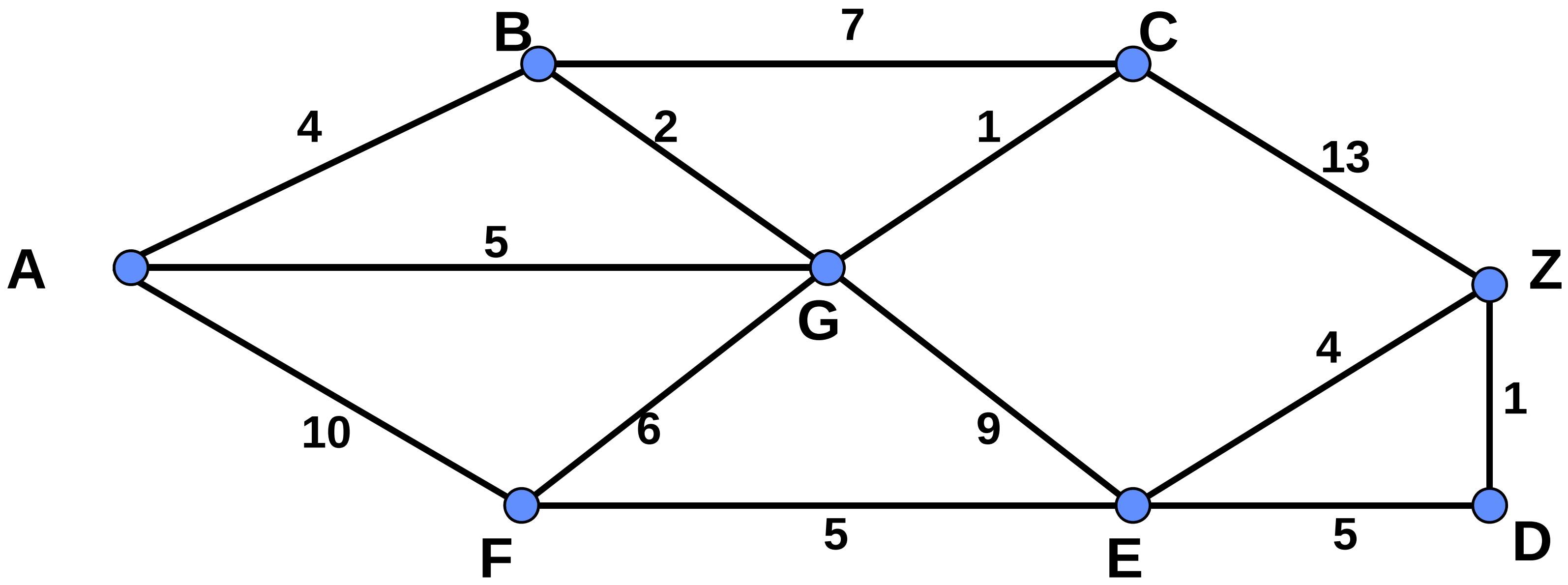
$$\Delta P = \frac{2}{2} + 1$$

$$P_{new,C} = \frac{0.5 + 2}{1 + 2} = 0{,}83$$

$$P_{new,G} = \frac{0.35}{1 + 2} = 0{,}12$$

$$P_{new,E} = \frac{0.15}{1 + 2} = 0{,}05$$

Homework assignment 1: traditional approach



Find the shortest path from G to Z simulating Dijkstra‘s algorithm (cf. slides 4/5)!

Homework assignment 2: innovative approach

Consider the following network and the corresponding pheromone tables:

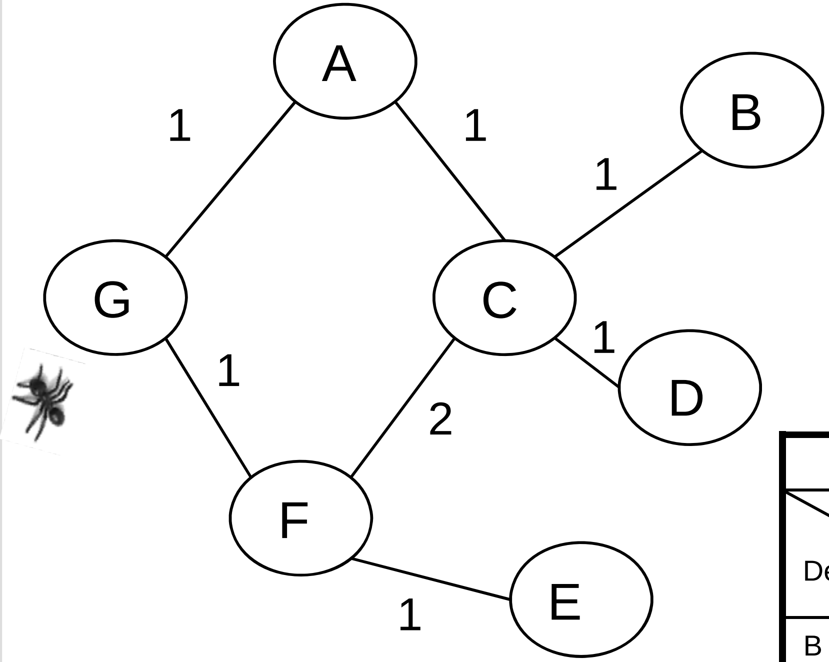


Table for F (extract)

Dest \ Next	C	G	E
B	0.5	0.35	0.15

Table for A (extract)

Dest \ Next	C	G
B	0.9	0.1

Table for C (extract)

Dest \ Next	A	B	D	F
B	0.05	0.85	0.05	0.05

Table for G (extract)

Dest \ Next	A	F
B	0.5	0.5

i) Generate an ant in G with destination B and let it run via A and C to B. Update the pheromone tables according to the presented method with the constants c1=c2=1, but without the restriction that at least 0,05 must remain as remainder probability.

ii) Generate now a second ant in G with destination B and let it run via F and C. Update the pheromone tables according to the presented method as in i).

Exchange the order of i) and ii): What do you observe?