Faculty of Transport and Traffic Sciences, Institute of Transport and Economics, Chair of Transport Services and Logistics

How To Setup Routes in B-u-S-Sim ?

b-u-s-sim.de // Prof. Dr. Jörn Schönberger

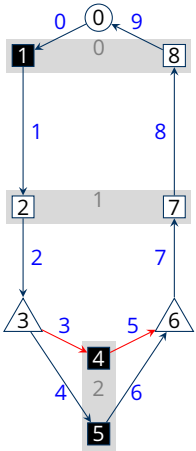
Agenda

Routes in B-u-S-Sim
- What is a Route?
- What is Needed to Define a Route in B-u-S-Sim ?
- The ***BUSSIM_LINE***-Object
- Required C++ - Code Extensions / Modifications

Rotations - Sending a Vehicle Along a Route
- Specification of a Vehicle
- Construction of a Vehicle Rotation
- Scheduling the Start of a Vehicle Rotation

Incorporating a Yard in Route Specification
- Modeling the Depot with ***BUSSIM_POINT***- & ***BUSSIM_ARC***-Instances
- Deployment- & Parking-Routes

What is a Route?



- A route describes a path through the B-u-S-Sim -infrastructure graph
- It starts at a node i^{START} and terminates at a node i^{DEST}
- Both nodes are ***BUSSIM_POINT*** instances and must be of the type `switch` or `trackpos`!
- In B-u-S-Sim , we represent a route as a finite sequence of adjacent ***BUSSIM_ARC*** instances
- Preparation: to have at least one required terminal or start node in each ***BUSSIM_STOP***-instance
  - assign node 0 to the ***BUSSIM_STOP***-instance with ID=0
  - assign nodes 4 & 5 the ***BUSSIM_STOP***-instance with ID=2

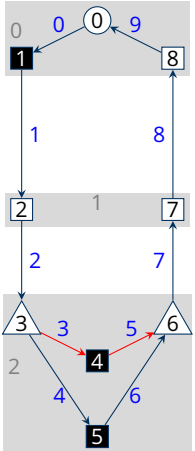


Figure: Example graph

Figure: modified graph

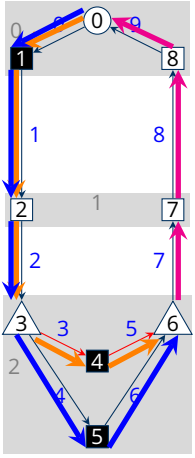
What is Needed to Define a Route in B-u-S-Sim ?

- One route represents a one-directional trip from A to B
  - can be used to represent circle routes
  - modeling a complete round trip with a second route going from B to A
  - The backward route (B→A) may differ from a pure reversal of the forward route (A→B)

- Properties that define a route
  - a unique identification number
  - a key value that allows to group several routes (e.g. Linie 13 route from Prohlis to Kaditz with Linie 13 route from Kaditz to Prohlis)
  - A line name (verbal description, e.g. “Linie 13”)
  - A color for displaying the route in the simulation window (given in RGB color scheme)

- A sequence of ***BUSSIM_ARC*** instances defining the travel path

Routes in the Example Infrastructure



- Route 0: from stop 0 to stop 2 (platform 4)
  (0;1), (1;2), (2;3), (3;4), (4;5) ⇒ arc sequence 0, 1, 2, 3, 5
- Route 1: from stop 2 (platform 4) to stop 0
  (6;7), (7;8), (8;0) ⇒ arc sequence 7, 8, 9
- Route 2: from stop 0 to stop 2 (platform 5)
  (0;1), (1;2), (2;3), (3;5), (5;5) ⇒ arc sequence 0, 1, 2, 4, 6

Figure: modified graph

The *BUSSIM_LINE*-Object

attribut	type	description	orange	magenta	blue
ID	int	unique key to identify a route	0	1	2
SERVICE	int	value to group several ***BUSSIM_LINE*** instances	13	13	13
RED	double	degree of red color according RGB-model (all values between 0 and 1 are allowed)	1.0	1.0	0.0
GREEN	double	degree of green color according RGB-model (all values between 0 and 1 are allowed)	0.55	0.0	0.0
BLUE	double	degree of blue color according RGB-model (all values between 0 and 1 are allowed)	0.0	1.0	1.0
LineName	char[256]	string containing a short verbal description of this instance	13:->2(p4)	13:->0	13:->2(p5)

Table: attributes to be set for a `BUSSIM_LINE` instance (with 3 examples)

Required C++ - Code Extensions / Modifications

Adjustment of the 5th parameter of the ***BUSSIM_NETWORK*** - constructor in `main.cpp`

```
class BUSSIM_NETWORK                NET(3,9,10,1, 3 ,0,0,0);
```

BUSSIM_LINE-instantiation

- `BUSSIM_LINE::configure(int _ID, int _SERVICE, double _RED, double _GREEN, double _BLUE, const char _LINENAME[256])`
- one call for each instance to be placed in `BUSSIM_NETWORK::specify_lines(void)`

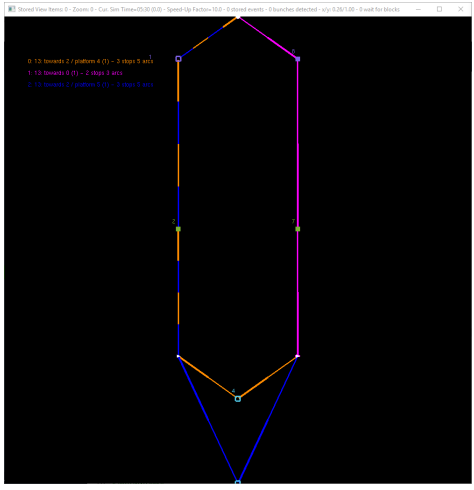
BUSSIM_LINE-route construction

- `BUSSIM_LINE::append_arc(class BUSSIM_ARC _ARC)`
- to be placed in `BUSSIM_NETWORK::specify_lines(void)`
- after the call of the `BUSSIM_LINE::configure(...)` instruction of an instance,

Required C++ - Code-Snippets for Route Definitions

```
1  void BUSSIM_NETWORK::specify_lines(void)
2  {
3      this->LINE[0].configure(0,13,1.0,0.55,0.0,"13:->2(p4)");
4      this->LINE[0].append_arc(this->ARC[0]);
5      this->LINE[0].append_arc(this->ARC[1]);
6      this->LINE[0].append_arc(this->ARC[2]);
7      this->LINE[0].append_arc(this->ARC[3]);
8      this->LINE[0].append_arc(this->ARC[5]);
9      this->LINE[1].configure(1,13,1.0,0.0,1.0,"13:->0");
10     this->LINE[1].append_arc(this->ARC[7]);
11     this->LINE[1].append_arc(this->ARC[8]);
12     this->LINE[1].append_arc(this->ARC[9]);
13     this->LINE[2].configure(2,13,0.0,0.5,1.0,"13:->2(p5)");
14     this->LINE[2].append_arc(this->ARC[0]);
15     this->LINE[2].append_arc(this->ARC[1]);
16     this->LINE[2].append_arc(this->ARC[2]);
17     this->LINE[2].append_arc(this->ARC[4]);
18     this->LINE[2].append_arc(this->ARC[6]);
19  }
```

Presentation in B-u-S-Sim -Simulation Window



Specification of a Vehicle

- In B-u-S-Sim vehicles can travel only along previously specified routes
- For each vehicle in the simulation it is necessary to determine in detail when and where it is during the simulation
- B-u-S-Sim uses three objects for storing the required information
  - a ***BUSSIM_VEHICLE***-object instance stores the required vehicle properties incl. its initial position as well as the final vehicle position in the infrastructure graph
  - a ***BUSSIM_ROTATION***-object instance exists uniquely for each stored ***BUSSIM_VEHICLE*** instance and contains the detailed travel path operated by a vehicle throughout the complete simulation
  - a ***BUSSIM_SCHEDULE***-object instance exists uniquely for each stored ***BUSSIM_ROTATION*** instance and contains the detailed times when a ***BUSSIM_ROTATION***-item is started
- Currently, a ***BUSSIM_SCHEDULE***-object instance is automatically derived from a ***BUSSIM_ROTATION***-object instance

The *BUSSIM_VEHICLE*-Object

attribute	type	description / content
ID	int	unique identification key
vehicle_category	int	determines if we have a (BUSSIM_VEHCAT_TRAM) or a bus (BUSSIM_VEHCAT_BUS)
velocity	double	average vehicle speed in km/h
capacity	int	max. amount of allowed passengers to be on board
ON_ARC	BUSSIM_ARC	initial arc (vehicle is position at its beginning)
parc_arc	BUSSIM_ARC	final arc (vehicle is parked at its end)

Table: attributes to be specified for a ***BUSSIM_VEHICLE*** instance

BUSSIM_VEHICLE Instance Installation: C++ - Code-Modifications

Adjustment of the 6th parameter of the ***BUSSIM_NETWORK*** - constructor in `main.cpp`

```
class BUSSIM_NETWORK                NET(3,9,10,1,3, 1 ,0,0);
```

BUSSIM_VEHICLE-instantiation

- using `configure(int _ID, int _VEH_CAT, double _VELOCITY, int _CAPACITY, class BUSSIM_ARC _ON_ARC, class BUSSIM_ARC _PARK_ARC, class BUSSIM_NETWORK *_NET)`
- to be placed in `BUSSIM_NETWORK::specify_vehicles(void)`

Example: places a tram vehicle with `ID=0` at the beginning of the arc with ID=7 and the parking arc has the `ID` 5 (150 passengers, and average speed 60km/h)

```
1  void BUSSIM_NETWORK::specify_vehicles(void)
2  {
3      this->VEHICLE[0].configure(0,BUSSIM_VEHCAT_TRAM,60,150,this->ARC[7],this->ARC[5],this);
4  }
```

A Vehicle Rotation as a Sequence of *BUSSIM_ROUTE* Instances

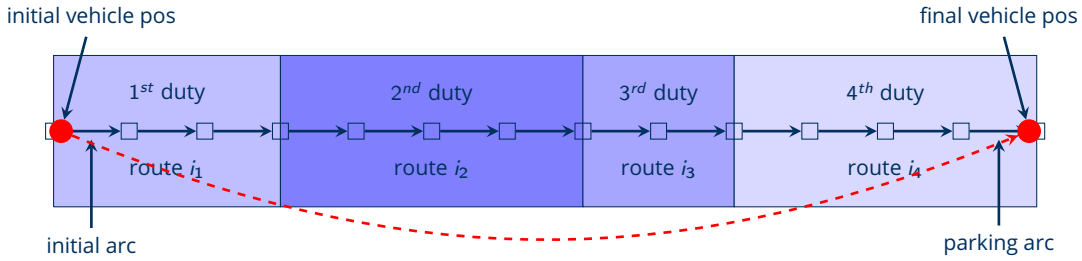


Figure: a vehicle rotation with 4 duties: rotation = (i_1;i_2;i_3;i_4)

- A **rotation** determines the travel path of a vehicle during the simulation
- Defined as a sequence of ***BUSSIM_ROUTE*** instances with adjacent start and end nodes
- The route in the i^{th} position of a rotation is called i^{th} duty of this vehicle
- IMPORTANT: the first arc in the 1st duty must be the initial arc, the final arc in the last duty in a rotation must coincide with the parking arc

Rotation Specification - Source Code Extensions (Example 1)

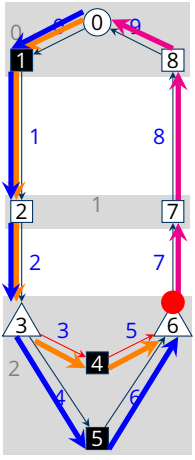


Figure: modified graph

- Initially, the vehicle (●) is placed at the beginning of arc with ID=7
- Example 1: The vehicle should travel one round trip through node 4 ⇒ rotation = (1;0)

Rotation Coding in `BUSSIM_NETWORK::specify_rotations(void)`

```
1  void BUSSIM_NETWORK::specify_rotations(void)
2  {
3      this->append_duty_to_vehicle_rotation(0,1);
4      this->append_duty_to_vehicle_rotation(0,0);
5  }
```

Rotation Specification - Source Code Extensions (Example 2)

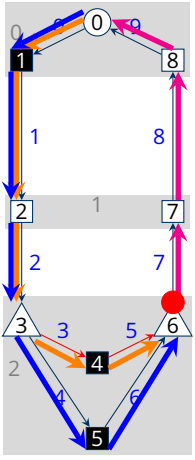


Figure: modified graph

- Initially, the vehicle (●) is placed at the beginning of arc with ID=7
- Example 2: The vehicle should travel a first round trip through node 4 followed by a second round trip through node 5 and a third round trip through node 4 ⇒ rotation = (1;0;1;2;1;0)

Rotation Coding in `BUSSIM_NETWORK::specify_rotations(void)`

```
1  void BUSSIM_NETWORK::specify_rotations(void)
2  {
3      this->append_duty_to_vehicle_rotation(0,1);
4      this->append_duty_to_vehicle_rotation(0,0);
5      this->append_duty_to_vehicle_rotation(0,1);
6      this->append_duty_to_vehicle_rotation(0,2);
7      this->append_duty_to_vehicle_rotation(0,1);
8      this->append_duty_to_vehicle_rotation(0,0);
9  }
```

Scheduling the Start of a Vehicle Rotation

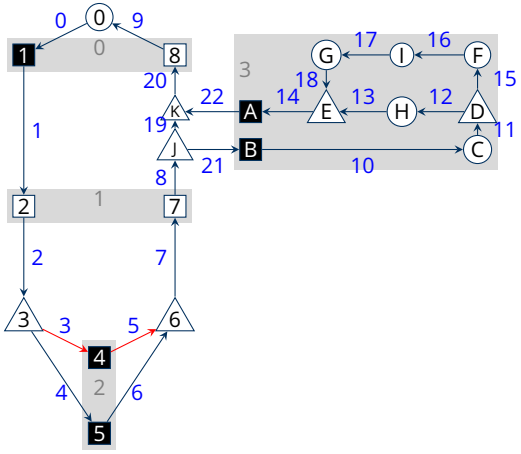
- General scheduling rules in B-u-S-Sim
  - every vehicle rotation requires the specification of the execution starting time
  - at least one vehicle's rotation requires the starting time 0

- use the method `BUSSIM_VEHICLE::set_activation_time(double _activation_time)` to set the rotation starting time

Specification of rotation activation time in `BUSSIM_NETWORK::specify_vehicle_activation_times(void)`

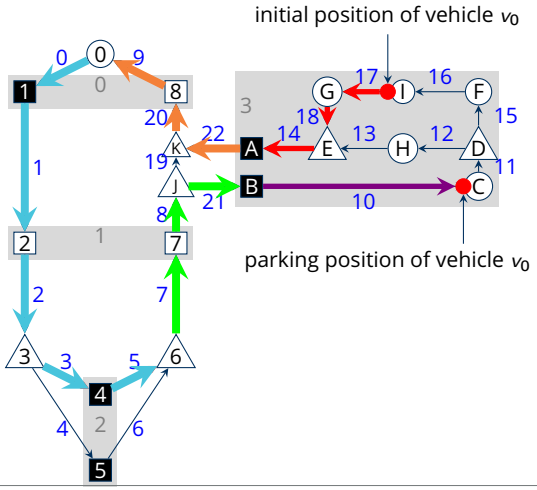
```
1  void BUSSIM_NETWORK::specify_vehicle_activation_times(void)
2  {
3      // set the vehicle activation times
4      this->VEHICLE[0].set_activation_time(0);
5  }
```

Incorporating a Yard in Route Specification - Preparations



- Define at least one entry node (A) and at least one exit node (B) of the yard (both are ***BUSSIM_POINT*** instances of type *regular*)
- Model the yard's infrastructure using ***BUSSIM_POINT*** instances as well as ***BUSSIM_ARC*** instances
- Add an additional ***BUSSIM_STOP*** instance (3)
- Assign all ***BUSSIM_POINT*** instances (A-G) as well as ***BUSSIM_ARC*** instances (10-18) to this ***BUSSIM_STOP*** instance

Incorporating a Yard in Route Specification - Deployment & Parking



- Situation
  - vehicle v_0 waits at the beginning of arc 17 for its deployment
  - it is planned to serve route (0;1;2;3;5)
  - after this service it has to park at the end of arc 10 in the yard
- Routes to be specified
  - the regular route (0;1;2;3;5)
  - the deployment route (22;20;9) from the yard's exit (A) to the beginning of the first regular route
  - the parking route (7;8;21) from the last regular route end to the yard's entry (B)
- B-u-S-Sim automatically adds
  - the first part (17;18;14) on the yard of the deployment route to the yard's exit node (A), and
  - the final part of the parking route (10) from the yard's entry node B to the parking position on the yard

Summary

Routes in B-u-S-Sim
- What is a Route?
- What is Needed to Define a Route in B-u-S-Sim ?
- The ***BUSSIM_LINE***-Object
- Required C++ - Code Extensions / Modifications

Rotations - Sending a Vehicle Along a Route
- Specification of a Vehicle
- Construction of a Vehicle Rotation
- Scheduling the Start of a Vehicle Rotation

Incorporating a Yard in Route Specification
- Modeling the Depot with ***BUSSIM_POINT***- & ***BUSSIM_ARC***-Instances
- Deployment- & Parking-Routes