Simulation laboratory 2: Discrete events simulation

Rico Krueger

Transport and Mobility Laboratory
School of Architecture, Civil and Environmental Engineering
École Polytechnique Fédérale de Lausanne

February 25, 2020

Goals

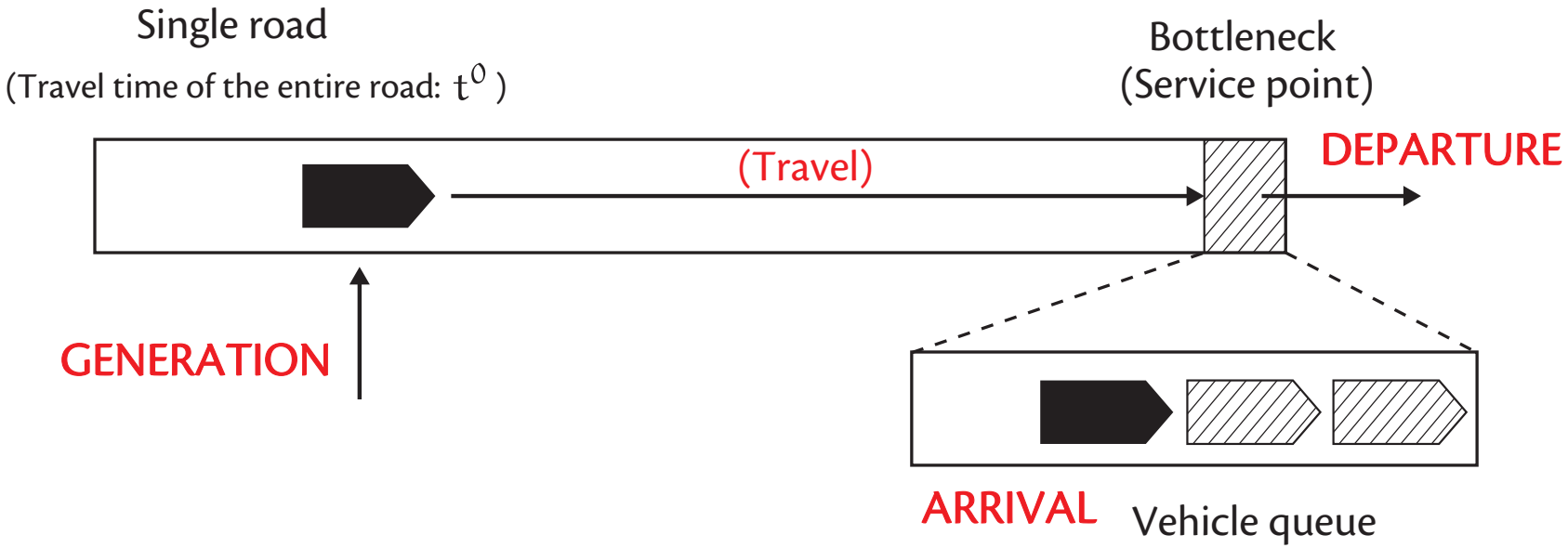
Discrete events simulation:

- Understand how to simulate events
- Apply the Poisson process
- Simulate a time-varying queue

Implementation:

- Simulate vehicle queue on a single-lane road with a bottleneck

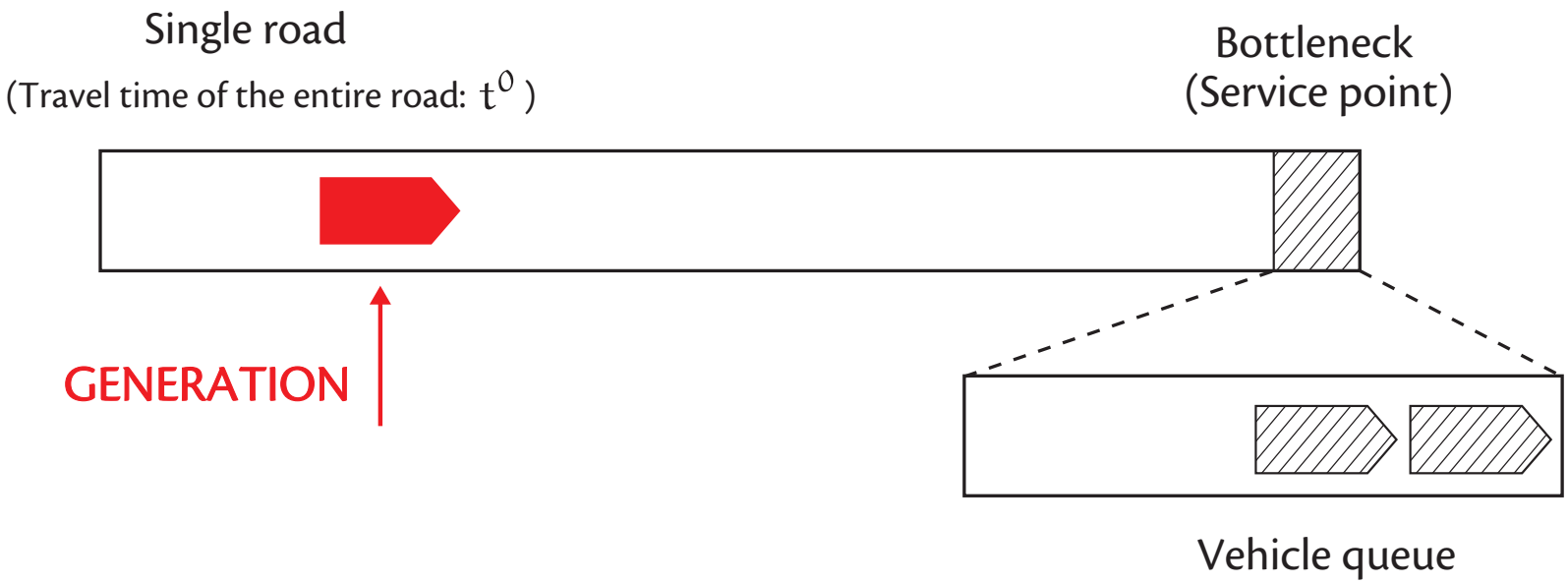
Problem definition: Vehicle queue on a single road



Events:

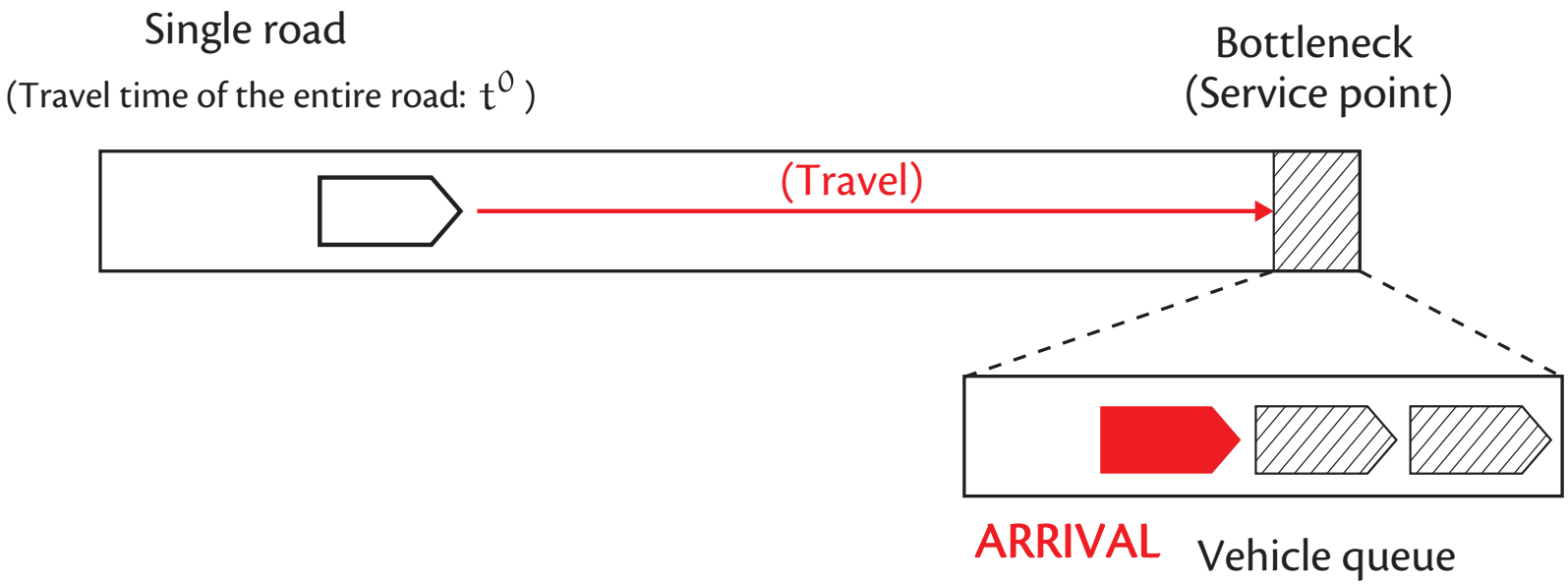
1. Vehicle **GENERATION**
2. Vehicle **ARRIVAL** at the queue
3. Vehicle **DEPARTURE** from the queue

Vehicle **GENERATION**



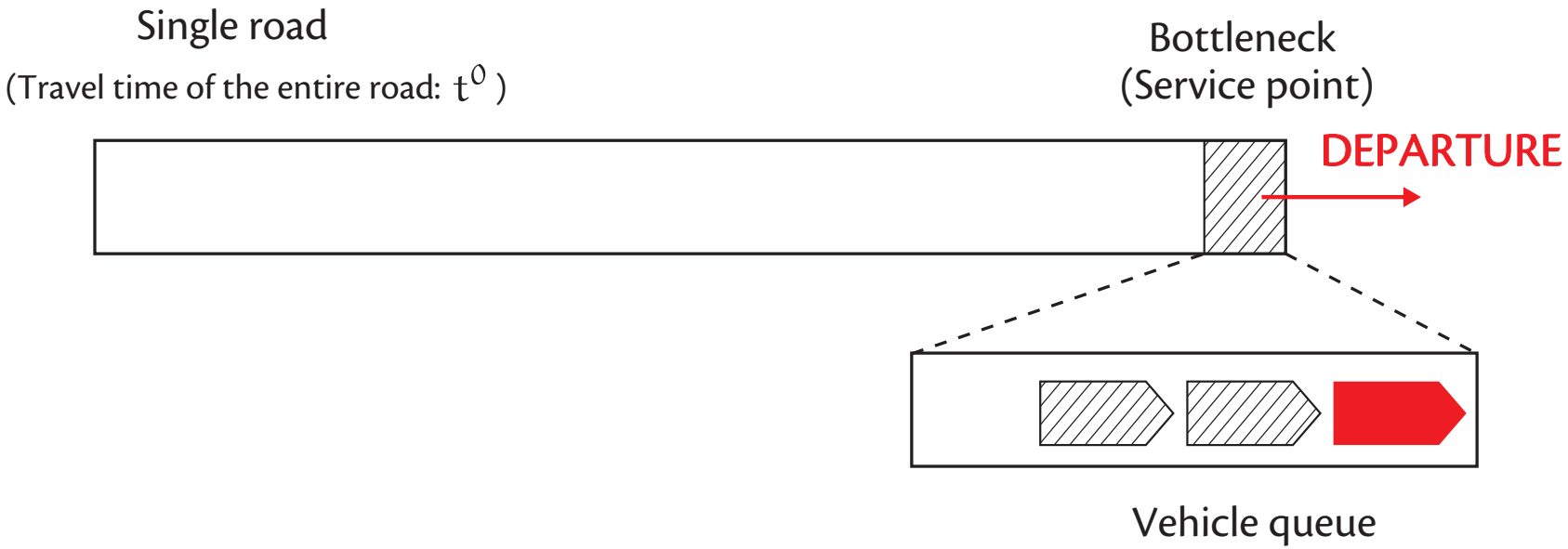
- New vehicle enters at time t^g, following a Poisson process with rate λ.
- Create **GENERATION** event at time $t^g_{i+1} = t^g_i + \mathrm{Exp}(\lambda)$.
- Vehicle entry point is uniformly distributed over the road (imagine vehicles pulling out of their driveways).

Vehicle **ARRIVAL** at queue



- Travel time on the entire road is t^0, thus vehicle arrives downstream of the queue $tt \sim U(0, t^0)$ after entering (see the previous slide).
- Create **ARRIVAL** event at time $t_i^a = t_i^g + tt$.
- Increase queue by 1, i.e., $q := q + 1$

Vehicle **DEPARTURE** from queue



- Service time for a vehicle at bottleneck is $t^s \sim Exp(\mu)$.
- Create **DEPARTURE** event at time $t^d_{i+1} = t^d_i + t^s$.
- Reduce queue by 1, i.e., $q := q - 1$
- Attention at special cases, e.g., only one vehicle in the queue.

Simulation

State variables:

- t: Time
- q: Number of vehicle in the queue

Parameters (scenario):

- T: The duration of vehicle generation
- t^0: Travel time of the entire road
- λ: Rate for vehicle generation
- μ: Rate for service time (road capacity)

Simulation

Events:

- List of future events sorted in chronological order.
- Initialization of the simulation: first event (Generation).
- Process the next event:
  1. Update the variables.
  2. Collect statistics.
  3. Generate and add new events to the list.
  4. Remove the processed event from the list.
  5. Finish the simulation if the list is empty, go to next event otherwise.

Event triggers event

Event	**Triggered event**	**Queue**
Sim. Start	Generation, Sim. End	
Generation	Generation (if $t < T$), Arrival	
Arrival	Departure (if $q = 1$)	$q = q + 1$
Departure	Departure (if $q > 0$)	$q = q - 1$
Sim. End		

Exercise

Jupyter notebooks:

1. **classes_examples.ipynb** illustrates object-oriented programming features of Python.
2. **discrete_events_simulation_blank.ipynb** for implementation of queue simulation.

To do:

- Implement the simulation procedure in function "simulate".
- Collect and analyse simulation statistics.

Hints:

- Use classes to represent different types of events.
- Use the provided Scenario class.

Queue length over time - random runs