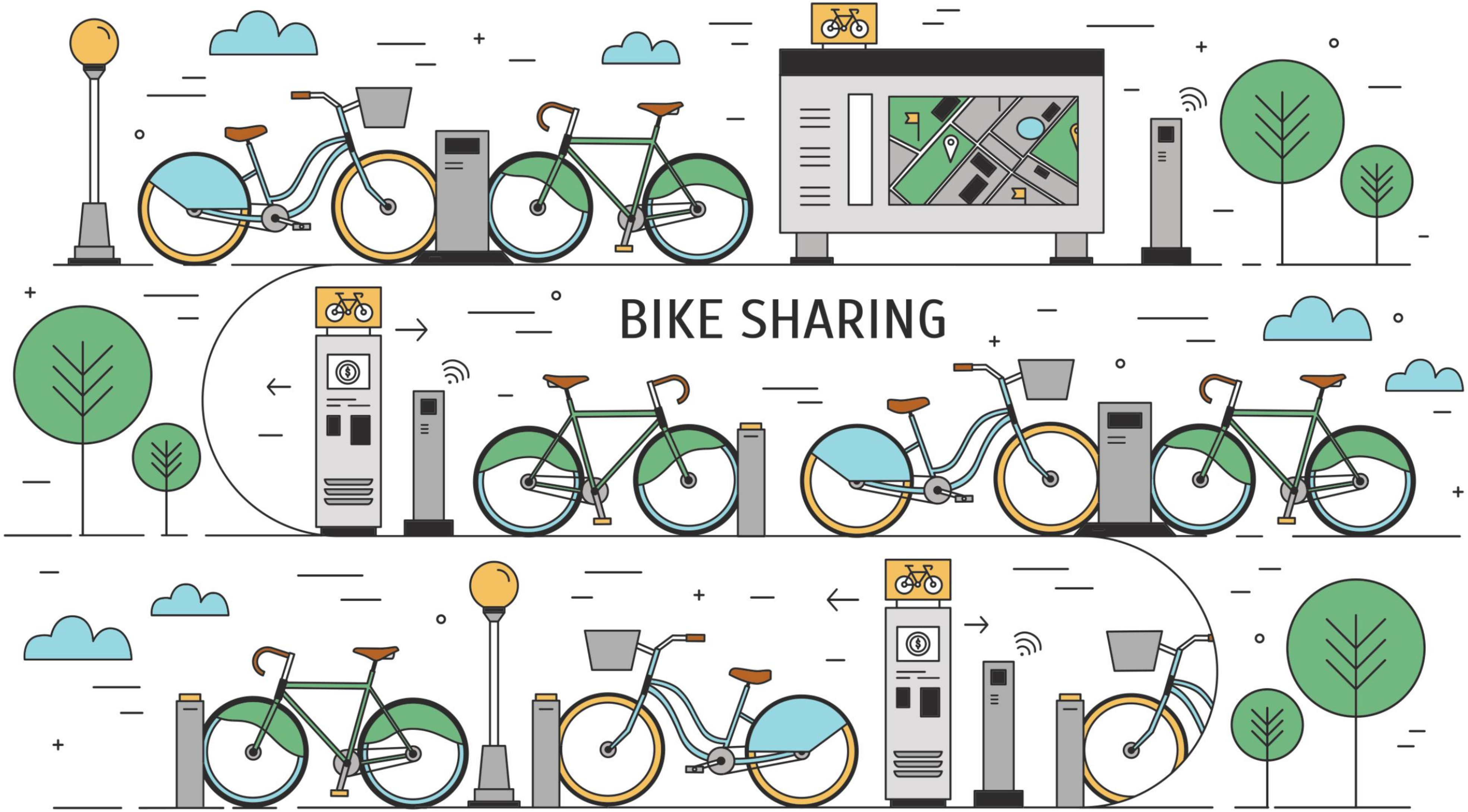
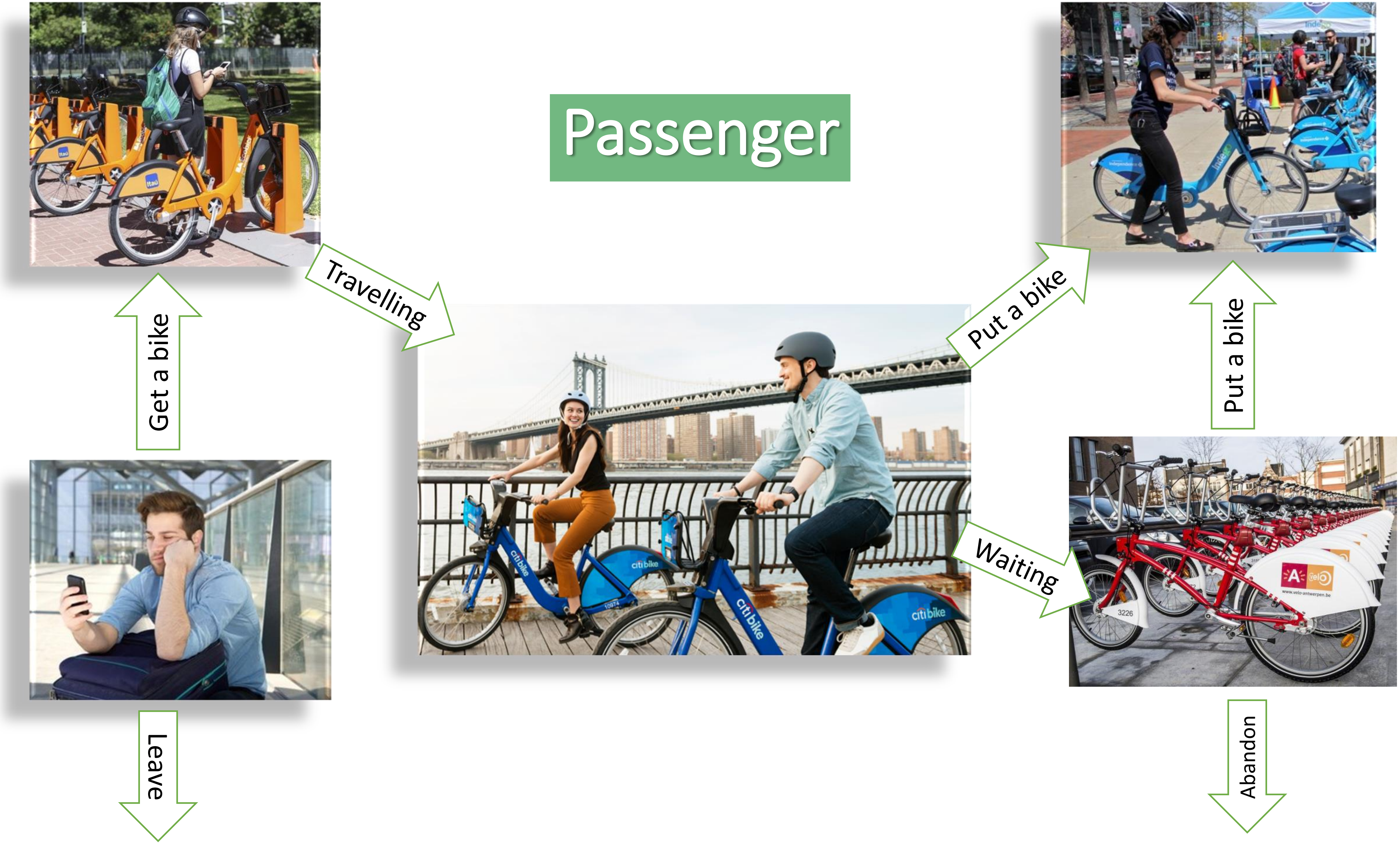
Modelling Bike Sharing with SimPy



By Xiaoyan Li and Lanpei Li

Passenger



Get a bike

Travelling

Put a bike

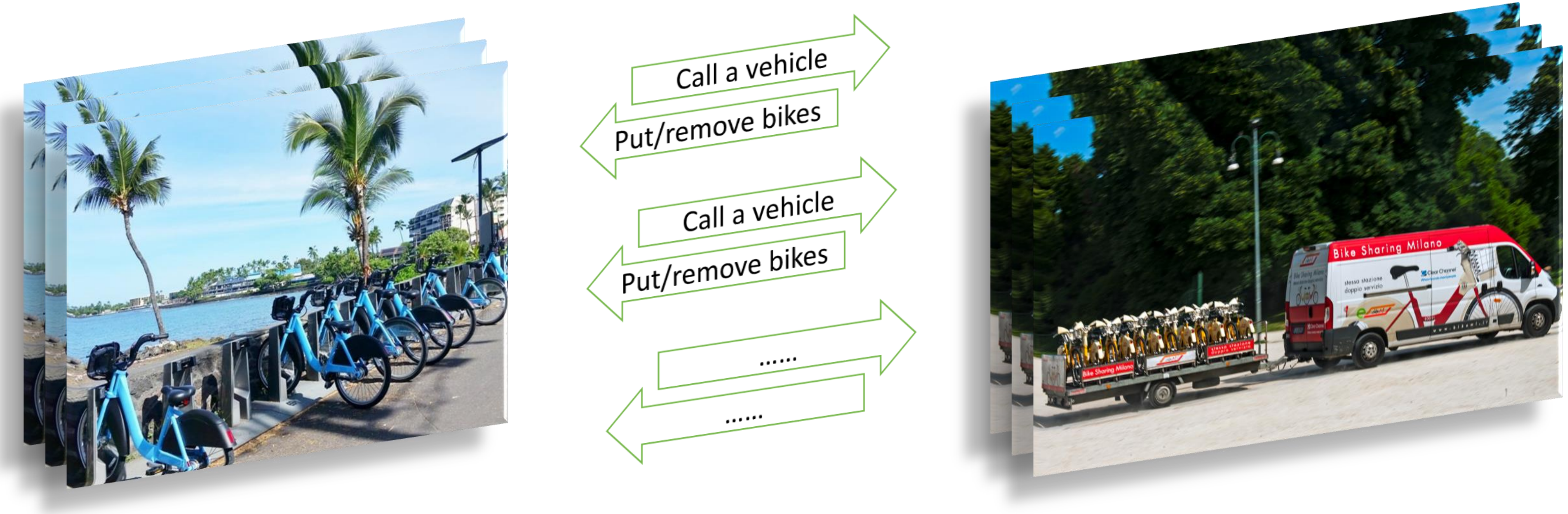
Put a bike

Waiting

Abandon

Leave

Station



Call a vehicle

Put/remove bikes

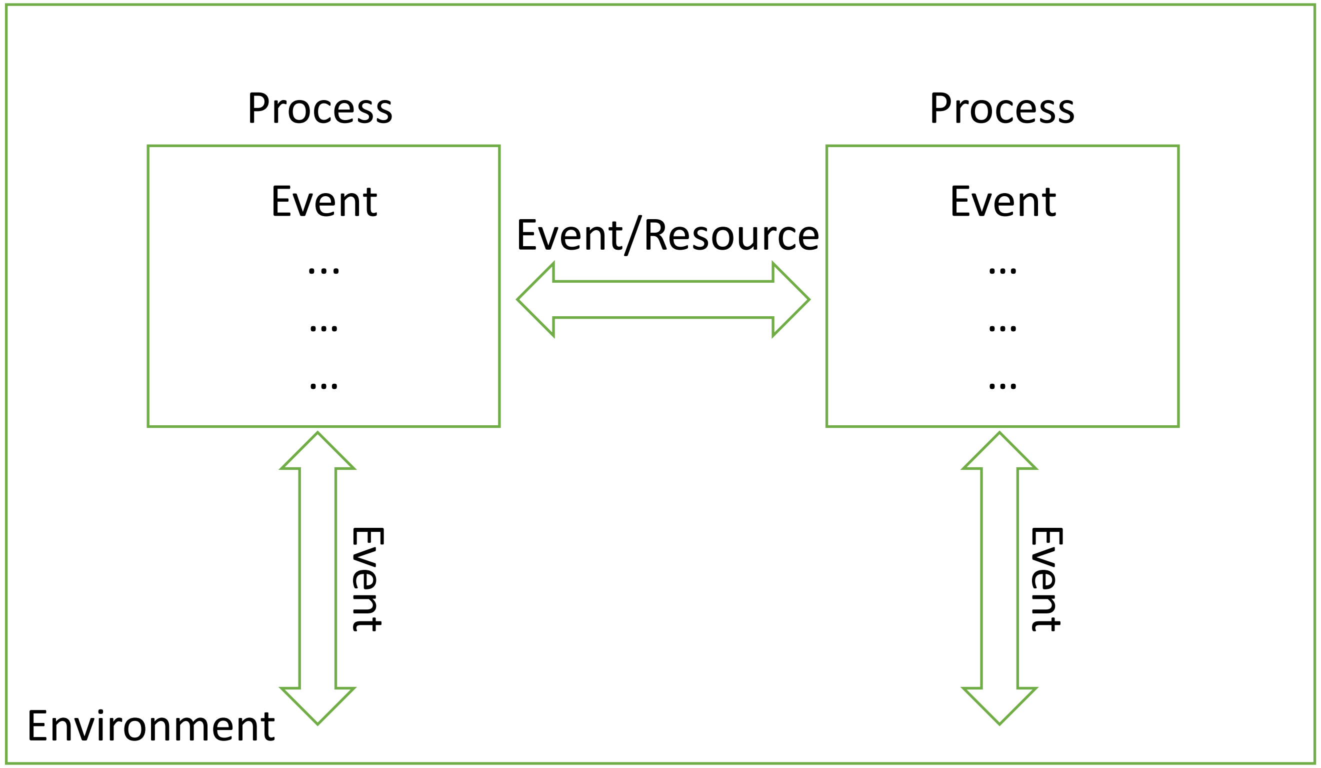
Call a vehicle

Put/remove bikes

......

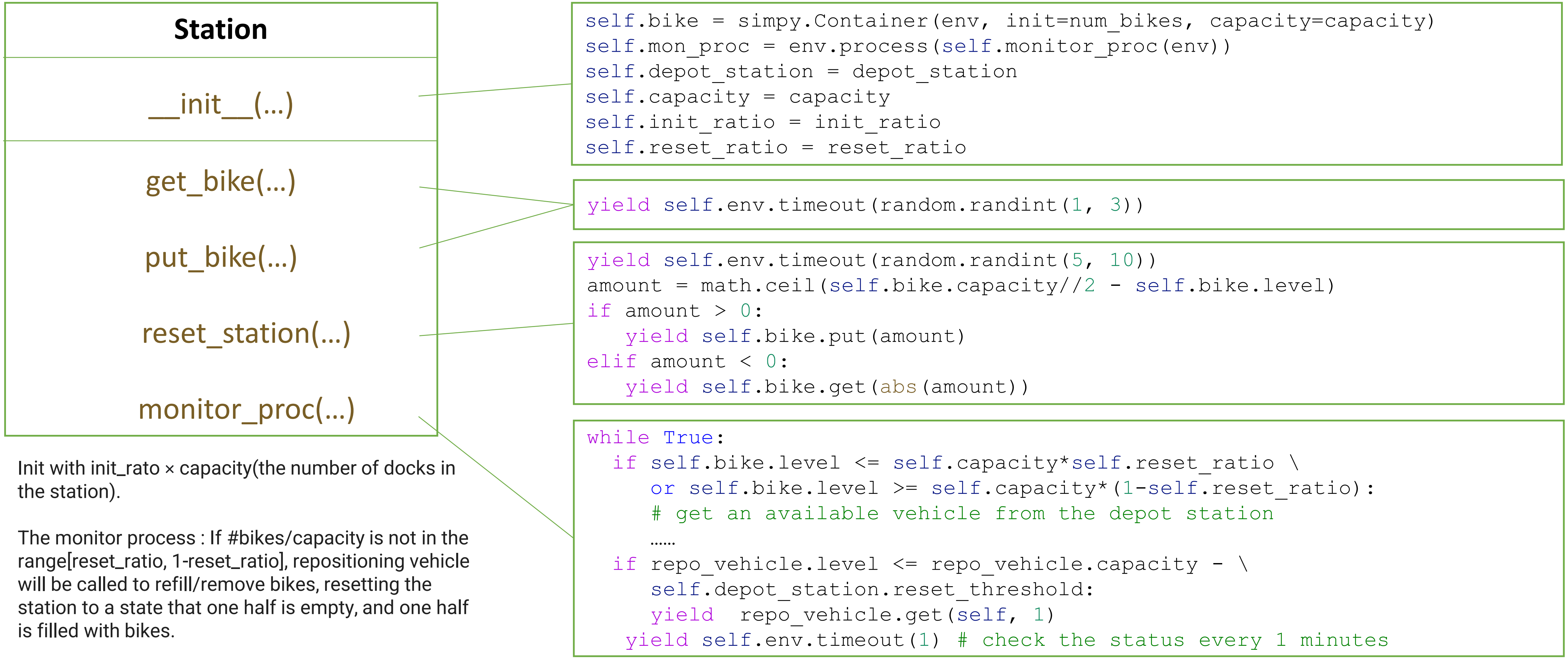
......

Simpy



- **Processes** : modelling behaviour of active components.
- **Events** : Process, Timeout,…
- **Shared resources** :
  - **Resources** – Resources that can be used by a limited number of processes at a time .
  - **Containers** – Resources that model the production and consumption of a homogeneous, undifferentiated bulk.
  - **Stores** – Resources that allow the production and consumption of Python objects.

Implementation



Station

__init__(...)

get_bike(...)

put_bike(...)

reset_station(...)

monitor_proc(...)

```
self.bike = simpy.Container(env, init=num_bikes, capacity=capacity)
self.mon_proc = env.process(self.monitor_proc(env))
self.depot_station = depot_station
self.capacity = capacity
self.init_ratio = init_ratio
self.reset_ratio = reset_ratio
```

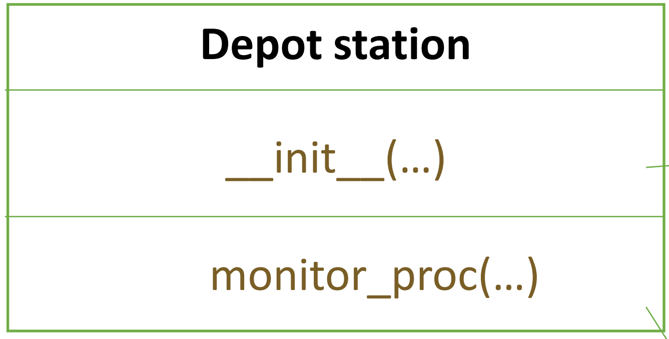
```
yield self.env.timeout(random.randint(1, 3))
```

```
yield self.env.timeout(random.randint(5, 10))
amount = math.ceil(self.bike.capacity//2 - self.bike.level)
if amount > 0:
   yield self.bike.put(amount)
elif amount < 0:
   yield self.bike.get(abs(amount))
```

```
while True:
  if self.bike.level <= self.capacity*self.reset_ratio \
     or self.bike.level >= self.capacity*(1-self.reset_ratio):
     # get an available vehicle from the depot station
     ......
  if repo_vehicle.level <= repo_vehicle.capacity - \
     self.depot_station.reset_threshold:
     yield  repo_vehicle.get(self, 1)
   yield self.env.timeout(1) # check the status every 1 minutes
```

- Init with init_rato × capacity(the number of docks in the station).
- The monitor process : If #bikes/capacity is not in the range[reset_ratio, 1-reset_ratio], repositioning vehicle will be called to refill/remove bikes, resetting the station to a state that one half is empty, and one half is filled with bikes.

Implementation



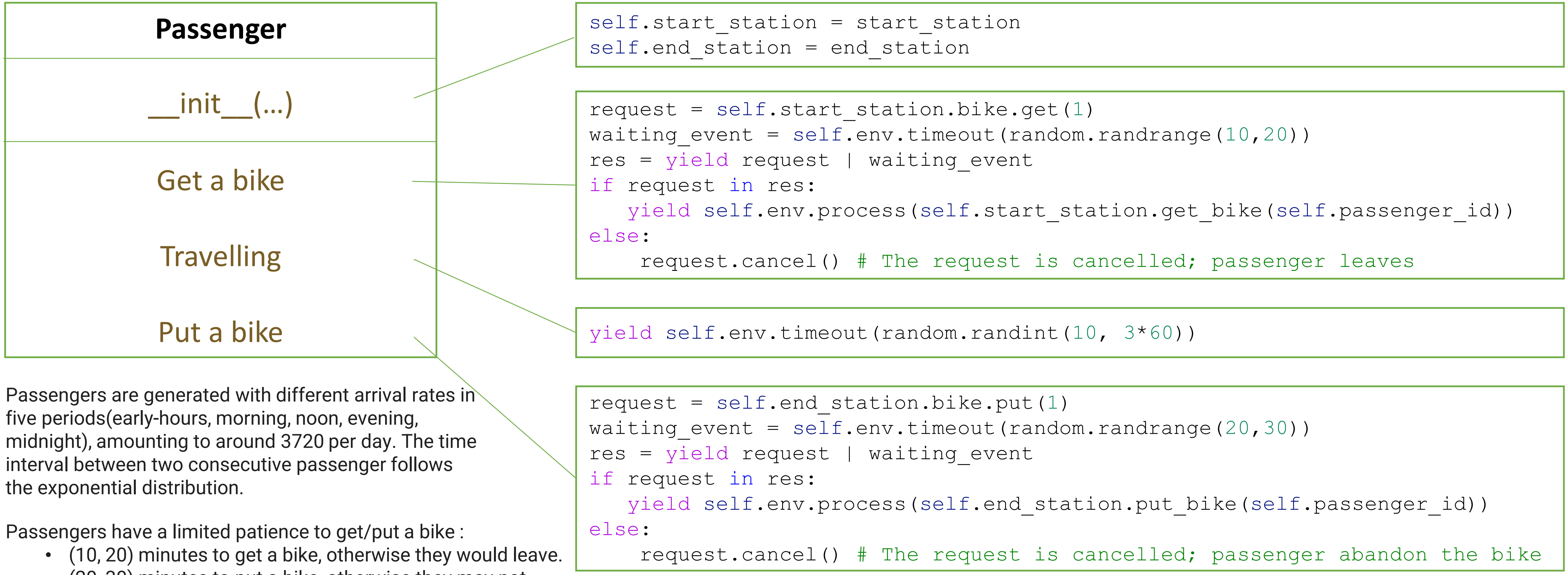
```
self.vehicle_store = [Vehicle(id, MyContainer(env, init=num_stations//2, \
                      capacity=num_stations//2), True) \
                      for id in range(num_vehicles)]
self.reset_threshold = reset_threshold
self.reset_delay = reset_delay
for c_id in range(num_vehicles):
    self.mon_proc = env.process(self.monitor_proc(env, c_id))
```

```
while True:
   # get the repo vehicle with c_id
   ……
   # calculate departure_time_interval
   ……
   waiting_event = self.env.timeout(departure_time_interval)
   yield vehicle_signal_start | waiting_event
   if len(repo_vehicle.stations) > 0:
   # get station list stas from repo_vehicle.stations
   ……
   for station in stas:
        yield self.env.process(station.reset_station())
        yield repo_vehicle.put(station, 1)
   yield self.env.timeout(1) # check the status every 1 minutes
```

- Init with capacity(num_vehicles). A vehicle is a container with capacity 10, meaning one vehicle can only serve at most 10 stations during one trip.

- The monitor process : when a vehicle receives calls from some number of stations, then it start departing to serve them. Note the number could be different in peak commuting hours.

Implementation



```
self.start_station = start_station
self.end_station = end_station
```

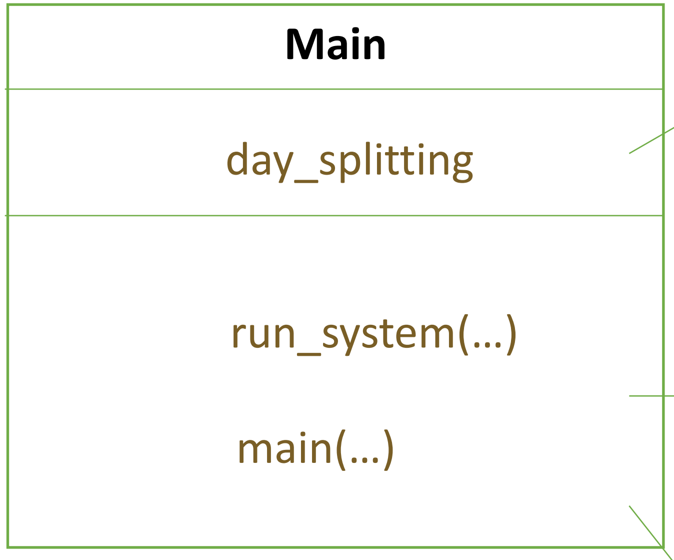
```
request = self.start_station.bike.get(1)
waiting_event = self.env.timeout(random.randrange(10,20))
res = yield request | waiting_event
if request in res:
   yield self.env.process(self.start_station.get_bike(self.passenger_id))
else:
    request.cancel() # The request is cancelled; passenger leaves
```

```
yield self.env.timeout(random.randint(10, 3*60))
```

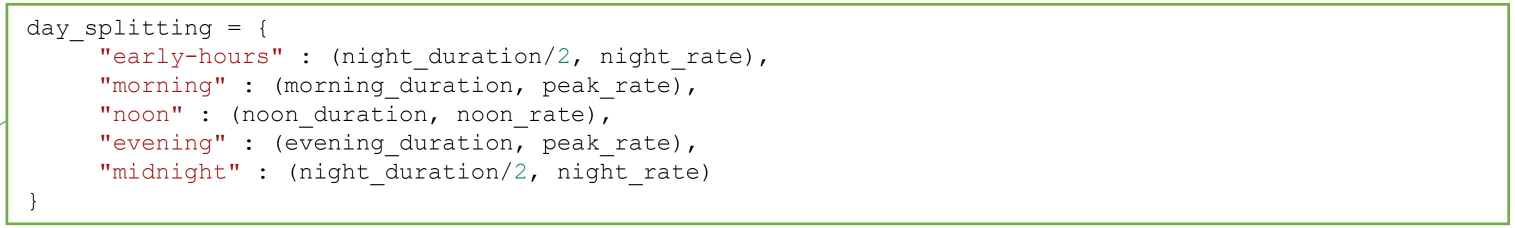
```
request = self.end_station.bike.put(1)
waiting_event = self.env.timeout(random.randrange(20,30))
res = yield request | waiting_event
if request in res:
   yield self.env.process(self.end_station.put_bike(self.passenger_id))
else:
    request.cancel() # The request is cancelled; passenger abandon the bike
```

- Passengers are generated with different arrival rates in five periods(early-hours, morning, noon, evening, midnight), amounting to around 3720 per day. The time interval between two consecutive passenger follows the exponential distribution.
- Passengers have a limited patience to get/put a bike :
  - (10, 20) minutes to get a bike, otherwise they would leave.
  - (20, 30) minutes to put a bike, otherwise they may not park the bike in the dock properly.

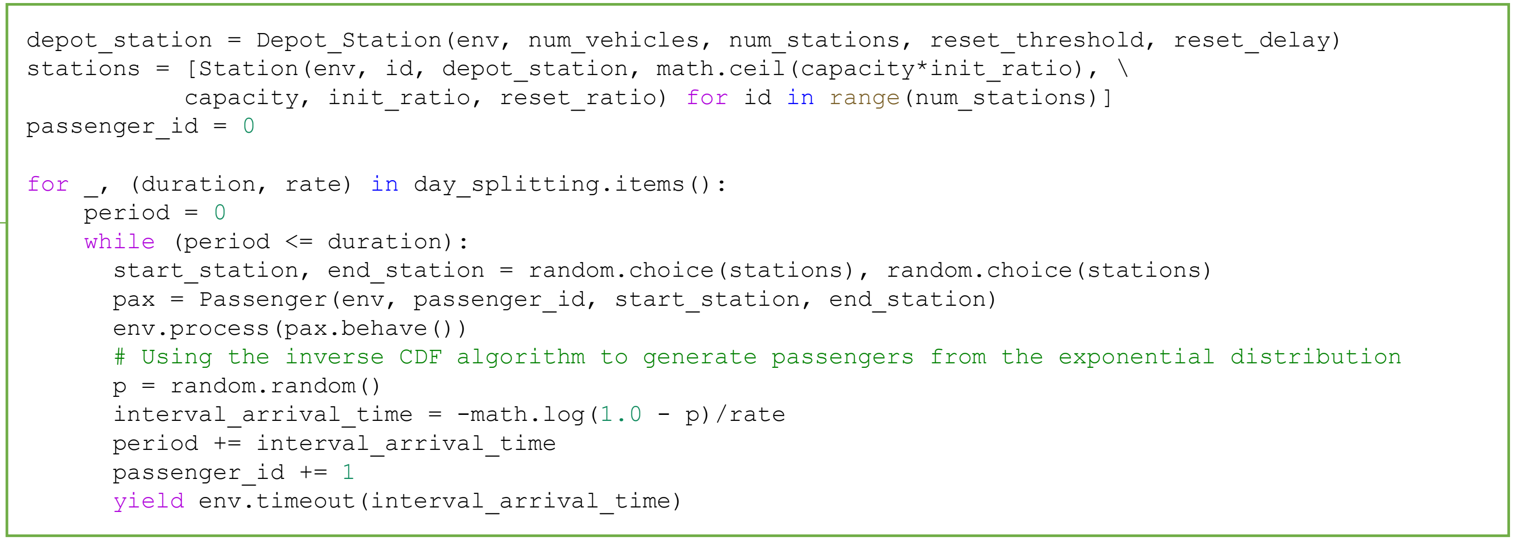
Implementation



- Run the system.

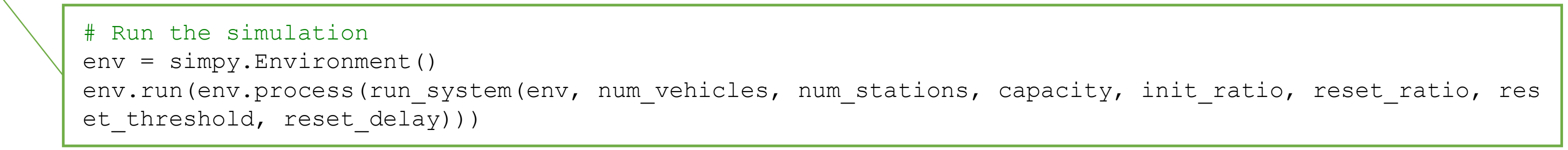


```
day_splitting = {
     "early-hours" : (night_duration/2, night_rate),
     "morning" : (morning_duration, peak_rate),
     "noon" : (noon_duration, noon_rate),
     "evening" : (evening_duration, peak_rate),
     "midnight" : (night_duration/2, night_rate)
}
```



```
depot_station = Depot_Station(env, num_vehicles, num_stations, reset_threshold, reset_delay)
stations = [Station(env, id, depot_station, math.ceil(capacity*init_ratio), \
           capacity, init_ratio, reset_ratio) for id in range(num_stations)]
passenger_id = 0

for _, (duration, rate) in day_splitting.items():
    period = 0
    while (period <= duration):
      start_station, end_station = random.choice(stations), random.choice(stations)
      pax = Passenger(env, passenger_id, start_station, end_station)
      env.process(pax.behave())
      # Using the inverse CDF algorithm to generate passengers from the exponential distribution
      p = random.random()
      interval_arrival_time = -math.log(1.0 - p)/rate
      period += interval_arrival_time
      passenger_id += 1
      yield env.timeout(interval_arrival_time)
```



```
# Run the simulation
env = simpy.Environment()
env.run(env.process(run_system(env, num_vehicles, num_stations, capacity, init_ratio, reset_ratio, res
et_threshold, reset_delay)))
```

Case study

- Base case (no **Patience strategy**, no **Refilling strategy**).

N.stations	20
Capacity	25
Init-ratio	0.7

Average-waiting-time	33 mins 19 secs
Max-waiting-time	374 mins 56 secs
N_passengers_ arrived	3743
N_passengers_served	3428

- Base case + **Patience strategy**.

N.stations	20
Capacity	25
Init-ratio	0.7

Average-waiting-time	4 mins 24 secs
Max-waiting-time	29 mins 0 secs
N_impatiences_get	705
N_impatiences_put	9
N_passengers_ arrived	3713
N_passengers_served	2904

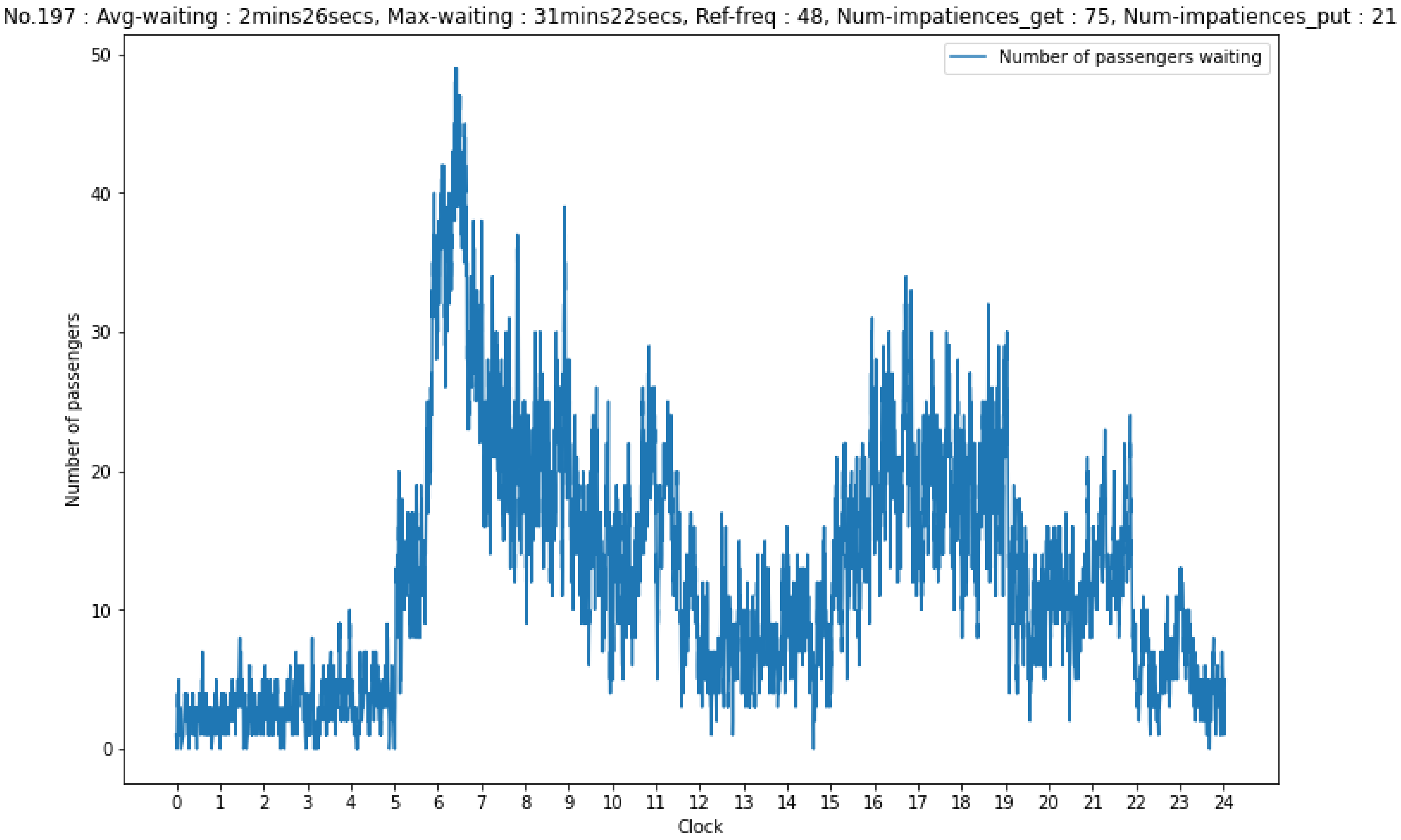
with Refilling strategy

N.vehicles	20	10	5	5	5	5	5
Capacity	25	25	25	25	5	5	25
Init-ratio	0.7	0.6	0.7	0.8	0.6	0.7	0.6
Reset-ratio	0.5	0.5	0.1	0.3	0.1	0.5	0.3
Reset-threshold	6	4	8	6	4	8	6
Reset-delay	10	10	30	20	10	30	30
Average-waiting-time	1 mins 60 secs	1 mins 60 secs	2 mins 26 secs	2 mins 3 secs	6 mins 23 secs	6 mins 53 secs	2 mins 7 secs
Max-waiting-time	13 mins 38 secs	10 mins 60 secs	31 mins 22 secs	26 mins 19 secs	30 mins 6 secs	30 mins 29 secs	28 mins 40 secs
Refilling-times	244	288	48	101	174	78	75
N_impatiences_get	0	0	75	4	966	1126	6
N_impatiences_put	0	0	21	0	142	176	6

1. **num_stations** (20) : the number of stations.
2. **num_vehicles** ([5,10,20]) : the number of vehicles in a depot station.
3. **capacity** ([5,15,25]) : the capacity of each station.
4. **init_ratio** ([0.6,0.7,0.8]) : #bikes / capacity in each station initially.
5. **reset ratio** ([0.1,0.3,0.5]) : #bikes/capacity to call for repositioning vehicles.
6. **reset_threshold** ([4,6,8]) : the number of stations received (10 - reset_threshold) for a vehicle to depart.
7. **reset_delay** ([10,20,30]) : the time interval to send a vehicle.

Plotting

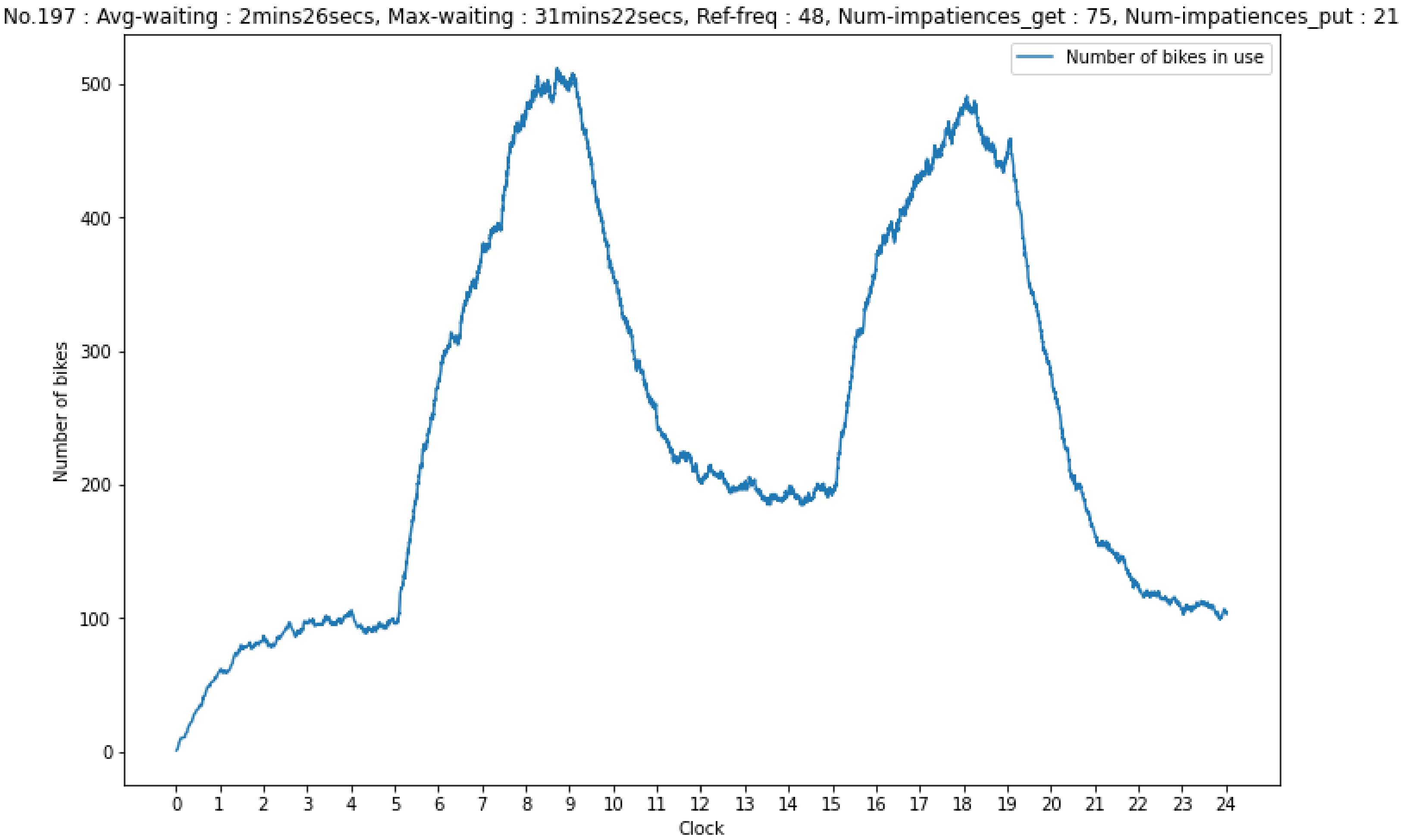
N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	8
Reset-delay	30



No.197 : Avg-waiting : 2mins26secs, Max-waiting : 31mins22secs, Ref-freq : 48, Num-impatiences_get : 75, Num-impatiences_put : 21

Plotting

N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	8
Reset-delay	30

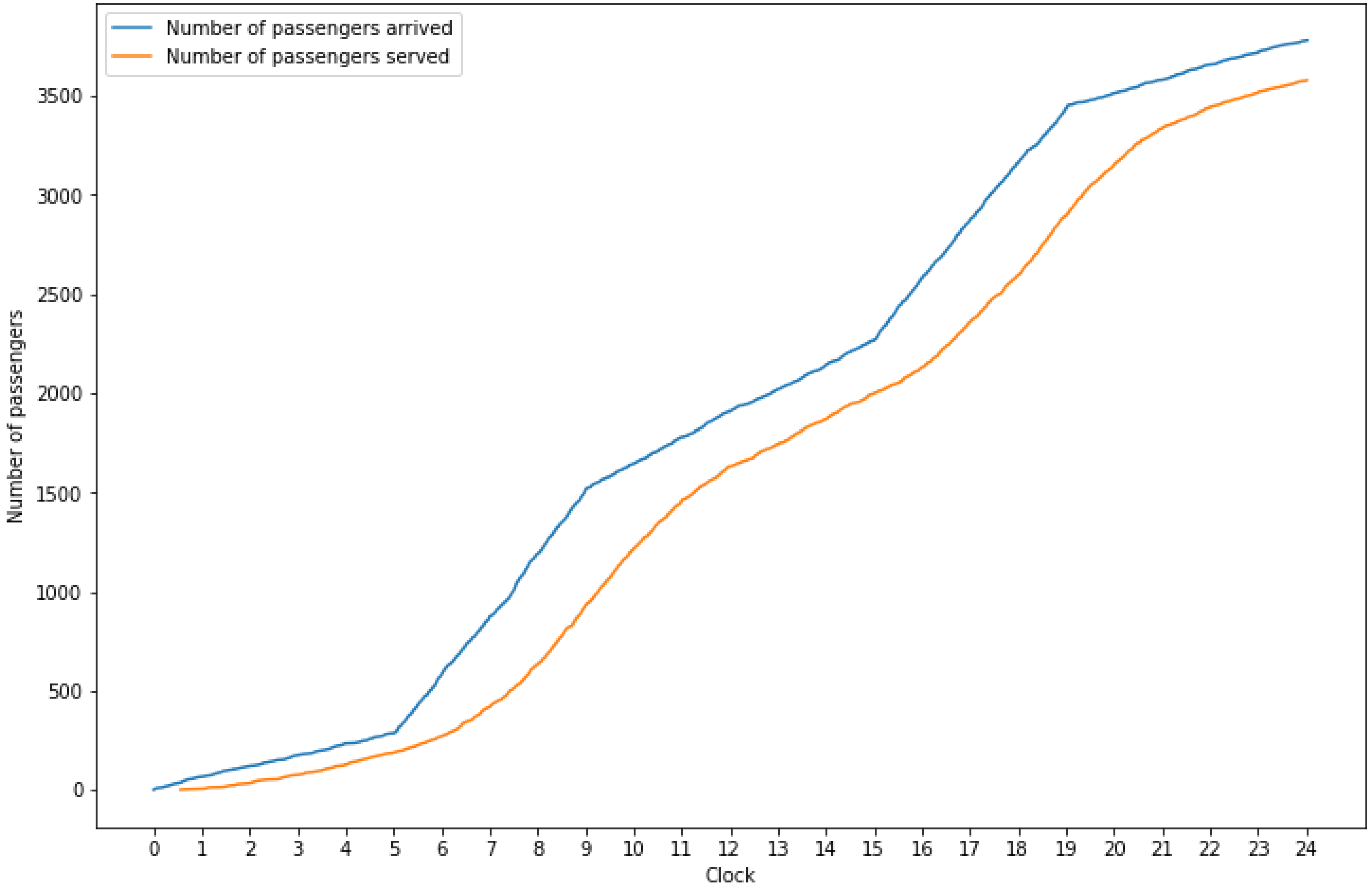


No.197 : Avg-waiting : 2mins26secs, Max-waiting : 31mins22secs, Ref-freq : 48, Num-impatiences_get : 75, Num-impatiences_put : 21

Plotting

N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	8
Reset-delay	30

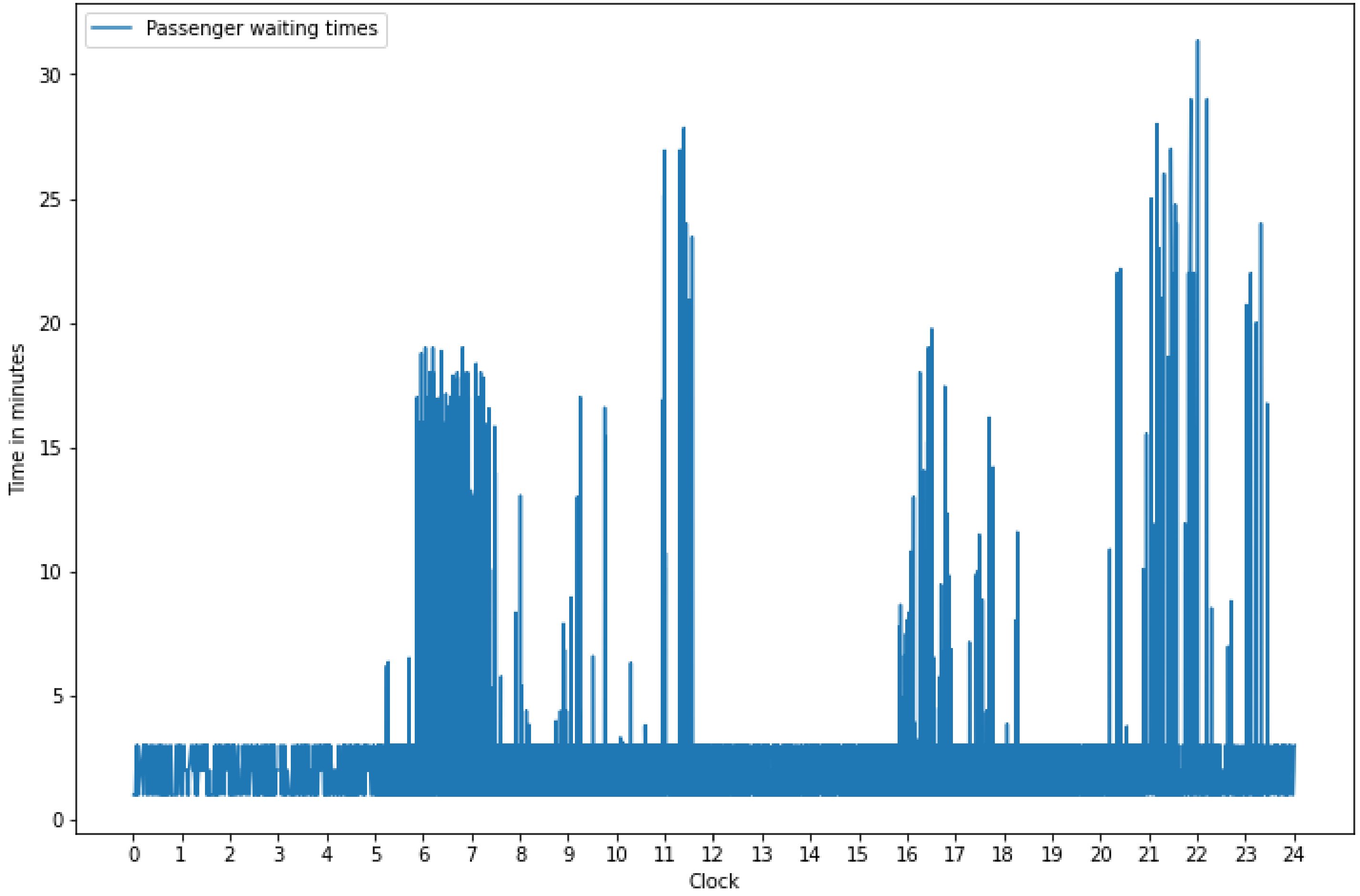
No.197 : Avg-waiting : 2mins26secs, Max-waiting : 31mins22secs, Ref-freq : 48, Num-impatiences_get : 75, Num-impatiences_put : 21



Plotting

N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	8
Reset-delay	30

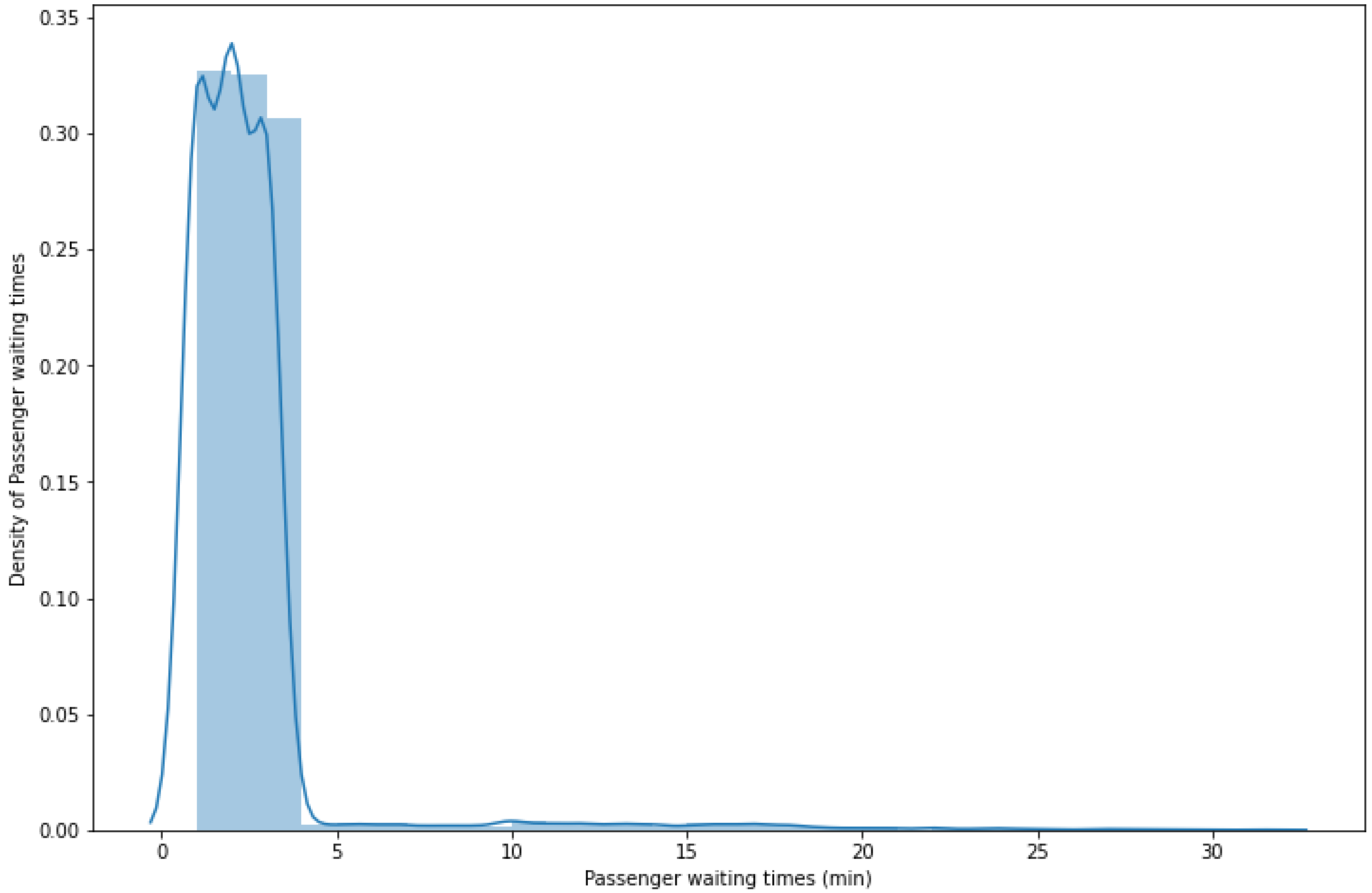
No.197 : Avg-waiting : 2mins26secs, Max-waiting : 31mins22secs, Ref-freq : 48, Num-impatiences_get : 75, Num-impatiences_put : 21



Plotting

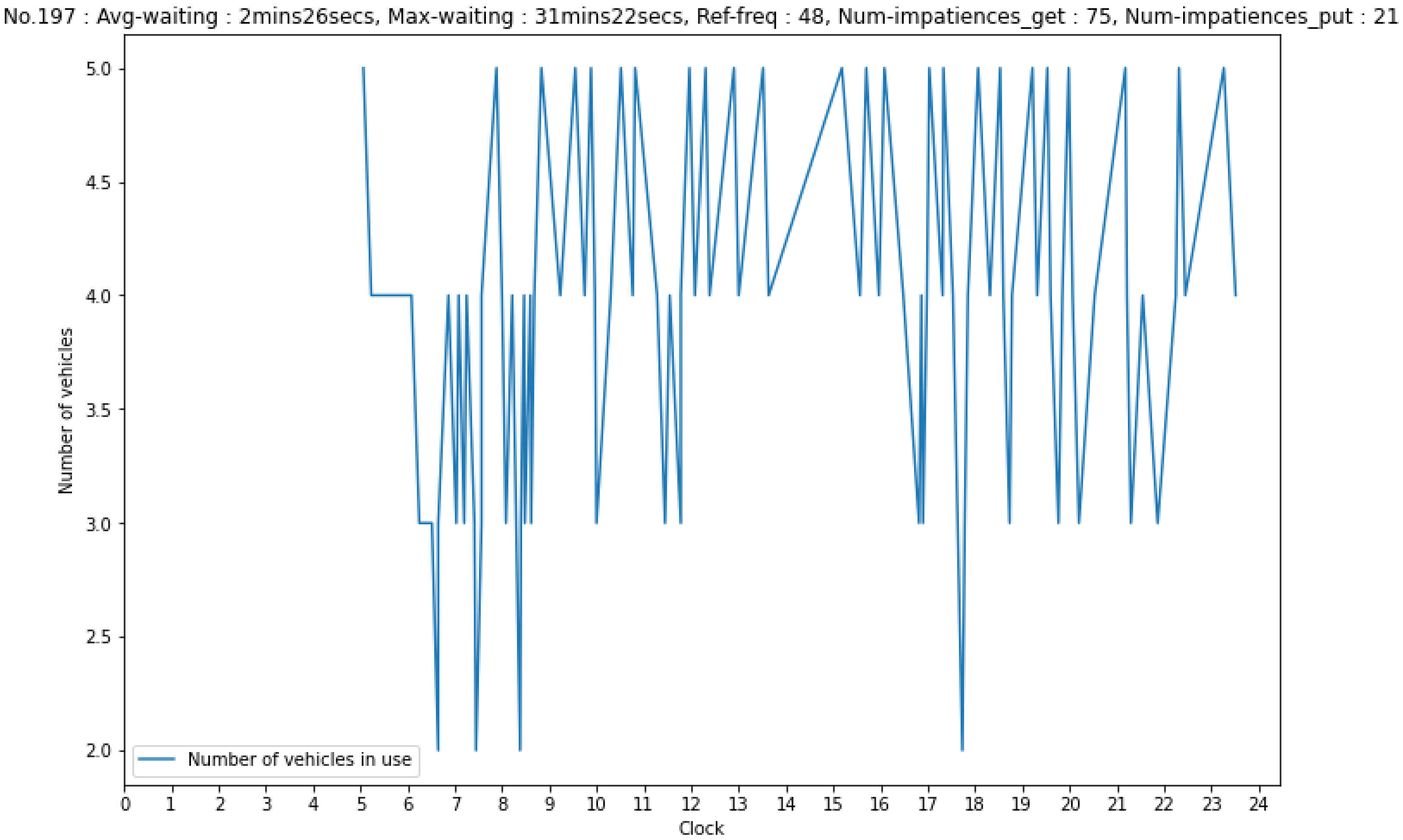
N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	8
Reset-delay	30

No.197 : Avg-waiting : 2mins26secs, Max-waiting : 31mins22secs, Ref-freq : 48, Num-impatiences_get : 75, Num-impatiences_put : 21



Plotting

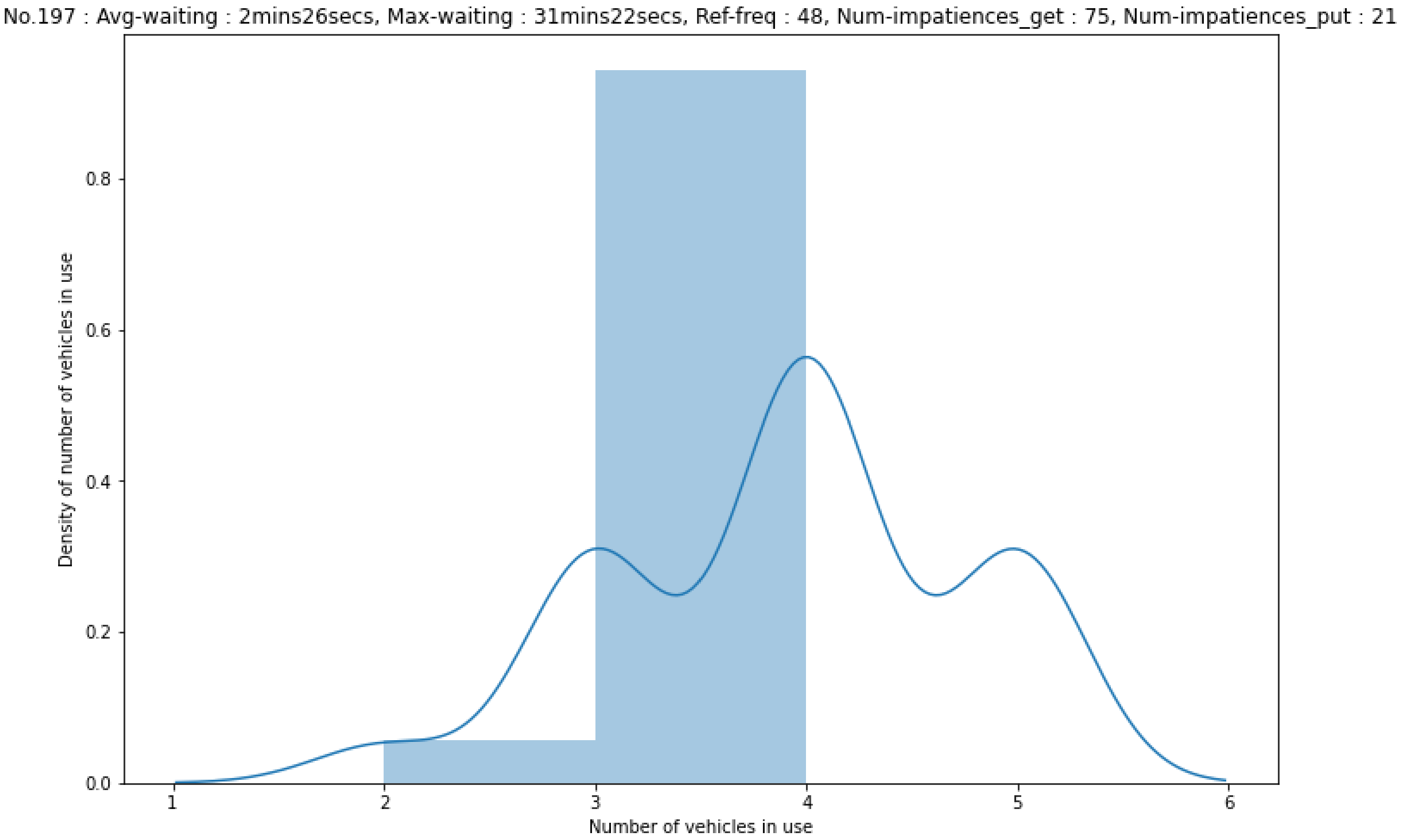
N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	8
Reset-delay	30



No.197 : Avg-waiting : 2mins26secs, Max-waiting : 31mins22secs, Ref-freq : 48, Num-impatiences_get : 75, Num-impatiences_put : 21

Plotting

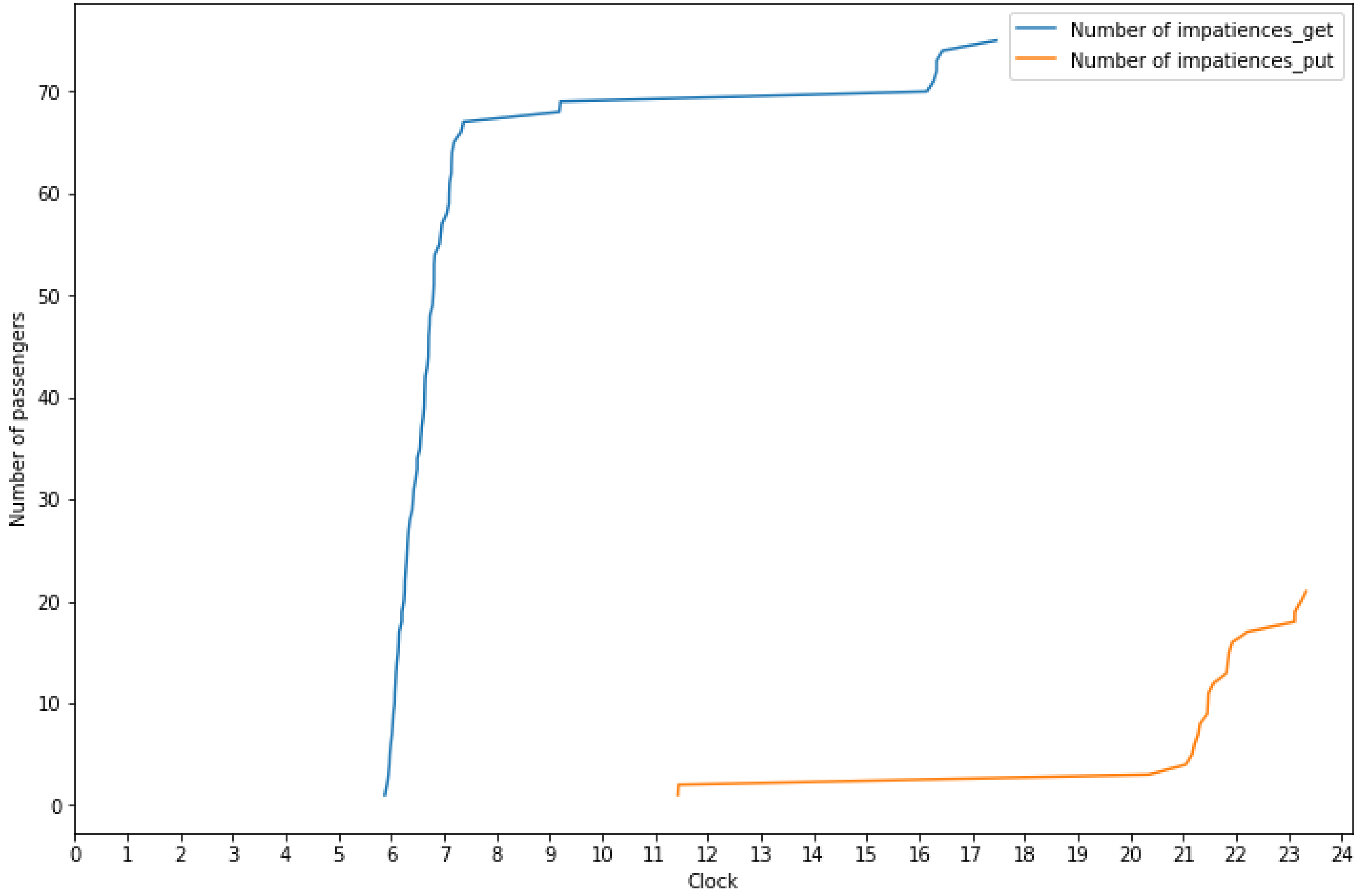
N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	8
Reset-delay	30



Plotting

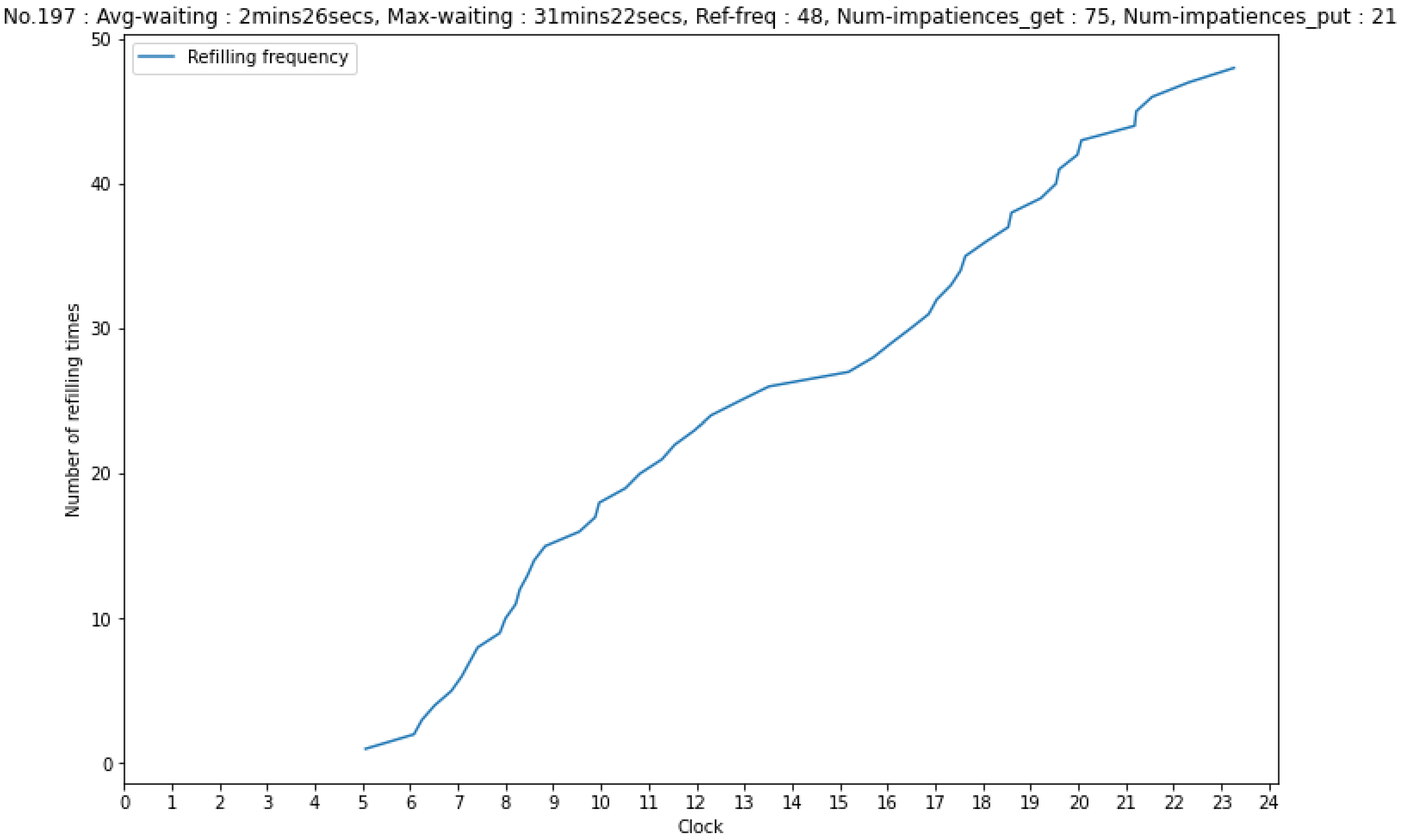
N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	8
Reset-delay	30

No.197 : Avg-waiting : 2mins26secs, Max-waiting : 31mins22secs, Ref-freq : 48, Num-impatiences_get : 75, Num-impatiences_put : 21



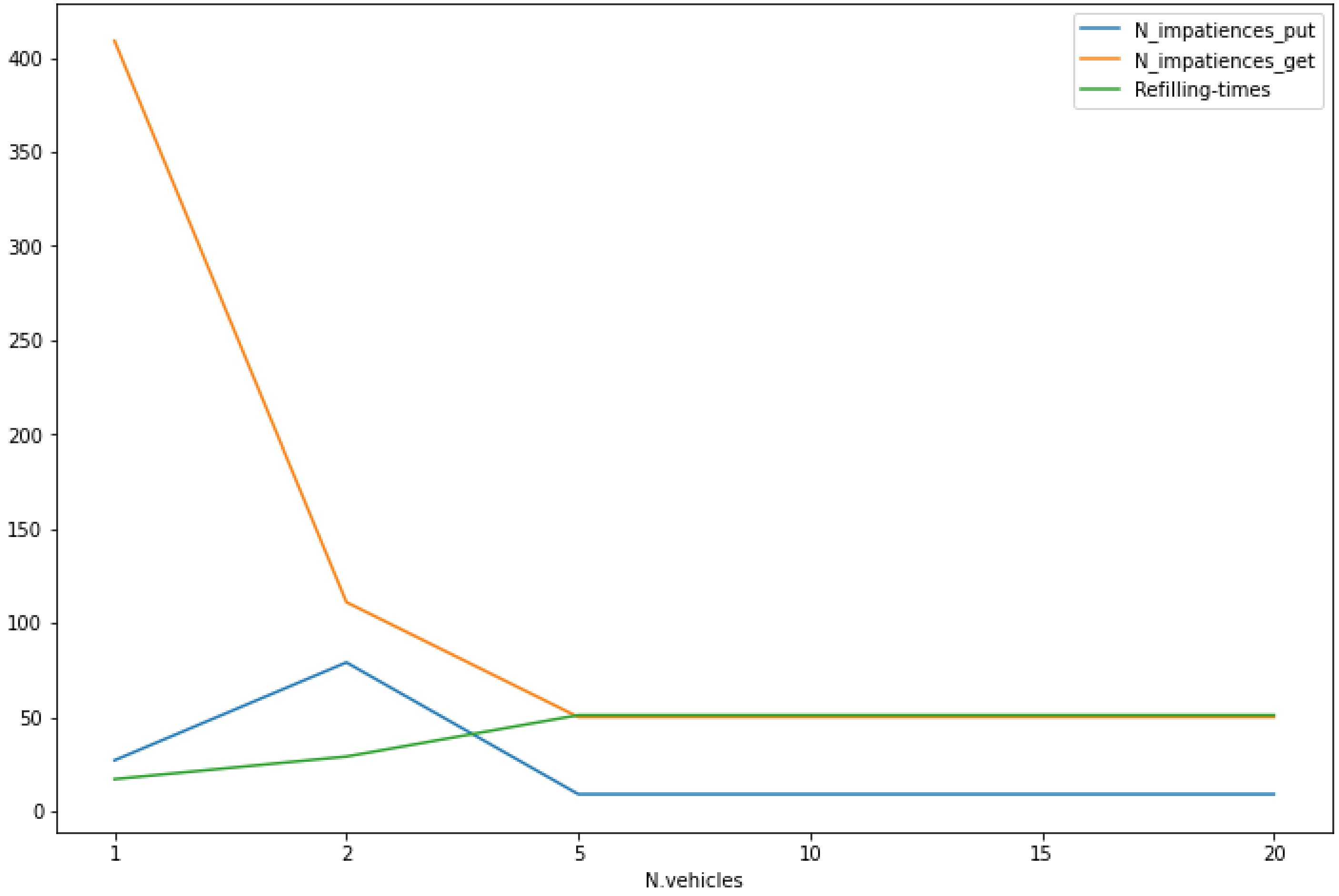
Plotting

N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	8
Reset-delay	30


No.197 : Avg-waiting : 2mins26secs, Max-waiting : 31mins22secs, Ref-freq : 48, Num-impatiences_get : 75, Num-impatiences_put : 21

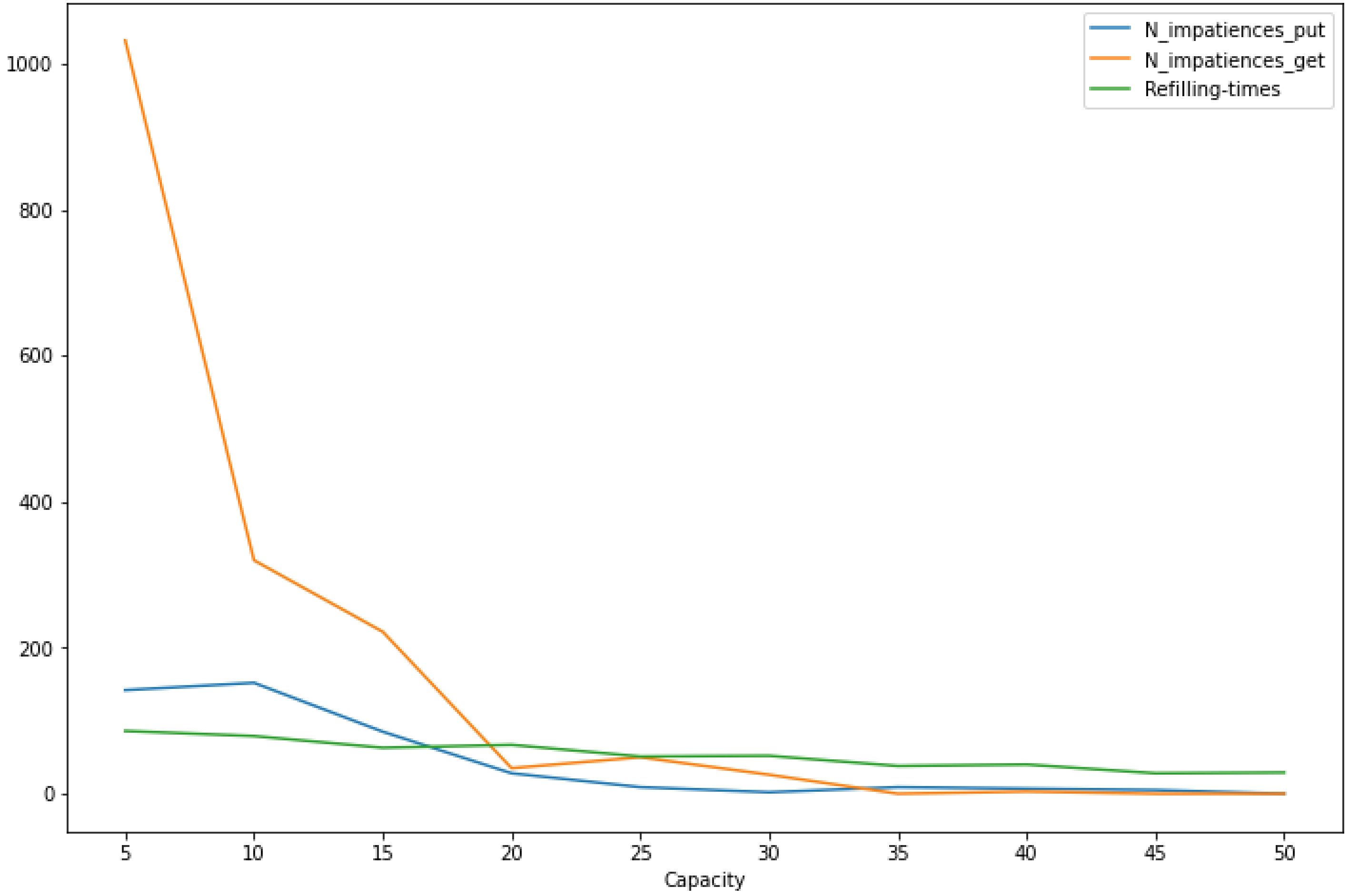
Grid search

N.stations	20
N.vehicles	1,2,5,10,15,20
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	6
Reset-delay	30



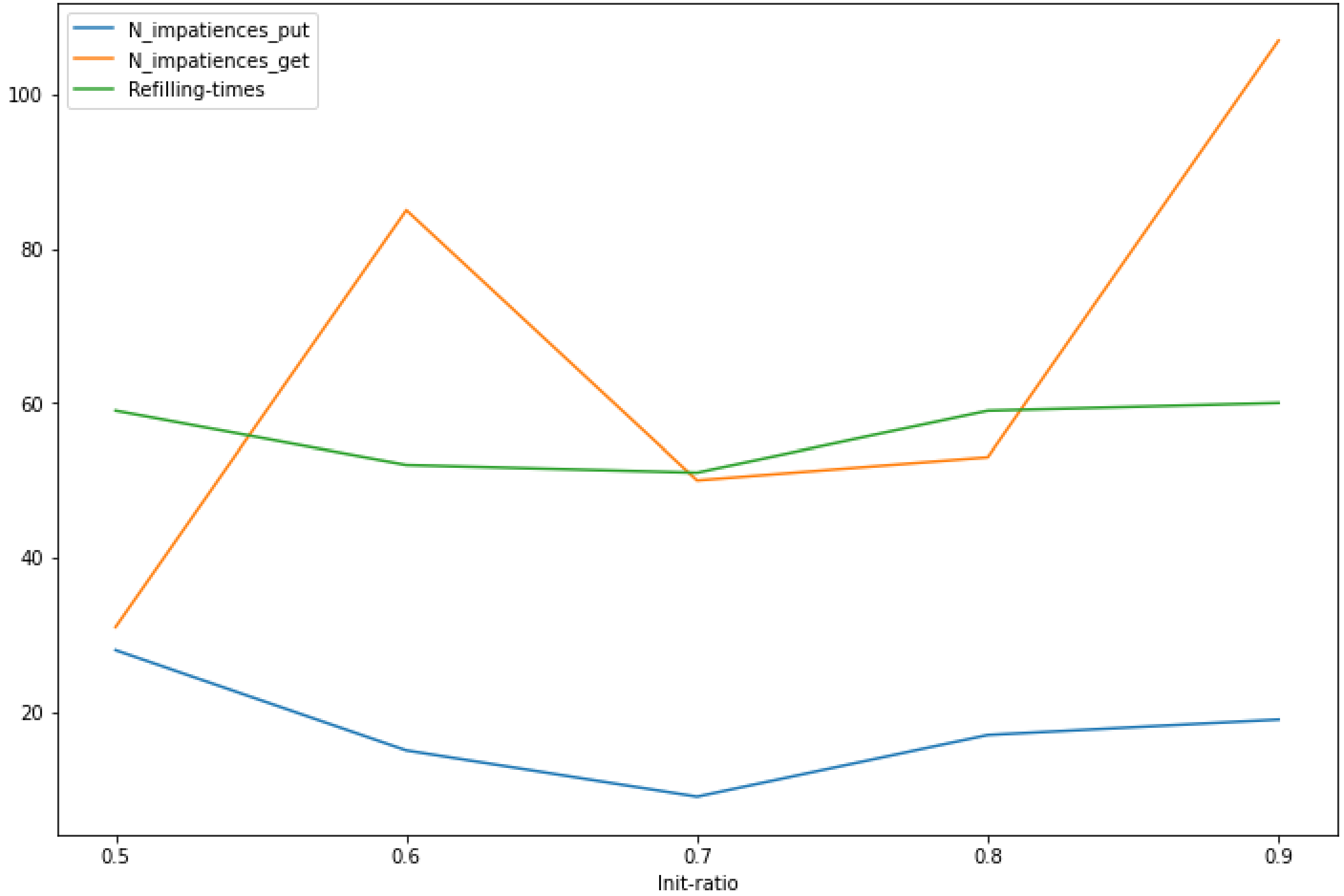
Grid search

N.stations	20
N.vehicles	5
Capacity	5,10,15,20,25,30, 35,40,45,50
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	6
Reset-delay	30



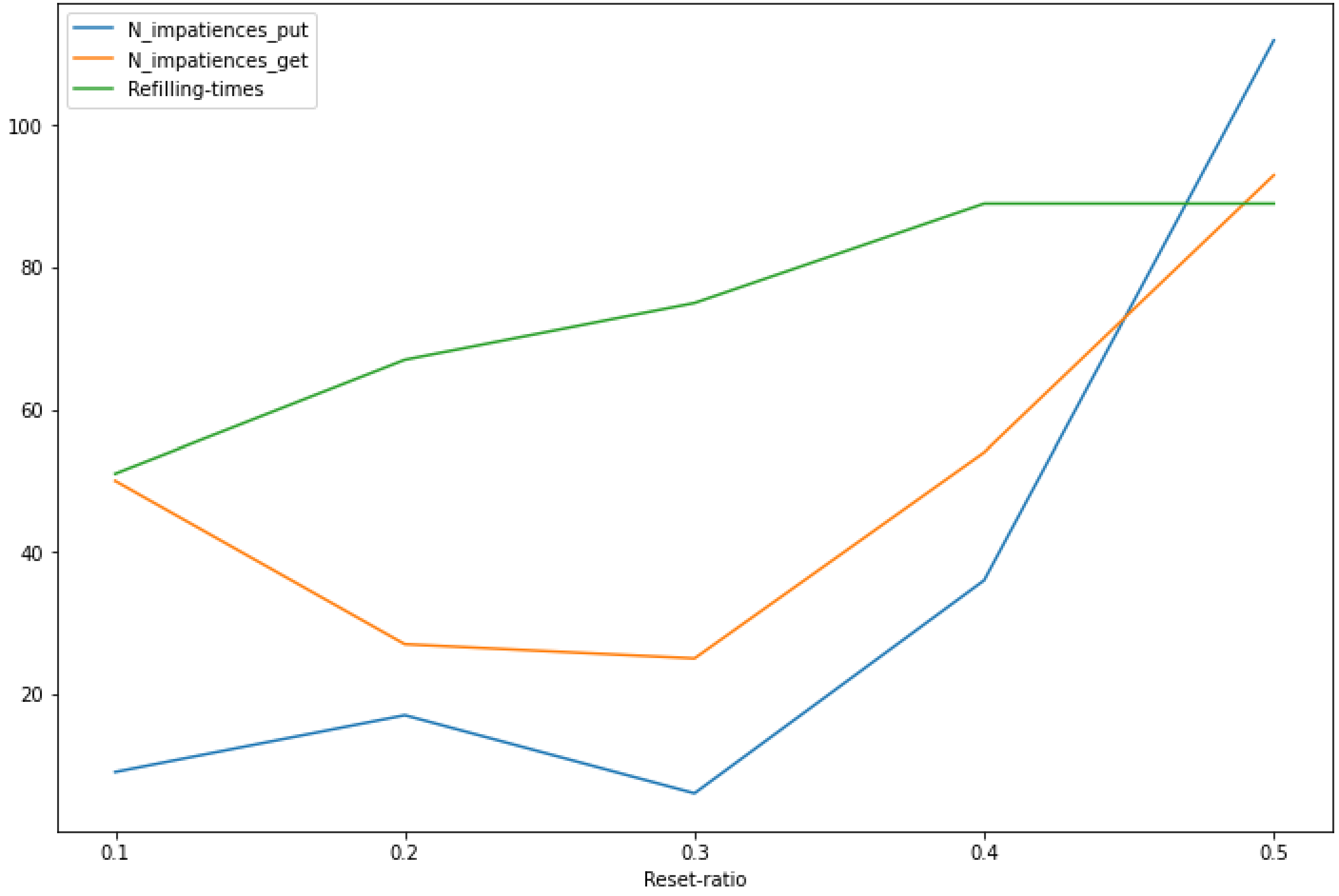
Grid search

N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.5,0.6,0.7,0.8,0.9
Reset-ratio	0.1
Reset-threshold	6
Reset-delay	30



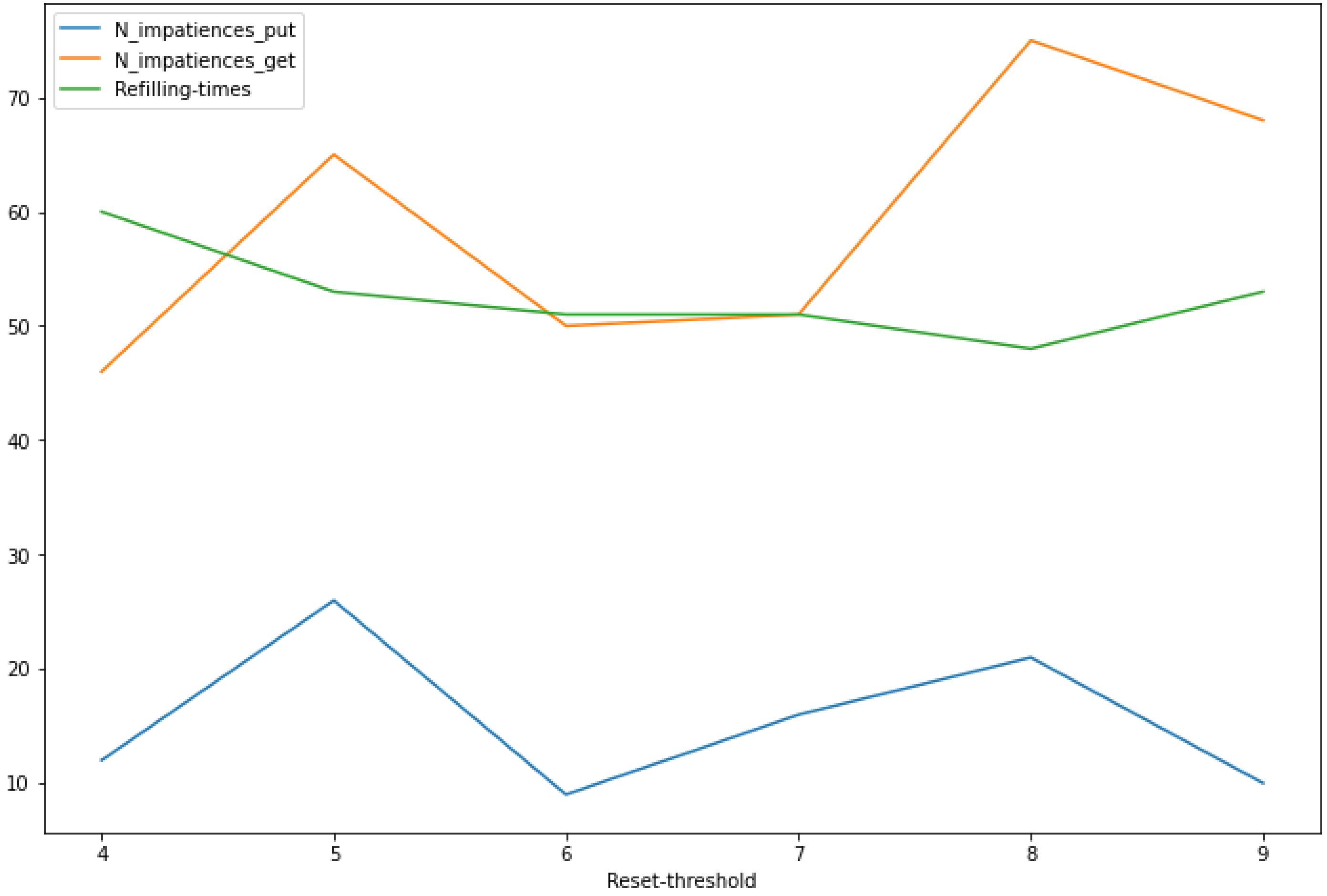
Grid search

N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1,0.2,0.3,0.4,0.5
Reset-threshold	6
Reset-delay	30



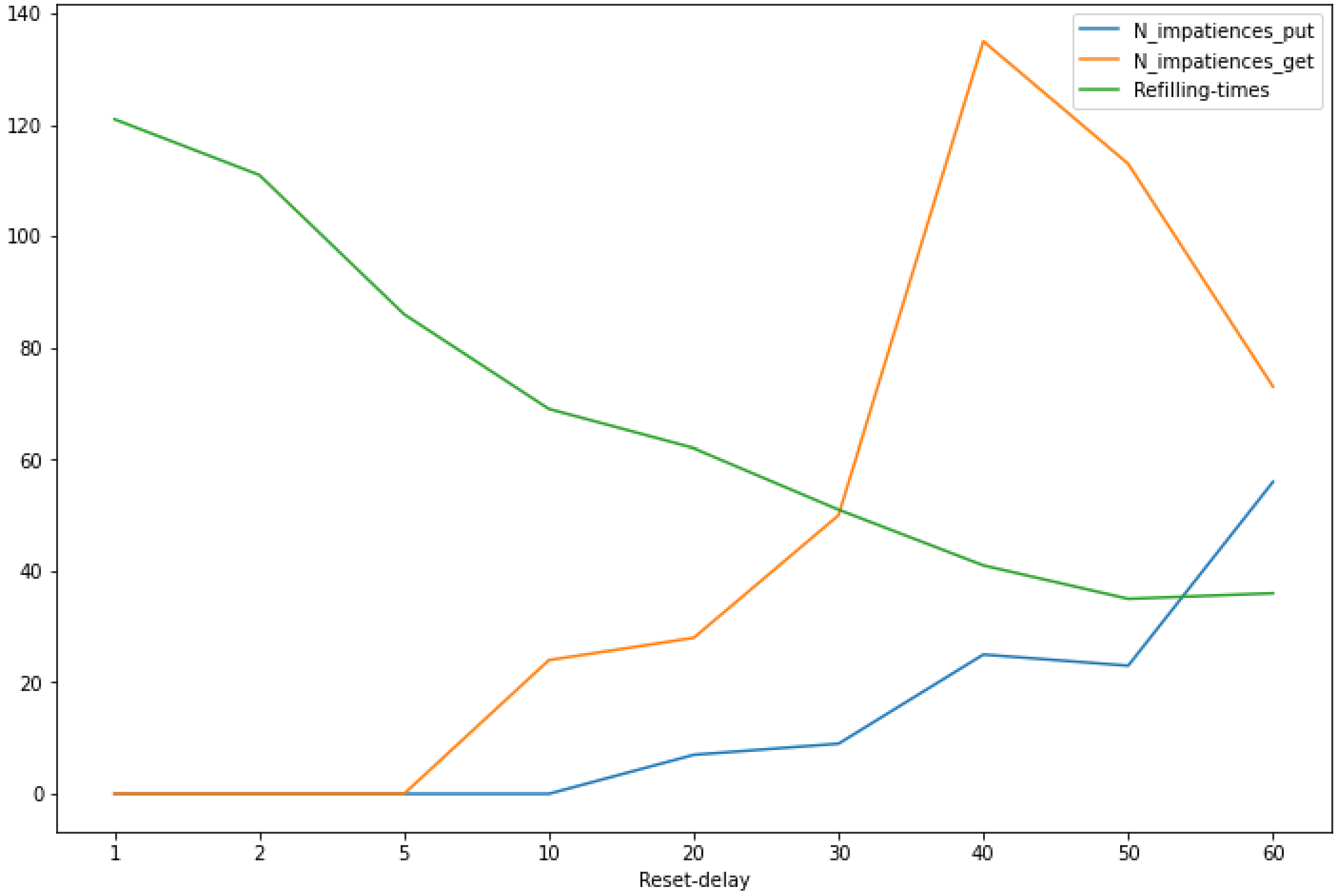
Grid search

N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	4,5,6,7,8,9
Reset-delay	30



Grid search

N.stations	20
N.vehicles	5
Capacity	25
Init-ratio	0.7
Reset-ratio	0.1
Reset-threshold	6
Reset-delay	1,2,5,10,20,30,40, 50,60



Thank you !