from \_\_future\_\_ import absolute\_import

from \_\_future\_\_ import print\_function

import os

import sys

import optparse

import random

# we need to import python modules from the $SUMO\_HOME/tools directory

if 'SUMO\_HOME' in os.environ:

 tools = os.path.join(os.environ['SUMO\_HOME'], 'tools')

 sys.path.append(tools)

else:

 sys.exit("please declare environment variable 'SUMO\_HOME'")

from sumolib import checkBinary # noqa

import traci # noqa

def generate\_routefile():

 random.seed(42) # make tests reproducible

 N = 3600 # number of time steps

 # demand per second from different directions

 pWE = 1. / 10

 pEW = 1. / 11

 pNS = 1. / 30

 with open("my\_routes.rou.xml", "w") as routes:

 print("""<routes>

 <vType id="typeWE" accel="0.8" decel="4.5" sigma="0.5" length="5" minGap="2.5" maxSpeed="16.67" \

guiShape="passenger"/>

 <vType id="typeNS" accel="0.8" decel="4.5" sigma="0.5" length="7" minGap="3" maxSpeed="25" guiShape="bus"/>

 <route id="right" edges="51o 1i 2o 52i" />

 <route id="left" edges="52o 2i 1o 51i" />

 <route id="down" edges="54o 4i 3o 53i" />""", file=routes)

 vehNr = 0

 for i in range(N):

 if random.uniform(0, 1) < pWE:

 print(' <vehicle id="right\_%i" type="typeWE" route="right" depart="%i" />' % (

 vehNr, i), file=routes)

 vehNr += 1

 if random.uniform(0, 1) < pEW:

 print(' <vehicle id="left\_%i" type="typeWE" route="left" depart="%i" />' % (

 vehNr, i), file=routes)

 vehNr += 1

 if random.uniform(0, 1) < pNS:

 print(' <vehicle id="down\_%i" type="typeNS" route="down" depart="%i" color="1,0,0"/>' % (

 vehNr, i), file=routes)

 vehNr += 1

 print("</routes>", file=routes)

# The program looks like this

# <tlLogic id="0" type="static" programID="0" offset="0">

# the locations of the tls are NESW

# <phase duration="31" state="GrGr"/>

# <phase duration="6" state="yryr"/>

# <phase duration="31" state="rGrG"/>

# <phase duration="6" state="ryry"/>

# </tlLogic>

def run():

 """execute the TraCI control loop"""

 step = 0

 # we start with phase 2 where EW has green

 traci.trafficlight.setPhase("0", 2)

 while traci.simulation.getMinExpectedNumber() > 0:

 traci.simulationStep()

 if traci.trafficlight.getPhase("0") == 2:

 # we are not already switching

 if traci.inductionloop.getLastStepVehicleNumber("0") > 0:

 # there is a vehicle from the north, switch

 traci.trafficlight.setPhase("0", 3)

 else:

 # otherwise try to keep green for EW

 traci.trafficlight.setPhase("0", 2)

 step += 1

 traci.close()

 sys.stdout.flush()

def get\_options():

 optParser = optparse.OptionParser()

 optParser.add\_option("--nogui", action="store\_true",

 default=False, help="run the commandline version of sumo")

 options, args = optParser.parse\_args()

 return options

# this is the main entry point of this script

if \_\_name\_\_ == "\_\_main\_\_":

 options = get\_options()

 # this script has been called from the command line. It will start sumo as a

 # server, then connect and run

 if options.nogui:

 sumoBinary = checkBinary("C:/Program Files (x86)/Eclipse/Sumo/bin/sumo.exe")

 else:

 sumoBinary = checkBinary("C:/Program Files (x86)/Eclipse/Sumo/bin/simo-gui")

 # first, generate the route file for this simulation

 generate\_routefile()

 # this is the normal way of using traci. sumo is started as a

 # subprocess and then the python script connects and runs

 traci.start([sumoBinary, "-c", "my\_config.sumocfg",

 "--tripinfo-output", "tripinfo.xml"])

 run()