```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
# vim:fenc=utf-8
#

"""
About: Placement of a single recoder between multi-hop relays
Email: xianglinks@gmail.com
"""

import os
import random

import kodo

symbols = 100
symbol_size = 50
runs = 100


def benchmark(num_relays, err_rates, recoder_pos):

    if recoder_pos + 1 > num_relays:
        raise RuntimeError("Invalid recoder position.")
    if num_relays + 1 != len(err_rates):
        raise RuntimeError("Invalid link error rates.")

    recv_num_lst = list()

    for run in range(runs):
        encoder_factory = kodo.FullVectorEncoderFactoryBinary8(
            max_symbols=symbols,
            max_symbol_size=symbol_size)

        encoder = encoder_factory.build()
        encoder.set_systematic_off()

        decoder_factory = kodo.FullVectorDecoderFactoryBinary8(
            max_symbols=symbols,
            max_symbol_size=symbol_size)

        decoder = decoder_factory.build()

        # Recoder is implemented as decoder in Kodo
        recoder = decoder_factory.build()

        # Create random data for source
        data_in = os.urandom(encoder.block_size())
        encoder.set_const_symbols(data_in)
        encoder.set_systematic_off()

        num = 0

        # Count number of transmissions until the receiver get all packets.
        while not decoder.is_complete():

            num += 1

            # Sender sends one packet
            packet = encoder.write_payload()
            packet_lost = False

            # Go through all relays
            for r_idx in range(0, num_relays):
                if r_idx < recoder_pos:
                    # Relays before the recoder
                    if random.random() < err_rates[r_idx]:
                        packet_lost = True
                elif r_idx == recoder_pos:
                    if not packet_lost and random.random() > err_rates[r_idx]:
                        # The recoder get a new packet from sender
                        recoder.read_payload(packet)
                    # The recoder can generate a packet even if no packets are
                    # received
                    packet = recoder.write_payload()
                    packet_lost = False
                else:
                    # Relays after the recoder
                    if random.random() < err_rates[r_idx]:
                        packet_lost = True
                        break

            if random.random() > err_rates[-1] and not packet_lost:
                # Check if packet is lost at the last link
                decoder.read_payload(packet)

        data_out = decoder.copy_from_symbols()
        if not (data_in == data_out):
            raise RuntimeError("data_out is different from data_in")
        recv_num_lst.append(num)

    mean_time_rec = (sum(recv_num_lst) / runs)

    print("Recoder position: %d \t" % recoder_pos +
          "Average receive number: "+"%5.2f" % (mean_time_rec))


if __name__ == "__main__":
    num_relays = 5
    # Error rates of each link, can be different
    # MARK: len(err_rates) == num_relays + 1
    err_rates = [0.5] * (num_relays + 1)

    print("# Number of relays: %d" % num_relays)
    print("# Link loss rates: %s" % ", ".join(map(str, err_rates)))

    for pos in range(0, num_relays):
        benchmark(num_relays, err_rates, pos)
```