Virtual Network Embedding

Assignment of practical lecture

Problem statement

- You have to maximize your profit
  - Given a substrate network G(V,E)
  - And a sequence of request graphs
    - Embedding of a request graph yields some income
    - But it also has some cost
- Substrate G(V,E):
  - Every node $n \in V$ has a finite capacity c(n)
  - Every edge $e \in E$ has an infinite capacity
- Cost of embedding:
  - Virtual links are routed on a shortest path between their endpoints
  - The cost of the embedding is the sum of the length of these paths
- Dealing with requests
  - Requests arrive one at a time (in an online fashion)
  - Your function has to decide to serve it or to reject it
    - If served, a mapping of the virtual nodes to the substrate nodes has to be specified
  - Each request has an associated time interval,
  - Once a request is embedded, it cannot be enforced to leave the system before its time is up

Problem statement

- event_gen.py :
  - For t=1, …, (n=2000), it returns a random graph G that is either
    - A path of length 2 (adjacency matrix A=[[0,1,0],[1,0,1],[0,1,0]])
    - A clique on 4 points (A=[[0,1,1,1],[1,0,1,1],[1,1,0,1],[1,1,1,0]])
  - If served, G
    - lasts for dur = 20 * rnd.random()
    - Its nodes have a performance value of nodes_perf = rnd.choice(range(1,4),G.n)
    - It means an income of val = 30*len(G.E) * rnd.random()
  - The Graph and Claim classes are described in VNE_classes.py
- keret.py:
  - It runs the simulation on the strategies found in the Strats folder (your task is to come up a strategy that gathers as much money as you can)
  - The host topology is two 4 long circles connected by and edge