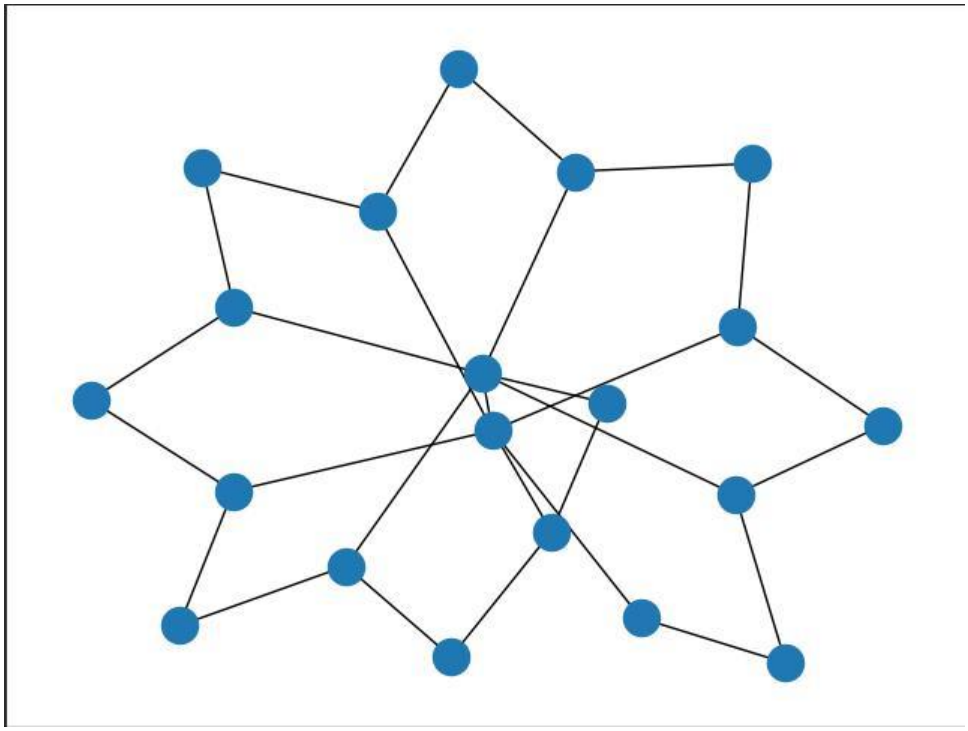
Technologie Sieciowe

Sprawozdanie Lista 2

Proponowana topologia grafa G=<V,E> |V| = 20, |E| = 30

Żaden z wierzchołków grafu nie jest izolowany



Implementacja w kodzie:

```
C = nx.cycle_graph(20)

C1 = nx.Graph(C)
for i in range(1, 10):
    C1.add_edge(i * 2, i % 2)
nx.set_edge_attributes(C1, 0.95, 'p')
nx.set_edge_attributes(C1, 0, 'a')
nx.set_edge_attributes(C1, 1500, 'c')
```

Atrybuty:

c - przepustowośc domyślnie jest ustawiona na 1500 bitów

a – faktyczna liczba pakietów

P - prawdopodobieńswto tego,że krawiędż działa poprawnie

Macierz natężeń jest definiowana losowo wartościami [0,30]

```
# N def
SIZE = P.number_of_nodes()
N = np.zeros((SIZE, SIZE))
for i in range(SIZE):
    for j in range(SIZE):
        if i == j:
            N[i][j] = 0
        else:
            N[i][j] = rand.randint(0, 30)
```

Funkcje pomocnicze:

```
# Funkcja opóźnienia T
def delay(Graph, Nmatrix):
    G = Nmatrix.sum()
    return (1 / G) * (
        sum([Graph.get_edge_data(*e).get('a') / (((Graph.get_edge_data(*e).get('c') / 1)- Graph.get_edge_data(*e).get('a')))
            for e in Graph.edges()]))


# a - suma natężeń na najkrótszej ścieżce
def set_a(Graph, Nmatrix):
    nx.set_edge_attributes(Graph, 0, 'a')
    for i, row in enumerate(Nmatrix):
        for j, n in enumerate(row):
            path = nx.shortest_path(Graph, i, j)
            for k in range(len(path) - 1):
                Graph[path[k]][path[k + 1]]['a'] += n
```

Definujemy funkcje **reliability** -liczącą niezawodność sieci zgodnie z opisem w zadaniu.

Przykładowe wywołanie funkcji dla parametrów domyślnych

(c = 1500,p =0.95 ,Tmax = 0.005,N i a – wygenerowane)

```
Przykladowe wywołanie z domyslnymi parametrami:
Succeded in 88.7 %. Average delay: 0.0056344766502760945
Realiability:  81.8489289740699 %

--------------------------------
```

Sprawdźmy,co się stanie jeżeli powiększamy przepustowość (3 razy 1500->1700->1900->2100)

```
Przepustowosc powiekszana o 200 co krok(3 razy):
Succeded in 88.2 %. Average delay: 0.004718589272770731
Realiability:  81.97278911564626 %
Succeded in 91.60000000000001 %. Average delay: 0.003203129079809029
Realiability:  96.50655021834062 %
Succeded in 95.5 %. Average delay: 0.0025201550957184124
Realiability:  97.38219895287958 %
Succeded in 96.2 %. Average delay: 0.0020136557618484615
Realiability:  99.68814968814968 %
--------------------------------
```

Wracamy do parametrów domyślnych i wywołujemy funkcje 5 razy,przy kazdym wywołaniu mnożymy N razy 1.1. Widać że dla każdego kroku niezawodność istotnie się zmniejsza. Dla 5 wywołania niezawodność wynosi 0%.

```
Succeded in 87.2 %. Average delay: 0.00510836776403772
Realiability: dla kroku # 1   =  78.89908256880734 %
--------------------------------
Succeded in 74.2 %. Average delay: 0.0052584493563268 95
Realiability: dla kroku # 2   =  75.47169811320755 %
--------------------------------
Succeded in 61.7 %. Average delay: 0.006809722780564512
Realiability: dla kroku # 3   =  66.45056726094003 %
--------------------------------
Succeded in 45.4 %. Average delay: 0.0068228148008734595
Realiability: dla kroku # 4   =  55.947136563876654 %
--------------------------------
Succeded in 31.1 %. Average delay: 0.0063891356411798885
Realiability: dla kroku # 5   =  0.0 %
--------------------------------
--------------------------------
```

Spróbujmy dodać dodatkowe krawędzie do grafu.

```
C2 = nx.Graph(C1)
for i in range(1, 10):
    C2.add_edge(20-(i * 2), 20-(i % 2))
nx.set_edge_attributes(C2, 0.95, 'p')
nx.set_edge_attributes(C2, 0, 'a')
nx.set_edge_attributes(C2, 1500, 'c')
nx.draw(C2)
plt.show()
set_a(C2, N)
```

Powtarzamy poprzedni test, i otrzymujemy istotnie lepsze wyniki:

```
MyGraph2 test with N*1.1 every round (5 times):
Succeded in 97.7 %. Average delay: 0.003374427244520986
Realiability: dla kroku #  1   =   98.15762538382803 %
--------------------------------
Succeded in 95.5 %. Average delay: 0.003396492302659771
Realiability: dla kroku #  2   =   97.80104712041884 %
--------------------------------
Succeded in 93.10000000000001 %. Average delay: 0.0044500727922570085
Realiability: dla kroku #  3   =   87.0032223415682 %
--------------------------------
Succeded in 75.3 %. Average delay: 0.005160006862458338
Realiability: dla kroku #  4   =   81.27490039840637 %
--------------------------------
Succeded in 65.3 %. Average delay: 0.007032094777849637
Realiability: dla kroku #  5   =   65.23736600306279 %
--------------------------------
--------------------------------
```

Podsumowanie:

Architektura sieci powinna być dobrana odpowiednio do potrzeb. Przemyślaną sieć powinna cechować niezawodna topologia i duża, odpowiednia do potrzeb, przepustowość.

Największą rolę ma przepustowość, dobrze jest gdy jest kilkakrotnie większa od aktualnego przepływu, tak by w razie awarii była w stanie zapewnić dalszą integralność sieci

Ważne jest by połączenia miały jak największą sprawność, ponieważ zniszczona krawędź jest dużym obciążeniem dla sieci