Technologie Sieciowe

Lista 2

Lazarenko Arina

257259

Cel listy:

Stworzyć model sieci :

- Zaproponować jej typologię przy $|V| = 20$, $|E| < 30$, nie ma izolowanych wierzchołków.
- Ustalić macierz natężenia strumienia pakietów.
- Określić funkcję przepustowości i przepływu dla jej krawędzi.

Stworzyć program szacujący niezawodność sieci, a następnie sprawdzić jak zmienia się przy:

- Ustalonej topologii i przepustowościach oraz rosnących wartościach w macierzy natężeń.
- Ustalonej macierzy natężeń i strukturze topologicznej oraz rosnących wartościach przepustowości.
- Ustalonej macierzy natężeń przy dodawaniu nowych krawędzi.

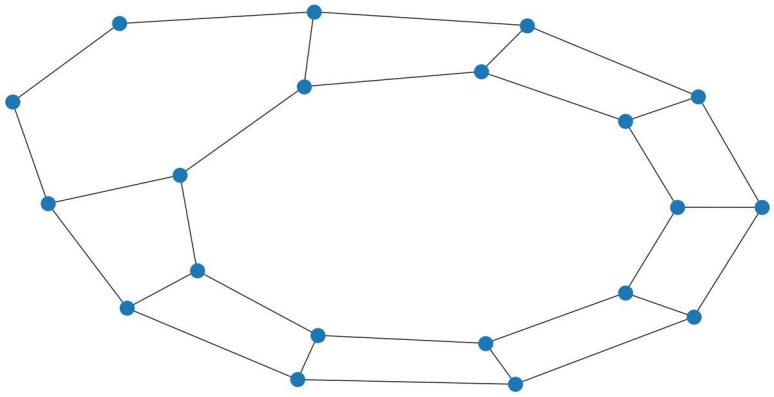
Implementacja

Programy napisane w *Python* z użyciem bibliotek *networkx* i *matplotlib*. Do testowania użyta też funkcja *linspace* z biblioteki *numpy*.

Model sieci

Topologia

Proponowana przeze mnie topologia jest połączeniem dwóch grafów cyklicznych o 9 i 11 węzłach w sposób pokazany na rysunku 1. Jest prosta w stworzeniu i nie ma węzłów izolowanych.



Rysunek 1: Zaproponowana topologia

Macierz natężeń

Macierz natężeń $N = [n(i,j)]$, gdzie $n(i,j)$ jest liczbą wysyłanych z węzła v_i do v_j w ciągu sekundy generujemy losowo, dbając o to żeby $n(i,i) = 0$ oraz $n(i,j) \neq 0$ dla $i \neq j$.

```
N = []
for i in range(20):
    N.append([])
    for j in range(20):
        if i == j:
            N[i].append(0)
        else:
            N[i].append(random.randint(1, 9))
print(np.matrix(N))
```

Rysunek 2: Kod generujący losową macierz

$$\begin{bmatrix}
0 & 2 & 3 & 2 & 9 & 2 & 1 & 6 & 9 & 6 & 3 & 4 & 5 & 7 & 3 & 5 & 2 & 9 & 6 & 9 \\
1 & 0 & 9 & 2 & 8 & 8 & 4 & 5 & 2 & 6 & 6 & 5 & 5 & 2 & 7 & 3 & 8 & 6 & 4 & 3 \\
8 & 1 & 0 & 8 & 5 & 5 & 4 & 1 & 4 & 6 & 2 & 5 & 4 & 7 & 4 & 2 & 9 & 4 & 3 & 3 \\
8 & 5 & 7 & 0 & 4 & 9 & 9 & 2 & 4 & 7 & 3 & 2 & 3 & 7 & 8 & 8 & 9 & 5 & 9 & 7 \\
9 & 1 & 1 & 7 & 0 & 6 & 6 & 9 & 1 & 4 & 1 & 7 & 4 & 5 & 9 & 5 & 8 & 5 & 3 & 5 \\
4 & 4 & 6 & 1 & 9 & 0 & 5 & 6 & 2 & 8 & 2 & 8 & 9 & 6 & 2 & 6 & 4 & 1 & 1 & 3 \\
5 & 4 & 2 & 7 & 8 & 3 & 0 & 8 & 4 & 3 & 6 & 8 & 9 & 7 & 1 & 7 & 2 & 2 & 9 & 3 \\
1 & 1 & 2 & 5 & 4 & 5 & 2 & 0 & 2 & 6 & 3 & 3 & 1 & 5 & 5 & 4 & 6 & 1 & 4 & 9 \\
5 & 5 & 8 & 1 & 9 & 6 & 7 & 2 & 0 & 1 & 4 & 2 & 3 & 9 & 8 & 7 & 2 & 2 & 1 & 6 \\
7 & 4 & 8 & 2 & 3 & 2 & 8 & 1 & 6 & 0 & 7 & 7 & 6 & 4 & 8 & 8 & 6 & 9 & 9 & 6 \\
3 & 6 & 9 & 3 & 1 & 9 & 3 & 2 & 5 & 8 & 0 & 7 & 8 & 7 & 9 & 7 & 3 & 1 & 2 & 4 \\
8 & 1 & 3 & 5 & 4 & 9 & 3 & 5 & 3 & 2 & 7 & 0 & 3 & 7 & 3 & 2 & 6 & 6 & 7 & 4 \\
1 & 5 & 2 & 8 & 5 & 9 & 4 & 8 & 7 & 6 & 7 & 8 & 0 & 9 & 3 & 7 & 3 & 8 & 2 & 8 \\
2 & 8 & 2 & 7 & 5 & 8 & 9 & 5 & 9 & 1 & 4 & 5 & 8 & 0 & 9 & 9 & 4 & 8 & 8 & 7 \\
8 & 6 & 9 & 6 & 6 & 2 & 6 & 5 & 3 & 1 & 9 & 4 & 9 & 5 & 0 & 4 & 2 & 2 & 2 & 3 \\
7 & 3 & 8 & 7 & 4 & 1 & 4 & 5 & 3 & 9 & 9 & 3 & 2 & 7 & 2 & 0 & 7 & 9 & 5 & 6 \\
7 & 7 & 8 & 8 & 7 & 5 & 4 & 4 & 8 & 3 & 5 & 6 & 5 & 1 & 1 & 5 & 0 & 9 & 1 & 6 \\
8 & 7 & 2 & 2 & 3 & 7 & 6 & 9 & 6 & 6 & 3 & 1 & 2 & 9 & 8 & 9 & 2 & 0 & 8 & 9 \\
4 & 9 & 2 & 1 & 7 & 3 & 6 & 2 & 5 & 1 & 7 & 6 & 5 & 7 & 7 & 5 & 4 & 4 & 0 & 8 \\
1 & 9 & 8 & 6 & 4 & 3 & 6 & 1 & 8 & 8 & 5 & 3 & 4 & 1 & 2 & 6 & 4 & 3 & 4 & 0
\end{bmatrix}$$

Rysunek 3: Wygenerowana losowa macierz natężeń

Funkcje krawędzi

Przepływ

Funkcję przepływu, czyli liczbę danych przepływających przez daną krawędź realizujemy wzorem:

$$a(e) = \sum_{v_i, v_j \in V} \left| \{e\} \cap path(v_i, v_j) \right| \times n(i,j)$$

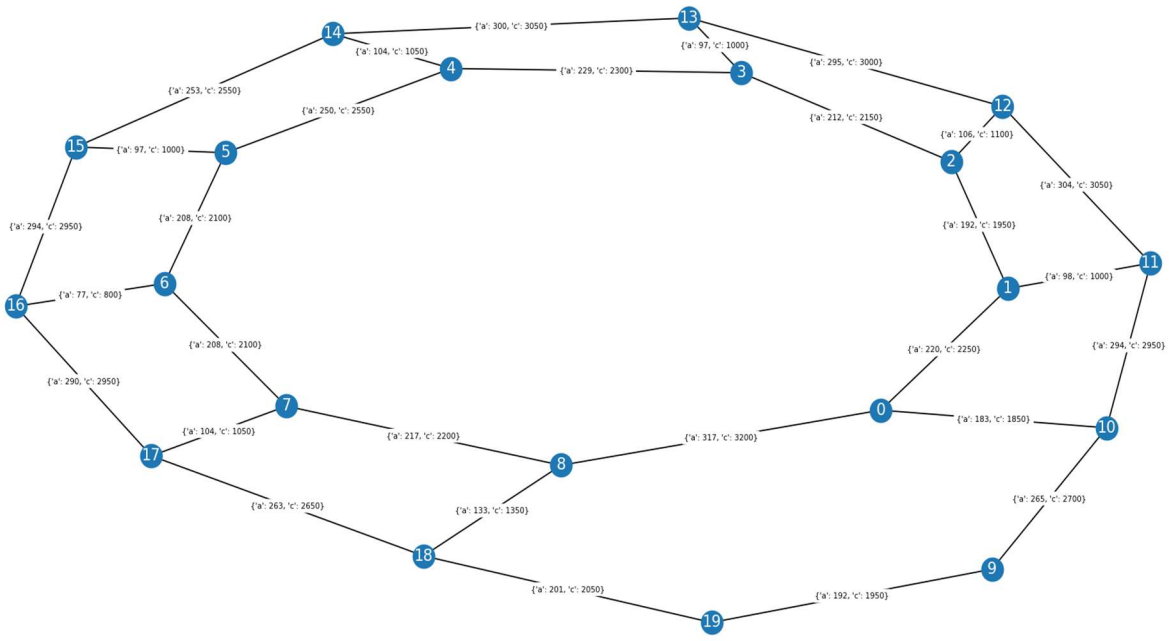
gdzie $path(v_i, v_j)$ jest zbiorem krawędzi znajdujących się na najkrótszej ścieżce z v_i do v_j(może istnieć więcej niż jedna ścieżka, wybieramy zwracaną przez funkcję *shortest_path*).

Przepustowość

Przyjmijmy, że pakiet ma rozmiar kilku kB – oznaczmy ten rozmiar jako S. Wówczas kabel o przepustowości 1 Mb/s w sekundę przepuszcza kilkadziesiąt takich pakietów – na przykład 50. Załóżmy też, że mamy do dyspozycji przewody o przepustowościach będących całkowitą wielokrotnością megabita na sekundę. Zależy nam na jak najtańszym skonstruowaniu naszej sieci, więc dla każdej krawędzi wybieramy kabel o najmniejszej odpowiedniej przepustowości mogącej obsłużyć przepływ dziesięciokrotnie większy niż obecny (przy rozmiarze pakietu S). Zgodnie z tymi założeniami, funkcję przepustowości definiujemy następująco:

$$\mathrm{c(e)} = \left\lfloor \frac{10 \times a(e)}{50} \right\rfloor \times 50 + 50$$

gdzie c(e) = 50n oznacza rzeczywistą przepustowość łącza n Mb/s. Zgodnie z tą umową, kiedy podczas testów podamy średni rozmiar pakietu m = n, mamy na myśli, że w rzeczywistości ma on wartość n · S.



Rysunek 4: Topologia badanej sieci wraz z wartościami funkcji c i a umieszczonymi na krawędziach. Dane zgodne z macierzą natężeń

Niezawodność

Oprócz topologii sieci, macierzy natężeń oraz wartości funkcji a i c, niezawodność zależeć będzie również od następujących parametrów:

- T_{max} – maksymalne opóźnienie pakietu w sieci
- p– prawdopodobieństwo nie uszkodzenia krawędzi w dowolnym interwale
- m – średnia wartość pakietu w bitach

Za miarę niezawodności sieci przyjmujemy prawdopodobieństwo P(T < Tmax). T jest średnim opóźnieniem pakietu w sieci, wyrażanym wzorem:

$$T = \frac{1}{G} \times \sum_{e \in E} \frac{a(e)}{\frac{c(e)}{m} - a(e)}$$

gdzie $G = \sum_{i,j} n(i,j)$.

Niezawodność będziemy więc szacować według następującej procedury:

1. W każdej iteracji rozpoczniemy z wejściową topologią sieci. Iteracja trwać będzie maksymalnie określoną liczbę interwałów.
2. Sprawdzimy, które krawędzie uszkodziły się w obecnym interwale. Jeśli graf został rozspójniony, próba kończy się niepowodzeniem.
3. Biorąc pod uwagę uszkodzenia krawędzi na nowo wyznaczymy wartości funkcji *a*.
4. Spróbujemy obliczyć wartość *T*. Jeśli dla dowolnego *e* otrzymamy $a(e) \geq \frac{c(e)}{m}$, próba kończy się niepowodzeniem (oznacza to, że krawędź została przeciążona).
5. Jeśli uda nam się obliczyć *T* i otrzymamy $T < T_{max}$ uznamy probę za zaliczoną.
6. Wynikiem jednej iteracji będzie liczba udanych prób podzielona przez maksymalny czas jej trwania.
7. Za niezawodność sieci przyjmiemy średnią arytmetyczną wyników wszystkich iteracji.

```python
def T(graph, matrix_sum, m):
    T = 0
    for i, j in graph.edges:
        a = graph[i][j]["a"]
        c = graph[i][j]["c"]
        if a >= c / m:
            return None
        else:
            T += a / (c / m - a)
    return T / matrix_sum


def reliability(graph, matrix, T_max, p, m, iterations=100, intervals=10):
    successful_trials = 0
    matrix_sum = sum(sum(row) for row in matrix)
    base_t = T(graph, matrix_sum, m)
    for _ in range(iterations):
        trial_graph = deepcopy(graph)
        for _ in range(intervals):
            broken = [e for e in nx.edges(trial_graph) if random.random() > p]
            if broken:
                trial_graph.remove_edges_from(broken)
                if not nx.is_connected(trial_graph):
                    break
                assign_flow(trial_graph, matrix)
                t = T(trial_graph, matrix_sum, m)
            else:
                t = base_t
            if not t or t >= T_max:
                break
            successful_trials += 1
    return successful_trials / (iterations * intervals)
```

Rysunek 5: Kody do sprawdzania niezawodności

Ogólne obserwacje

Te obserwacje pokrywają się dla każdego z poniższych wyników:

- Niezawodność rośnie ze wzrostem T_{max}
- Niezawodność rośnie ze wzrostem p
- Niezawodność maleje ze wzrostem m

W niektórych przypadkach zauważyć można odstępstwa od tych reguł, jednak dzieje się tak w większości przy skrajnych wartościach parametrów. Przed testami:

- T_{max} od T bazowej sieci dla obecnego m do dziesięciokrotności tej wartości – poniżej wyniki byłyby bliskie zeru dla dowolnego m
- p od 0.90 do 0.99 – jw.
- m od 1 do 10 – nasza sieć nie jest przygotowana na większe rozmiary, zgodnie ze zdefiniowaną przepustowością.

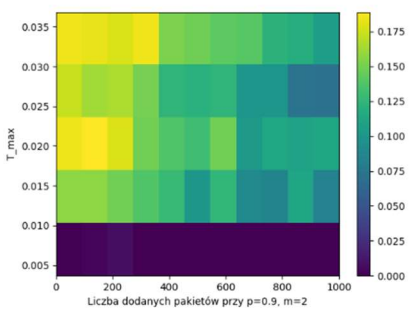
Zwiększenie liczby pakietów a niezawodność

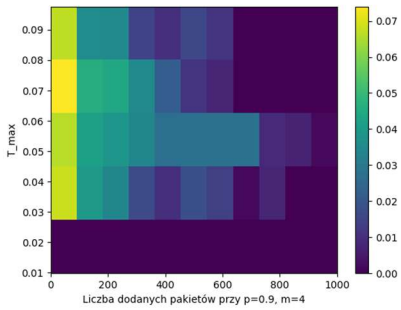
W pierwszym teście sprawdzimy zachowanie niezawodności sieci przy ustalonej topologii i określonej funkcji przepustowości oraz zmieniającej się macierzy natężeń. W każdej iteracji zwiększymy o step losowy element macierzy N (nieleżący na przekątnej), wyznaczymy nowe wartości funkcji a i oszacujemy niezawodność dla macierzy wygenerowanej wcześniej.

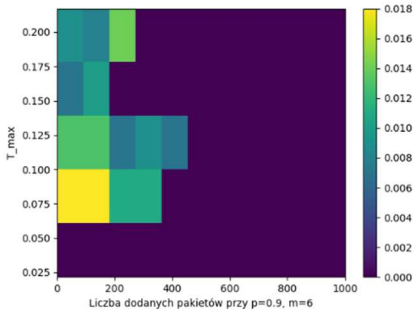
Dodawanie pakietów wydaje się że wpływa negatywnie.

Zmiana T_{max}

Wraz ze wzrostem $\boldsymbol{T_{max}}$ obserwujemy większą tolerancję na dodawanie pakietów, co objawia się na wykresach jako jaśniejszy lewy górny róg.

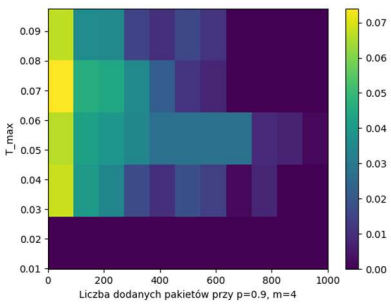


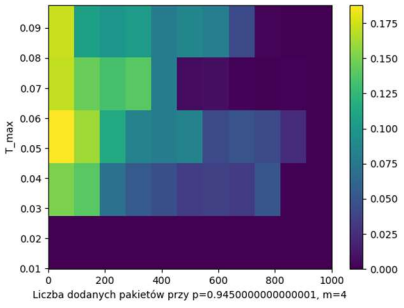


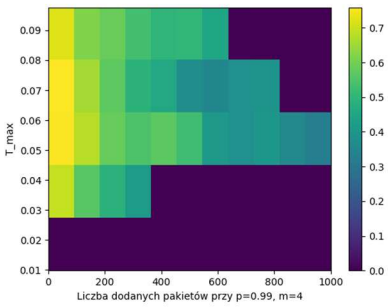


Zmiana p

Dzięki zmianie parametru p możemy wyraźniej zauważyć tendencję wspomnianą wyżej.

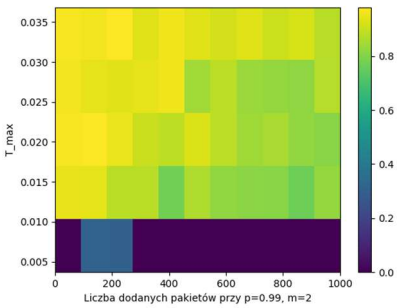


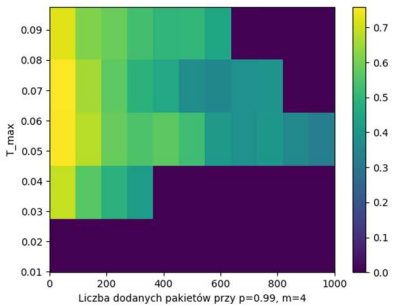


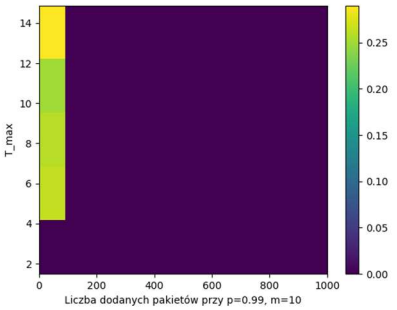


Zmiana m

Wzrost m zwiększa gwałtowność spadku niezawodności przy dodawaniu pakietów.







Zwiększanie przepustowości a niezawodność

Sprawdzimy teraz, jaki wpływ na niezawodność ma wzrost przepustowości połączeń. Będziemy stopniowo zwiększać przepustowość każdego połączenia o 1 Mb/s, czyli wartości funkcji c zmieniać się będą według wzoru:

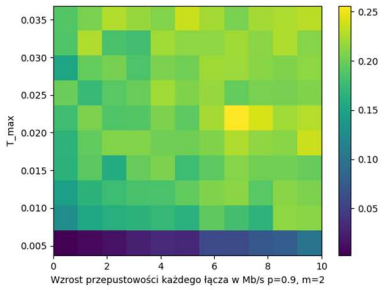
$$\hat{c} = c(e) + 50i$$

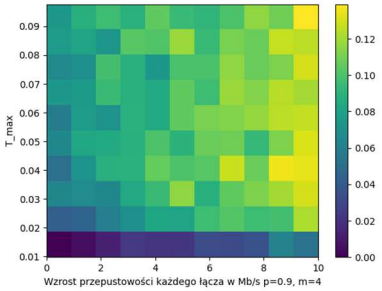
gdzie i to numer iteracji.

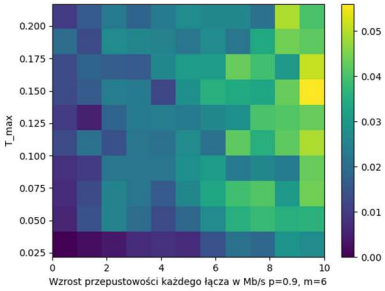
Zwiększenie przepustowośći prowadzi do wzrostu niezawodności.

Zmiana T_{max}

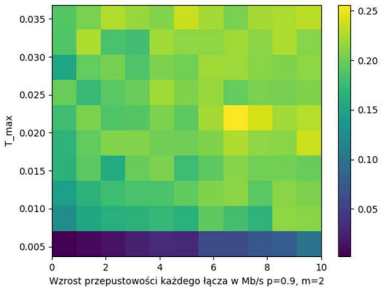
Przy maksymalnie zwiększonej przepustowości, największe wartości niezawodność osiąga zazwyczaj w okolicy najwyższego T_{max}

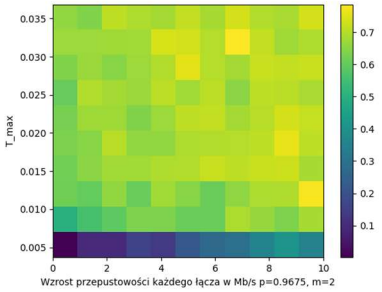


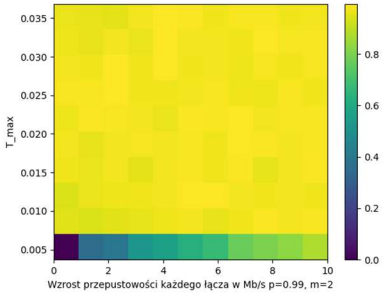




Zmiana p

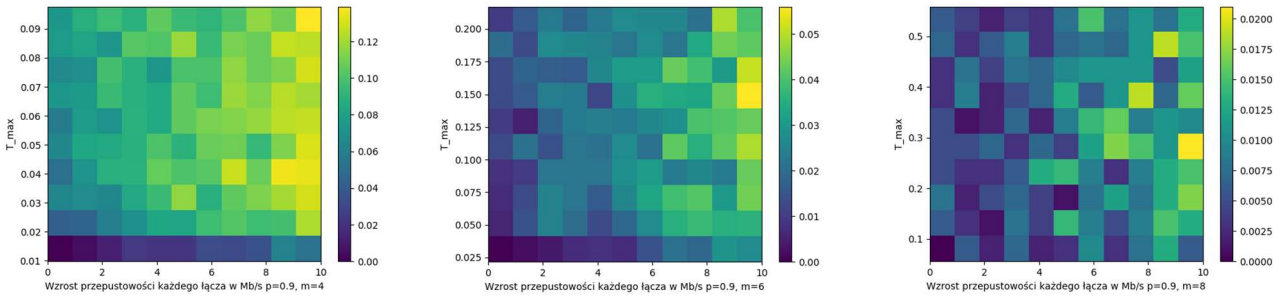






Zmiana m

Zwiększanie m z reguły zmniejsza amplitudę wartości niezawodności, przez co trudniej jest zauważyć zaobserwowaną wcześniej tendencję.



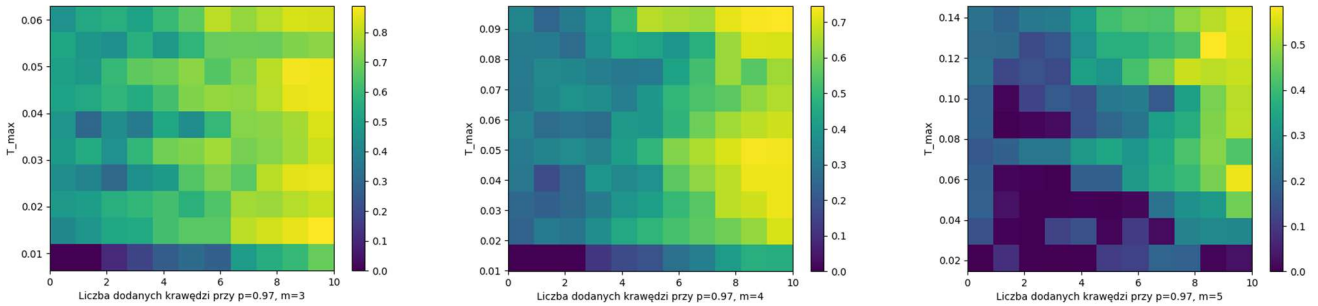
Dodawanie krawędzi a niezawodność

W ostatnim teście sprawdzimy, jak dodawanie krawędzi wpływa na niezawodność. Nowe połączenia będą miały przepustowość równą średniemu c dla sieci początkowej.

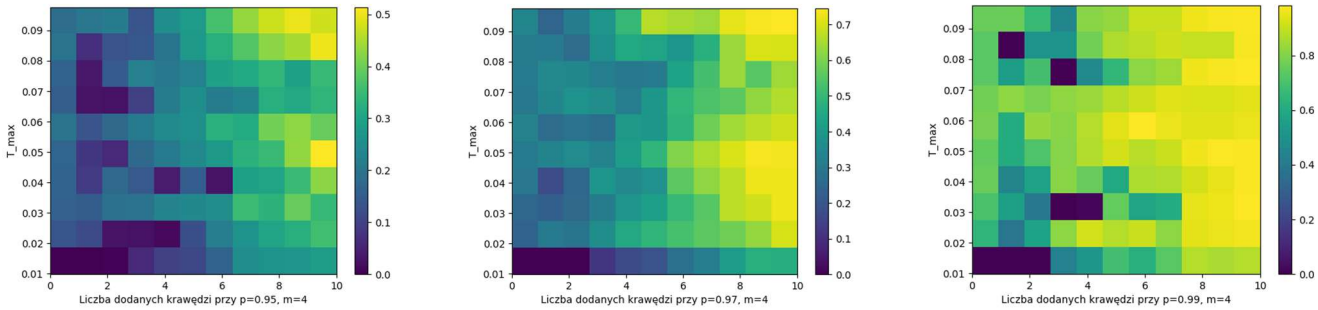
Dane pokazują że przy dodawaniu nowych krawędzi niezawodność rośnie. W wielu przypadkach dodanie 10 krawędzi o wiele poprawiało niezawodność.

Zmiana T_{max}

Największe wartości osiągane są przy największym T_{max} dodaniu największej liczby krawędzi. Niektóre wykresy (jak te pokazane) sugerują, że dodawanie krawędzi niesie za sobą szybszy wzrost niezawodności niż zwiększanie T_{max} (jasny prawy dolny róg).

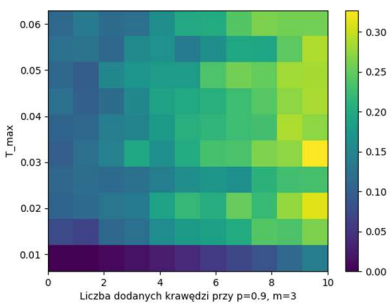


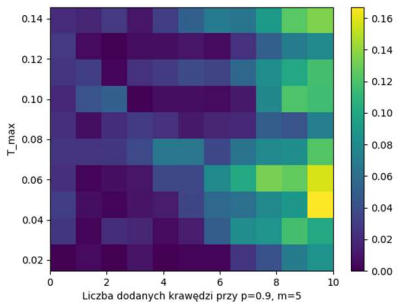
Zmiana p

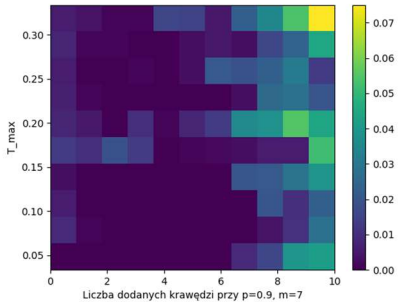


Zmiana m

Czym wyższe m, tym więcej krawędzi musimy dodać do uzyskania dobrych wyników.







Wnioski

- Przy przyjętych wielkościach kroków zwiększanie przepustowości skutkuje bardziej regularnym (chociaż niższym) wzrostem niezawodności niż dodawanie krawędzi.
- Prawdopodobieństwo awarii łącza ma tym większe znaczenie, im bliżej jesteśmy do maksymalnego wykorzystania przepustowości połączeń.
- Trudno stwierdzić, czy większy wpływ na niezawodność ma T_{max} czy p. W przypadku niskich m różnice w testowanych wartościach T_{max} są niewielkie, więc jego zwiększenie jest prawie niezauważalne przy zasadniczo lepszych wynikach. Przy większych rozmiarach pakietów zmniejszenie awaryjności kabli może się okazać mniej uciążliwe.