Technologie sieciowe - lista 2

Należało stworzyć:

- model sieci $S =< G, H >$
- macierz natężeń strumienia pakietów $N = [n_{i,j}]$ - liczba pakietów od źródła $v(i)$ do ujścia $v(j)$

Implementacja symulacji

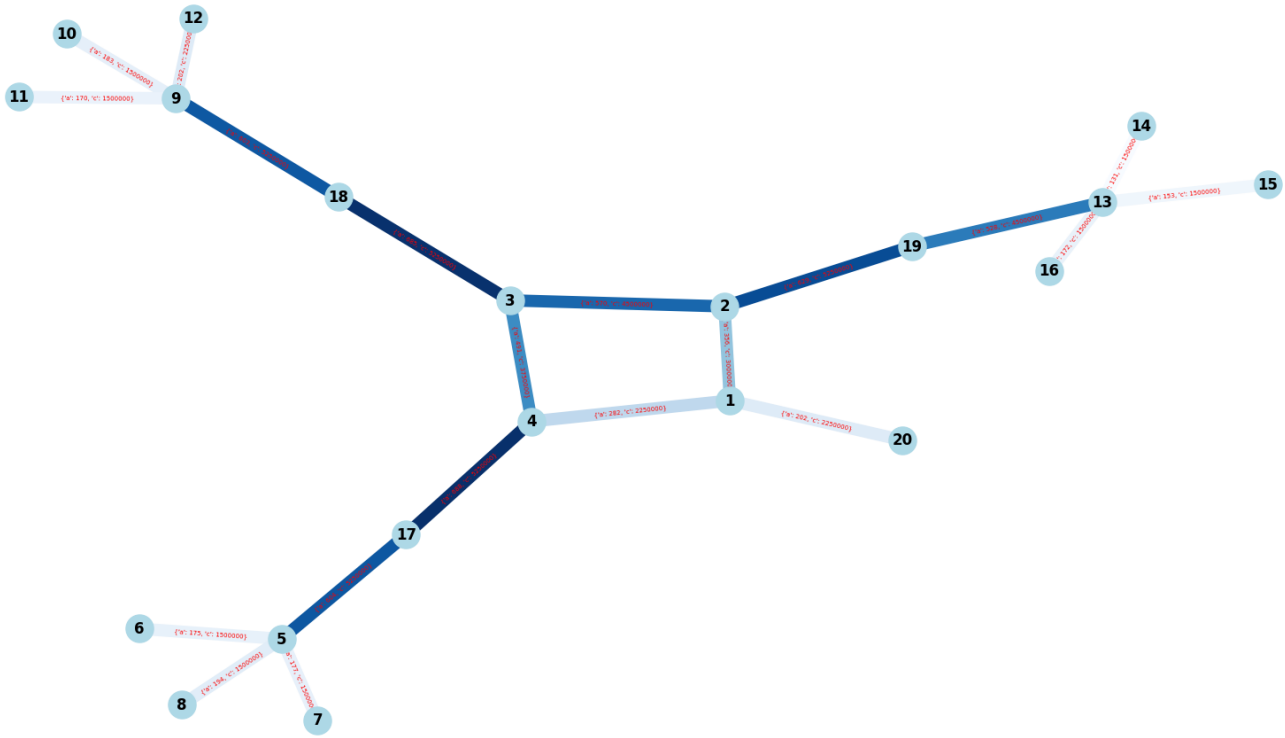
- Użyty język: Python 3.11
- Model sieci opiera się o graf nieskierowany z biblioteki `networkx`.
- Macierz natężeń strumienia N jest zaimplementowana za pomocą `numpy` i `numpy.random`.
- Symulacja odbywa się poprzez ponowne rysowanie grafu wraz z jego atrybutami, parametry symulacji (np. częstość rysowania) można zmieniać poprzez edytowanie pliku `config.py`.
- Do symulacji i rozwiązań zadań użyłem bibliotek `networkx` `matplotlib` `numpy`.
- Za funkcjonalność modyfikowania grafu G odpowiadają funkcje zawarte w `netsim_utils.py`

Z.1. Zaproponuj topologię grafu G tak aby:

- żaden wierzchołek nie był izolowany
- $|V| = 20$ - ilość wierzchołków
- $|E| < 30$ - ilość krawędzi

Model sieci

Topologia proponowanego przeze mnie grafu reprezentującego sieć:



$|G| = 20$, $|E| = 20$. Kolory krawędzi reprezentują obciążenie względem innych gałęzi sieci. Na krawędziach widoczne są wartości funkcji przepustowości (wyrażona w bajtach) i przepływu (wyrażona w ilości pakietów).
Z1.1 Zaproponuj N oraz funkcje:

- $a(e)$ - funkcja przepływu, liczba pakietów które rzeczywiście są wprowadzane do kanału kom. w ciągu 1s
- $c(e)$ - funkcja przepustowości, maksymalna liczba bitów którą można wprowadzić do kanału kom. w ciągu 1s
- $(\forall e \in E)\ c(e) > a(e)$

Macierz natężeń strumienia N

Generowana poprzez wstawienie 20x20 losowych wartości z przedziału $[0, \ldots, MAX\_PCK\_NO]$. Górne ograniczenie na losowe wartości $n_{i,j}$ można edytować w pliku `config.py`. Interpretowana według definicji z polecenia.
Przykładowa macierz N:

```
[[0 2 5 8 5 7 5 5 0 0 1 7 3 6 5 0 4 2 0 5]
[2 0 1 7 6 3 9 9 2 3 0 9 2 7 7 6 4 7 5 2]
[9 8 0 1 6 0 4 9 6 8 7 8 9 0 6 8 2 2 8 8]
[3 8 3 0 8 9 1 9 2 8 0 9 5 2 1 9 2 2 6 8]
[6 3 5 1 0 0 3 1 5 3 9 6 3 1 0 4 6 4 1 5]
[4 0 3 9 5 0 1 4 6 6 6 8 0 5 0 7 7 3 4 4]
[9 9 9 6 8 8 0 2 2 6 6 0 2 3 0 4 0 9 5 8]
[0 9 5 1 2 8 6 0 2 4 4 4 3 6 9 0 5 7 9 4]
[0 4 1 7 3 5 9 0 0 3 7 4 5 1 4 7 4 7 1 2]
[2 7 7 4 4 1 7 9 5 0 1 4 8 3 9 5 7 9 5 2]
[4 5 4 6 2 9 5 4 4 7 0 0 3 3 3 8 2 8 3 8]
[7 3 7 1 3 9 2 8 5 8 4 0 0 6 6 7 1 8 4 6]
[7 9 0 6 7 1 1 5 8 3 9 5 0 4 2 6 6 2 3 0]
[4 4 2 1 5 6 5 1 1 3 5 5 5 0 3 5 0 3 3 6]
[2 1 4 8 1 5 3 7 1 9 1 7 5 0 0 1 6 9 1 7]
[3 2 5 8 4 8 6 5 2 1 5 1 9 2 2 0 1 3 9 8]
[6 8 4 6 8 3 9 7 3 6 3 9 1 1 4 4 0 8 1 8]
[4 2 4 9 7 6 0 4 8 0 4 6 7 3 7 3 1 0 2 7]
[2 8 0 2 2 0 5 8 7 5 2 8 2 2 0 0 8 1 0 7]
[8 2 4 3 2 5 0 9 9 1 8 7 4 9 7 4 9 4 2 0]]
```

Funkcje przepływu i przepustowości

Definicja funkcji przepływu: Suma ilości pakietów, które przepłynęły przez krawędź e. Propagacje pakietów $v(i) \to v(j)$ odbywają się po najkrótszych ścieżkach.

$$a(e) = \sum_{i,j \in |G|} n_{i,j} \ \cdot X_{v(i) \to v(j)}(e)$$

gdzie $X_{v(i) \to v(j)}(e) = \begin{cases} 1 & : e \in v(i) \to v(j) \\ 0 & : p.p. \end{cases}$
Definicja funkcji przepustowości przyjąłem jako:

$$c(e) = (10^r \cdot \lfloor \frac{a(e)}{10^r} \rfloor + 10^r) \cdot 5 \cdot MAX\_PCK\_SIZE$$

gdzie MAX_PCK_SIZE - maksymalna wielkość pakietu w bajtach, edytowalna w `config.py`. Taka definicja powinna zapewnić bezpieczny zakres błędu i umożliwić na wprowadzanie większej ilości pakietów do sieci.

Z2. Zaimplementuj program szacujący niezawodność sieci wg. definicji:

Niezawodność sieci:

Za definicję niezawodności sieci przyjąłem empiryczne przybliżenie prawdopodobieństwa:

$$P(T < T_{max}) = \frac{\sum\limits_{\#testow} \frac{\sum\limits_{s=1}^{\#iteracji} X_{T<T_{max}}}{\#iteracji}}{\#testow}$$

gdzie $X_{T<T_{max}} = \begin{cases} 1 & : T > 0 \land T < T_{max} \\ 0 & : p.p. \end{cases}$
T - średnie opóźnienie pakietów wyrażone wzorem:

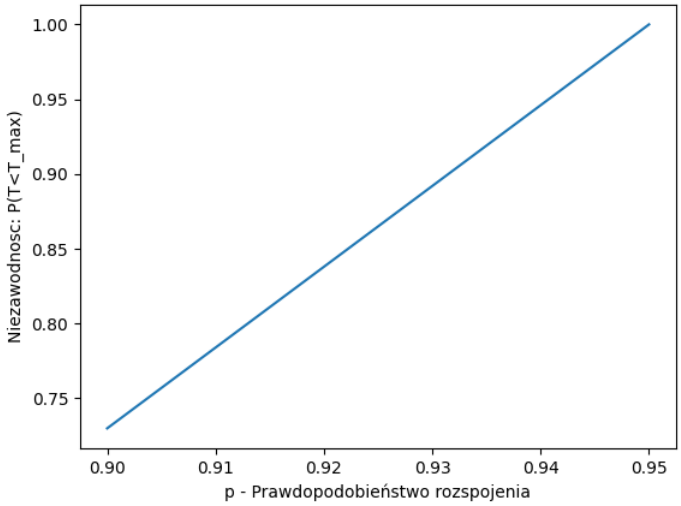
$$T = \frac{1}{G} \sum_{e \in E} \frac{a(e)}{\frac{c(e)}{m} - a(e)}$$

gdzie $G = \sum\limits_{i,j \in |N|} n_{i,j}$
Każdy przypadek testowy sieci jest przerywany w momencie gdy

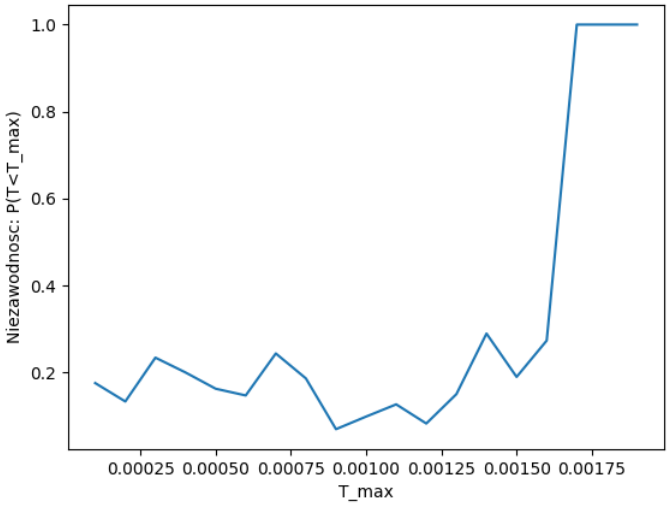
a) pewne połączenie zostanie zerwane z prawdopodobieństwem p (wtedy $T = 0$ implementacyjnie) lub
b) pewne połączenie zostanie przeciążone, czyli $a(e) > c(e)/m$ co można zaobserwować jako ujemną wartość T.
Liczba przypadków testowych w `config.py` jako `N_TESTCASES`

Testy

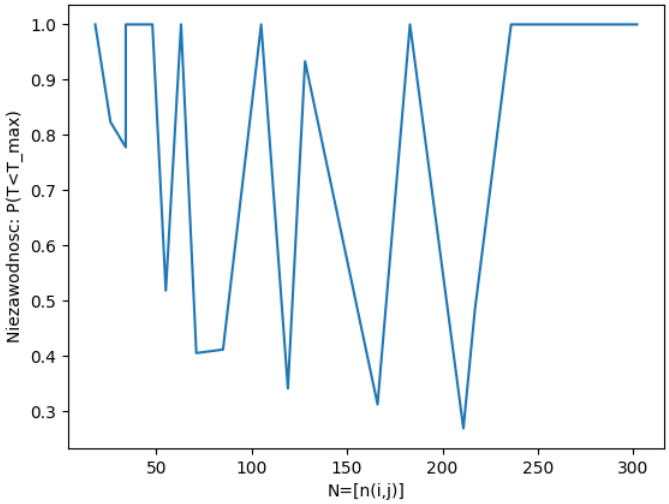
- **Test "statyczny"(nie zmieniam param. grafu):**
  Zależność pomiędzy niezawodnością a prawdopodobieństwem nie rozspojenia p grafu G, dla ustalonych T_{max}, m



  Niezawodność oczywiście będzie zwiększać się wraz z T_{max}:

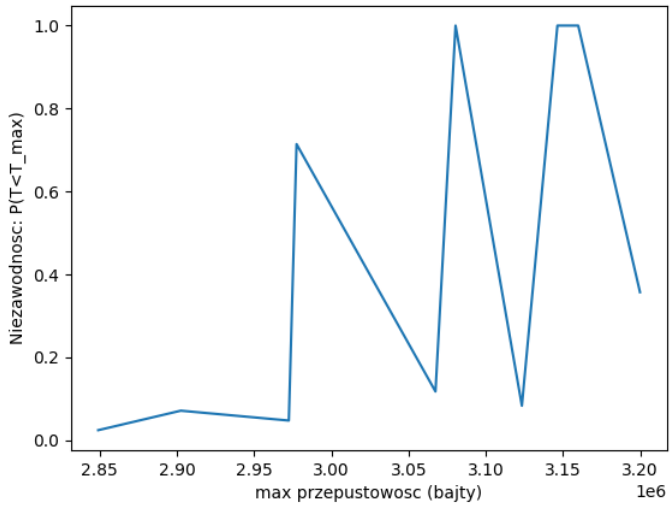


- **Test ze zwiększaniem wartości w N:**
  Zależność między średnią najwiekszą wartością w N a niezawodnością:

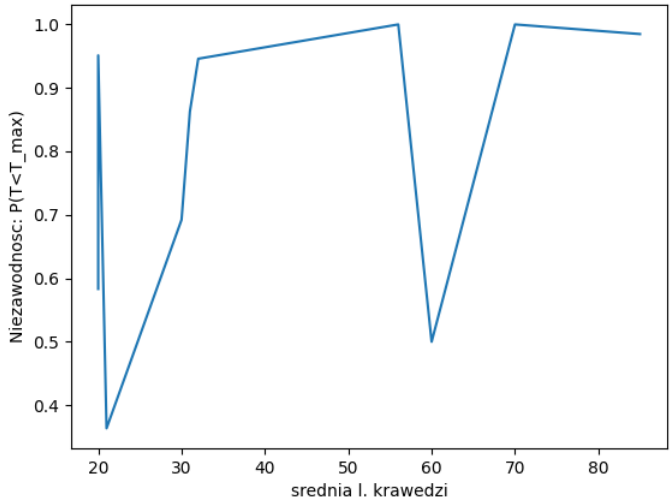


  Można zauważyć spadki w niezawodności w miarę wzrostu wartości w N.

- **Test ze zwiększaniem wartości funkcji przepustowości:**
  Zależność między średnią wartością $c(e)$ a niezawodnością:



- **Test z dodawaniem losowych połączeń do grafu:**
  Zależność między średnią ilością krawędzi a niezawodnością:



Wnioski:

- Obserwacja animacji krawędzi potwierdza dane na wykresie.
- Zwiększanie średniego rozmiaru pakietu lub dodawanie większej ilości pakietów do sieci ma ujemny wpływ na niezawodność.
- Dodawanie krawędzi ma dodatni wpływ na niezawodność, ponieważ pakiety są rozrzedzone w łączach.
- Zwiększenie przepustowości sprawia że sieć jest bardziej niezawodna.