Wizualizacja poissonowskiego strumienia zgłoszeń

Piotr Serafin 132821

Rozkład Poissona

Dyskretny rozkład prawdopodobieństwa, wyrażający prawdopodobieństwo szeregu wydarzeń mających miejsce w określonym czasie, gdy te wydarzenia występują ze znaną średnią częstotliwością i w sposób niezależny od czasu jaki upłynął od ostatniego zajścia takiego zdarzenia. Rozkład Poissona można również stosować w odniesieniu do liczby zdarzeń w innych określonych przedziałach, takich jak odległość, powierzchnia lub objętość.

Jeśli oczekiwaną liczbą zdarzeń w tym przedziale jest λ, to prawdopodobieństwo, że jest dokładnie k wystąpień jest równe:

$$f(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$$

gdzie:

- λ jest dodatnią liczbą rzeczywistą, równą oczekiwanej liczbie zdarzeń w danym przedziale czasu
- k jest liczbą wystąpień zdarzenia

Rozkład Poissona powstaje w związku z procesami Poissona. Ma on zastosowanie do różnych zjawisk dyskretnych właściwości, gdy prawdopodobieństwo wystąpienia zjawiska jest stałe w czasie lub przestrzeni. Zwykłym zastosowaniem rozkładu Poissona jest prognozowanie liczby zdarzeń w danym czasie [1].

In [23]:

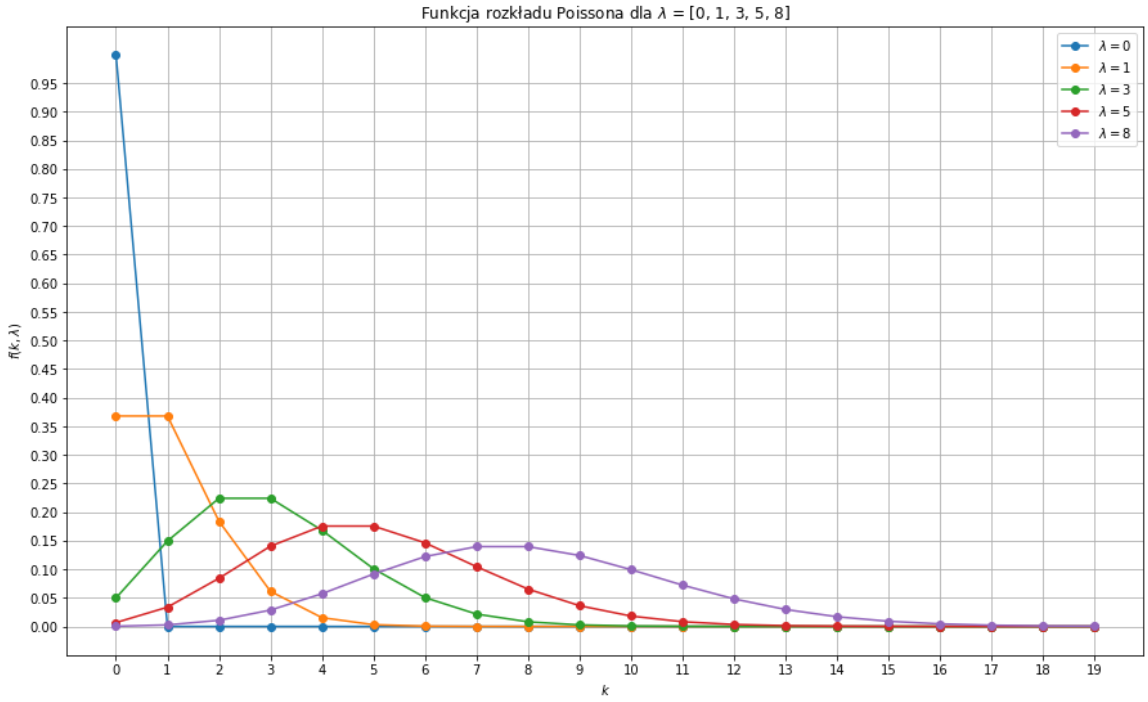
```
# Import bibliotek
import math
import numpy as np
import matplotlib.pyplot as plt

# Widgets
import ipywidgets as widgets
from IPython.display import display

%matplotlib inline

# Zmiana domyślnej wielkości wykreów - aspect ratio 5:3
plt.rcParams['figure.figsize'] = (15, 9)
```

In [24]:

```
# Funkcja zwracająca tablicę prawdopodobieństw dla zadanych parametrów k i lambd
a.
# Suma prawdopodobieństw wynosi 1 (Funkcja masy prawdopodobieństw) (PMF)
def poisson_distribution_pmf(K, lamb, t = 1):
    return np.array([poisson_distribution(k, lamb, t) for k in np.nditer(K)])

# Funkcja rozkładu prawdopodobieństwa (PDF)
def poisson_distribution(k, lamb, t = 1):
    return math.pow(lamb * t, k) * math.exp(-lamb * t) / math.factorial(k)
```

In [25]:

```
# Tablica parametrów k 0..20
K = np.arange(0, 20, 1)[:, None]

# Tablica dla różnych wartości parametru lambda 0..10
lamb_array = np.arange(0, 10, 1)[:, None]

# Macierz prawdopodobieństw dla wszystkich kombinacji k, lambda
poisson_array = np.array([poisson_distribution_pmf(K, lamb) for lamb in lamb_arr
ay])

# Alternatywnie funkcja pmf z biblioteki scipy:
# pmf_array = np.array([poisson(mu).pmf(K) for mu in mu_array]).reshape(mu_arra
y.shape[0], K.shape[0])

# Tablica przykładowych wartości lambdy dla których chcemy wyświetlić wykres
lamb_to_plot_array = [0,1,3,5,8]

# Dla wybranych wartości z lamb_to_plot_array rysuj przebieg
for l in lamb_to_plot_array:
    plt.plot(K, poisson_array[l, :], '-o', label='$\lambda = {}$'.format(int(lam
b_array[l])))

# Opis wykresu
plt.title('Funkcja rozkładu Poissona dla $\lambda$ = {}'.format(lamb_to_plot_arr
ay))
plt.ylabel('$f(k,\lambda)$')
plt.xlabel('$k$')
plt.xticks(K)
plt.yticks(np.arange(0, 1, step=0.05))
plt.grid()
plt.legend()
plt.show()
```


Funkcja rozkładu Poissona dla λ = [0, 1, 3, 5, 8]
λ = 0
λ = 1
λ = 3
λ = 5
λ = 8
f(k, λ)
k
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

Strumień zgłoszeń Poissona

Strumień zgłoszeń opisany jest w pełni przez zbiór reguł określających jednoznacznie proces napływu zgłoszeń do systemu obsługi w określonym przedziale czasu (statystyczny opis procesu przybywania zgłoszeń do systemu obsługi) [2].

Klasyfikacja strumieni zgłoszeń

- regularny jeśli tworzące go zdarzenia pojawiają się w zdeterminowanych przedziałach czasu
- stochastyczny jeśli tworzące go zdarzenia pojawiają się losowo
- jednorodny – charakteryzuje go zawsze jedna cecha
- niejednorodny – każde zgłoszenie ma co najmniej dwie cechy (jedną z nich jest zawsze czas napływu zgłoszenia)

Strumień ruchu opisany rozkładem Poissona jest przypadkiem strumienia prostego [2].

Cechy strumienia prostego

- Stacjonarność – prawdopodobieństwo pojawienia się pewnej liczby zgłoszeń w przedziale czasu zależy od długości tego przedziału, a nie jego położenia na osi czasu
- Brak pamięci - dla dowolnych rozłącznych przedziałów czasu liczba zgłoszeń zachodzących w jednym z nich nie zależy od liczby zgłoszeń zachodzących w pozostałych przedziałach
- Pojedynczość – dwa lub więcej zgłoszeń nie mogą się pojawić w tym samym czasie

Rozkład Poissona bardzo dobrze opisuje jeden strumień zgłoszeń generowany przez bardzo dużą liczbę źródeł ruchu, teoretycznie nieskończenie dużą.

Prawdopodobieństwo napływu k zgłoszeń w przedziale czasu t przy intensywności zgłoszeń λ [2]:

$$P_k(t) = \frac{(\lambda_i t)^k e^{-\lambda t}}{k!}$$

Generowanie strumienia zgłoszeń

Strumień zgłoszeń o rozkładzie Poissona charakteryzuje się wykładniczym rozkładem długości odstępów czasu między zdarzeniami [4].

Załóżmy, że 1200 połączeń wykonywanych jest w ciągu dnia, dlatego:

$$\frac{1200}{24} = 50 \left[\frac{połączeń}{godzinę} \right]$$

zatem:

$$(50/60)/60 \simeq 0.013 \left[\frac{połączeń}{sekundę} \right] \rightarrow \lambda = \frac{1}{0.013} = 72$$

Wynika z tego, że co 72 sekundy napływa zgłoszenie.

Na podstawie tych danych możemy odpowiedzieć na pytanie jakie jest prawdopodobieństwo, że w ciągu nabliższych 20 sekund pojawi się zgłoszenie?

Aby uzyskać odpowiedź na to pytanie należy wykorzystać dystrybuantę rozkładu wykładniczego.

Dystrybuanta rozkładu wykładniczego

Prawdopodobieństwo, że w czasie t nie pojawiło się żadne zgłoszenie, jest równe:

$$P_0(t) = e^{-\lambda t}$$

prawdopodobieństwo pojawienia się jednego zgłoszenia w czasie t wynosi:

$$P_1(t) = \lambda t e^{-\lambda t}$$

Prawdopodobieństwa te zależne są od długości przedziału czasu t. Nie zależą one od czasu pojawienia się poprzedniego zgłoszenia. Na podstawie wzorów na $P_0(t)$ i $P_1(t)$, można obliczyć dystrybuantę $F(t)$ pomiędzy kolejnymi zgłoszeniami, która określa prawdopodobieństwo, że czas pomiędzy zgłoszeniami T będzie krótszy od zadanego czasu t. Prawdopodobieństwo takie jest równoważne prawdopodobieństwu, że w czasie t pojawi się jedno lub więcej zgłoszeń klasy [3]:

$$F(t) = \sum^{\infty} P_n(t) = 1 - P_0(t) = 1 - e^{-\lambda t}$$

In [26]:

```
# Funkcja rozkładu prawdopodobieństwa. Dystrybuanta. (CDF)
def cummulative_distribution(mu, t = 1):
    return (1 - math.exp(-mu * t))
```

In [27]:

```
# Okno 5 minut (300s)
t_array = np.arange(0, 300, 1)

# Paramemtr lambda określony na podstawie zadanej intensywności zgłoszeń
lamb = 0

style = {'description_width': 'initial'}

# Intensywność
intensity = widgets.IntText(min=100, max=2400, value=1200, step=10,
                            continous_update=False, description='Intensywność:',
                            style=style)

def update_cdf_plot(intensity):

    # Lambda/s
    global lamb
    lamb = ((intensity/24)/60/60)

    cdf_array = np.array([cummulative_distribution(lamb, t) for t in t_array])

    plt.plot(t_array, cdf_array, '-o', label='Intensywność {} [zgł/dzień]'.forma
t(float(intensity)))

    plt.title('Dystrybuanta rozkładu wykładniczego')
    plt.ylabel('Prawdopodobieństwo P(t)')
    plt.xlabel('Czas [s]')
    plt.xticks(np.arange(0, 300, step=20))
    plt.yticks(np.arange(0, 1, step=0.05))
    plt.grid()
    plt.legend()
    plt.show()

widgets.interactive(update_cdf_plot, intensity=intensity)
```

Generownie czasu zgłoszeń

Następnym krokiem w symulacji jest wygenerowanie czasu pojawienia się zgłoszenia w systemie. Czas ten powinien zgadzać się z rozkładem wykładniczym długości odstępów czasu między zdarzeniami. Aby to zrobić, użyjemy generatora liczb pseudolosowych między 0 i 1. Należy znaleźć funkcję odwrotną do funkcji dystrybuanty rozkładu wykładniczego [4].

Na podstawie dystrybuanty $F(t)$ można zamienić próbki z generatora o rozkładzie równomiernym na próbki o rozkładzie opisanym przez dystrybuantę $F(t)$. W tym celu próbki otrzymane z rozkładu równomiernego o przedziale od 0 do 1 oznaczone jako U należy podstawić do funkcji dystrybuanty:

$$F(t) = U$$

Powyższy wzór można tak przekształcić, by na podstawie znajomości funkcji $F(t)$ rozkładu Poissona obliczyć t, które jest próbką z generatora losowegoo rozkładzie Poissona [3]:

$$F(t) = 1 - e^{-\lambda t}$$
$$e^{-\lambda t} = 1 - F(t)$$
$$-\lambda t = \ln(1 - F(t))$$
$$t = \frac{-\ln(1 - F(t))}{\lambda}$$
$$t = \frac{-\ln U}{\lambda}$$

Gdzie U to liczba pseudolosowa z zakresu $[0, 1)$

In [28]:

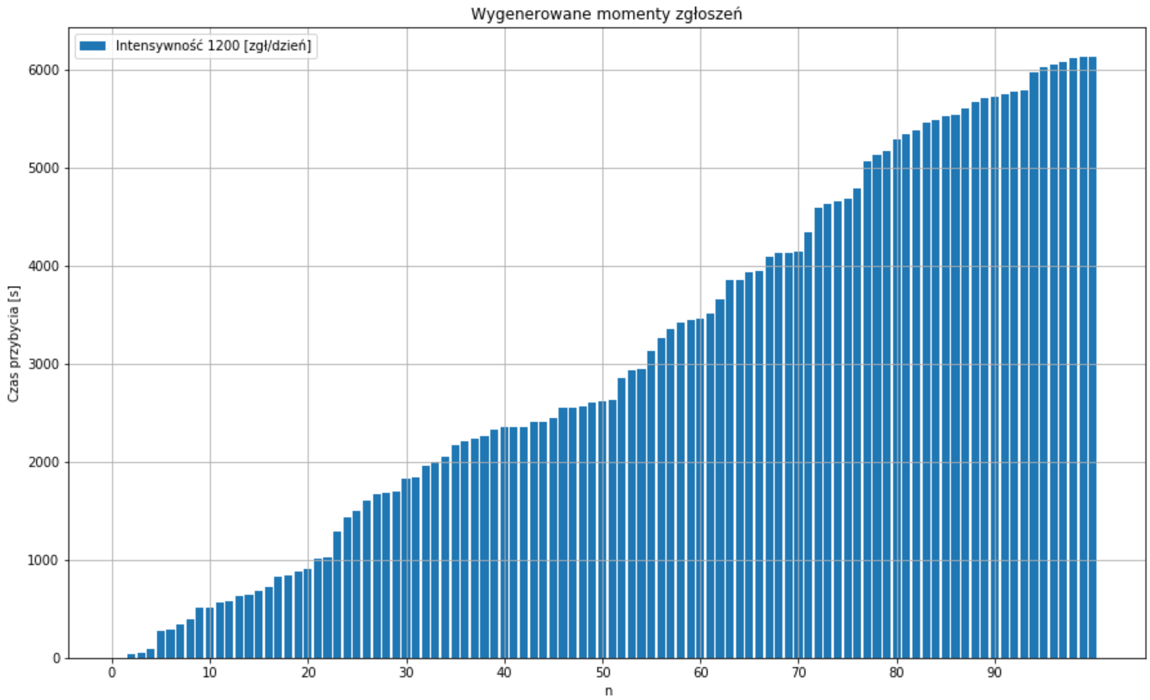
```
def inverse_cummulative_distribution(probability, mu):
    return (-math.log(1 - probability)/mu)
```

In [29]:

```
prob_array = np.arange(0, 1, 0.01)

def update_icdf_plot(intensity):

    icdf_array = np.array([inverse_cummulative_distribution(p,lamb) for p in pro
b_array])

    plt.plot(prob_array, icdf_array, '-o', label='Intensywność {} [zgł/dzień]'.f
ormat(int(intensity)))

    plt.title('Dystrybuanta rozkładu wykładniczego')
    plt.ylabel('Czas [s]')
    plt.xlabel('Prawdopodobieństwo P(t)')
    plt.xticks(np.arange(0, 1, step=0.05))
    plt.yticks(np.arange(0, 300, step=20))
    plt.grid()
    plt.legend()
    plt.show()


widgets.interactive(update_icdf_plot, intensity=intensity)
```

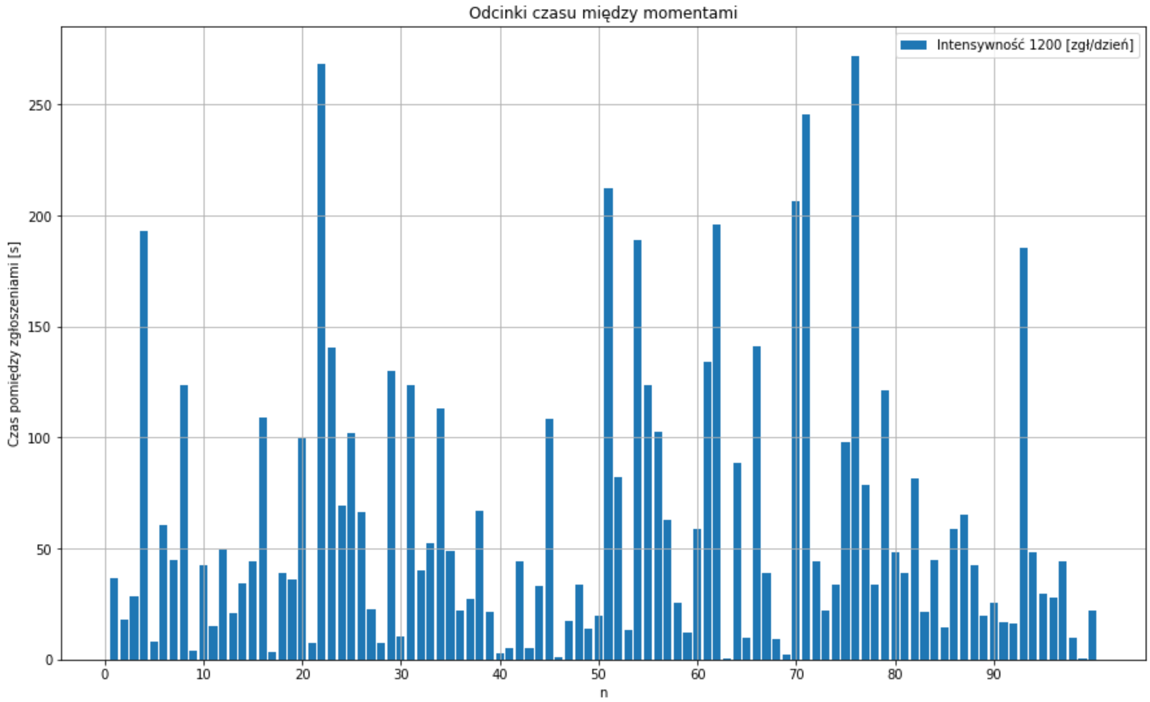
In [30]:

```
moments_arrival_time = [0]
time_differences = []

# 100 przedziałów
indexes = np.arange(1, 101, 1)

for i in indexes:
    # Zwraca pseudo losową liczbę z przedziału [0.0, 1.0)
    random_num = np.random.random()
    time_difference = inverse_cummulative_distribution(random_num, lamb)
    moments_arrival_time.append(time_difference + moments_arrival_time[len(momen
ts_arrival_time) - 1])
    time_differences.append(time_difference)

# Usuwa ostatni moment (out-of-range)
del moments_arrival_time[-1]

plt.bar(indexes, moments_arrival_time, label='Intensywność {} [zgł/dzień]'.forma
t(int(intensity.get_interact_value())))
plt.title('Wygenerowane momenty zgłoszeń')
plt.ylabel('Czas przybycia [s]')
plt.xticks(np.arange(0, 100, step=10))
plt.grid()
plt.xlabel('n')
plt.legend()
plt.show()
```



In [31]:

```
plt.bar(indexes, time_differences, label='Intensywność {} [zgł/dzień]'.format(in
t(intensity.get_interact_value())))
plt.title('Odcinki czasu między momentami')
plt.ylabel('Czas pomiędzy zgłoszeniami [s]')
plt.xticks(np.arange(0, 100, step=10))
plt.grid()
plt.xlabel('n')
plt.legend()
plt.show()
```



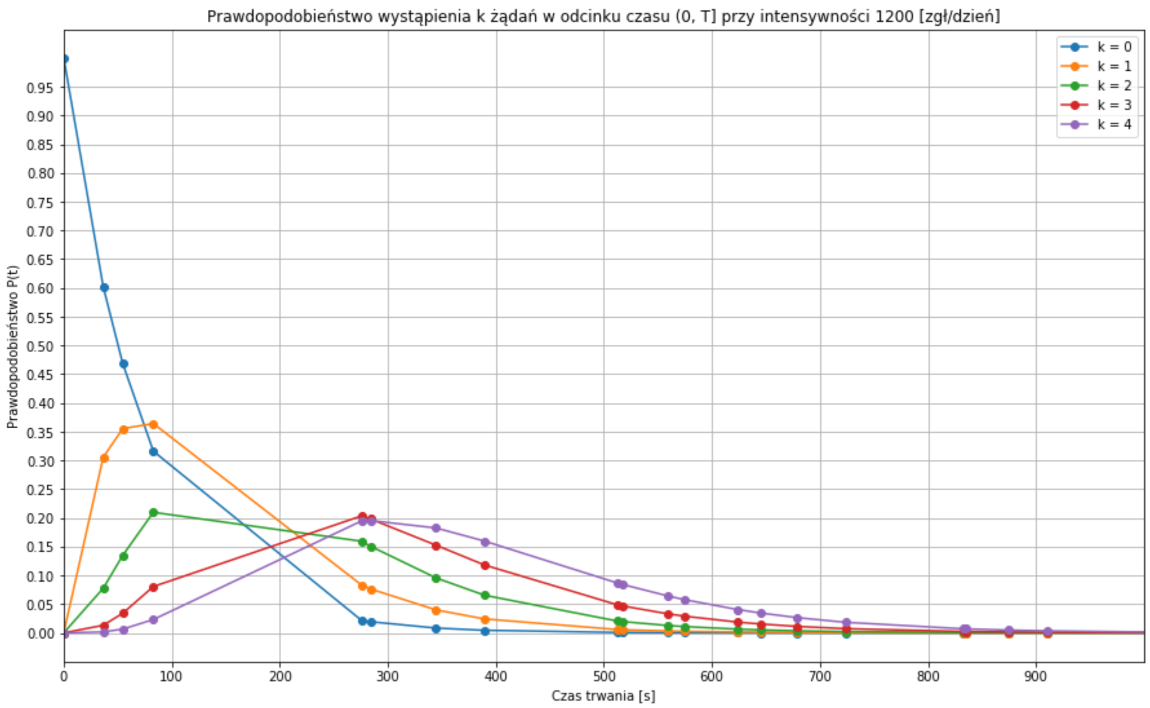
Prawdopodobieństwo wystąpienia k żądań w odcinku czasu [0, t]

Ostatnim krokiem symulacji jest wyliczenie prawdopodobieństwa napływu k zgłoszeń w wygenerowanych przedziałach. Korzystamy z ze wzoru definiującego rozkład Poissona. Prawdopodobieństwo napływu k zgłoszeń w przedziale czasu t przy intensywności zgłoszeń λ [4]

$$P(t) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}$$

In [32]:

```
k_calls_array = np.arange(0,5,1)[:, None]

# Wykorzystujemy losowe momenty do wygenerowania strumenia zgłoszeń o rozkładzie
 Poissona
final_array = np.array([poisson_distribution_pmf(k_calls_array, lamb, t) for t i
n moments_arrival_time])

for k in k_calls_array:
    plt.plot(moments_arrival_time, final_array[:, k], '-o',label='k = {}'.format
(int(k_calls_array[k])))

plt.title('Prawdopodobieństwo wystąpienia k żądań w odcinku czasu (0, T] przy in
tensywności {} [zgł/dzień]'
          .format(int(intensity.get_interact_value())))
plt.ylabel('Prawdopodobieństwo P(t)')
plt.xlabel('Czas trwania [s]')
plt.xlim(0,1000)
plt.xticks(np.arange(0, 1000, step=100))
plt.yticks(np.arange(0, 1, step=0.05))
plt.grid()
plt.legend()
plt.show()
```



Bibliografia

[1]: Rozkład Poissona. (2018, listopad 19). Wikipedia, wolna encyklopedia. Dostęp 16:20, listopad 23, 2018, Dostępny w Internecie: http://pl.wikipedia.org/w/index.php?title=Rozk%C5%82ad_Poissona&oldid=55076364 (http://pl.wikipedia.org/w/index.php?title=Rozk%C5%82ad_Poissona&oldid=55076364)

[2]: Klink, J. (2011). Inżynieria ruchu (telekomunikacyjnego) [Slajdy PDF].
Dostępny w Internecie: https://pst.pwr.edu.pl/moodle/course/view.php?id=82 (https://pst.pwr.edu.pl/moodle/course/view.php?id=82)

[3]: Kaliszan, A. Głąbowski, M. (2006). Symulator wiązki pełnodostępnej obsługującej zintegrowane nie-poissonowskie strumienie zgłoszeń [PDF].
Dostępny w Internecie: http://www.pwt.et.put.poznan.pl/srv/papers/PWT%202006_3658.pdf (http://www.pwt.et.put.poznan.pl/srv/papers/PWT%202006_3658.pdf)

[4]: Gulowaty, B. Grądalska, M. (2016). Wizualizacja poissonowskiego strumienia zgłoszeń [NodeJS].
Dostępny w Internecie: https://github.com/bgulowaty/Wizualizacja-poissonowskiego-strumienia-zgloszen (https://github.com/bgulowaty/Wizualizacja-poissonowskiego-strumienia-zgloszen)