# Algorytmy grafowe

SORTOWANIE TOPOLOGICZNE DFS I BFS

PAWEŁ KOLEC

**Wstęp**

Sprawozdanie porusza zagadnienie sortowania topologicznego wierzchołków grafu. Skupia się na dwóch metodach: procedurze przechodzenia w głąb (DFS) oraz wszerz (BFS). Analizowane są również dwie reprezentacje maszynowe grafu skierowanego: macierz sąsiedztwa i lista następników. Celem jest przedstawienie zasad działania, implementacji, złożoności czasowej i przestrzennej, a także zalet i ograniczeń tych metod.

W pierwszej części sprawozdania wykorzystałem dwie popularne reprezentacje maszynowe grafu: macierz sąsiedztwa i listę następników.

**1.Macierz sąsiedztwa:**

Macierz sąsiedztwa to dwuwymiarowa tablica o rozmiarze n x n, gdzie n oznacza liczbę wierzchołków w grafie. W macierzy sąsiedztwa wartość na pozycji (i, j) wskazuje, czy istnieje krawędź między wierzchołkami i oraz j. Jeśli istnieje krawędź, to wartość ta może przechowywać dodatkowe informacje, takie jak waga krawędzi.

**Złożoność pamięciowa:**

Macierz sąsiedztwa ma złożoność pamięciową O(n^2), ponieważ przechowuje informacje o każdej możliwej parze wierzchołków.

**Złożoność operacyjna:**

Znalezienie pojedynczej krawędzi: O(1), ponieważ wystarczy odczytać wartość na odpowiedniej pozycji w macierzy.

Znalezienie wszystkich następników wierzchołka: O(n), ponieważ trzeba przejrzeć cały wiersz odpowiadający danemu wierzchołkowi.

**2.Lista następników:**

Lista następników to struktura danych, w której dla każdego wierzchołka przechowywana jest lista jego bezpośrednich następników (czyli wierzchołków, do których istnieje krawędź). Dodatkowo, można przechowywać informacje o wagach krawędzi.

**Złożoność pamięciowa:**

Lista następników ma złożoność pamięciową O(n + m), gdzie n to liczba wierzchołków, a m to liczba krawędzi. Przechowuje się listę następników dla każdego wierzchołka oraz dodatkowe informacje o krawędziach.

**Złożoność operacyjna:**

Znalezienie następnika wierzchołka v: O(n), ponieważ należy przejrzeć listę następników wierzchołka v.

Znalezienie wszystkich następników v: O(n), ponieważ trzeba przejrzeć całą listę następników wierzchołka v.

**Sortowanie topologiczne DFS:**

**Wstęp:**

Sortowanie topologiczne jest algorytmem stosowanym w teorii grafów, który pozwala na uporządkowanie wierzchołków w acyklicznym grafie skierowanym. Jest szczególnie przydatne w sytuacjach, gdzie istnieją zależności pomiędzy elementami, które muszą być spełnione w określonej kolejności.

**Jak działa:**

Sortowanie topologiczne metodą Depth-First Search (DFS) polega na przeszukiwaniu grafu w głąb, zaczynając od dowolnego wierzchołka, a następnie odwiedzając rekurencyjnie wszystkie nieodwiedzone sąsiednie wierzchołki. Po zakończeniu rekurencyjnego przeglądu wszystkich sąsiadów danego wierzchołka, dodajemy go do listy wynikowej.

**Złożoności:**

**Dla macierzy sąsiedztwa:**

Złożoność czasowa: O(V^2) - czas potrzebny na znalezienie wszystkich następników dla danego wierzchołka wynosi O(V), ale musimy to zrobić dla każdego z V wierzchołków. Dla wszystkich krawędzi musimy wykonać operację, więc złożoność wynosi O(V^2).

Złożoność pamięciowa: O(V^2) - macierz sąsiedztwa wymaga pamięci na przechowywanie informacji o wszystkich możliwych połączeniach pomiędzy wierzchołkami, co daje złożoność kwadratową względem liczby wierzchołków.

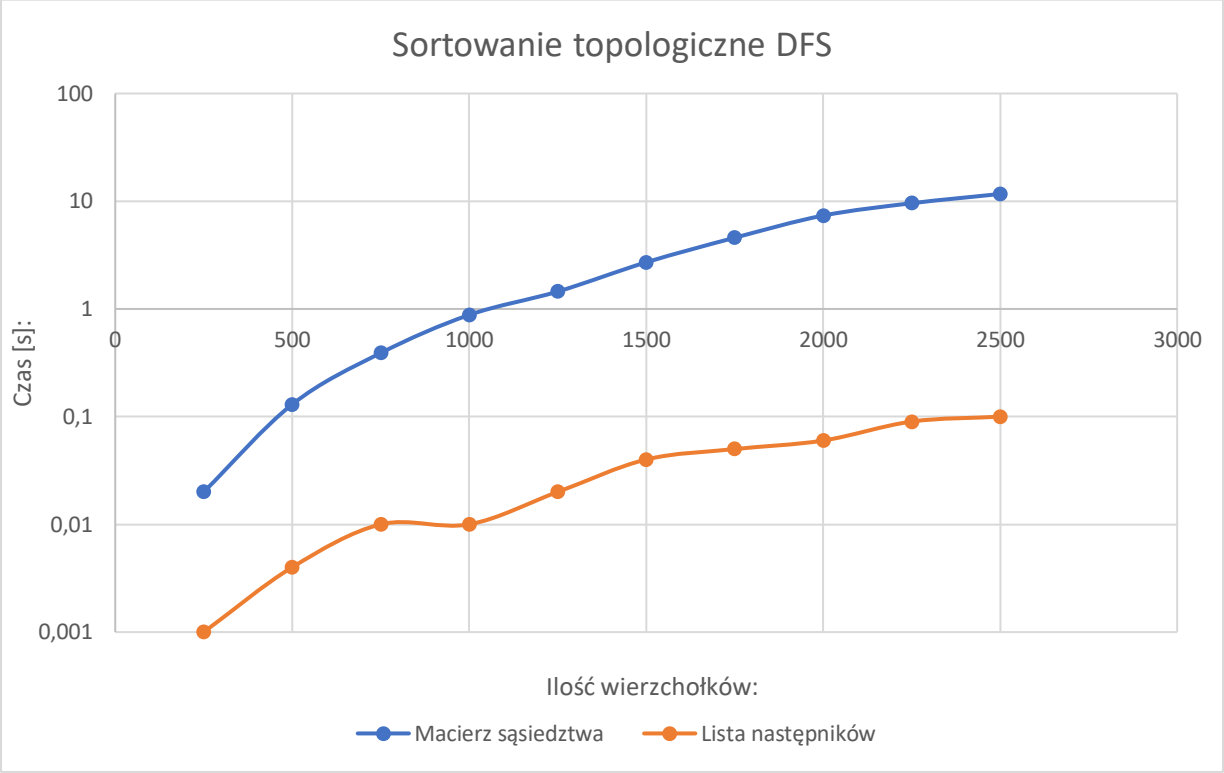
**Dla listy następników:**

Złożoność czasowa: O(V + E) - czas potrzebny na znalezienie wszystkich następników dla wszystkich wierzchołków wynosi O(V). Dodatkowo, musimy przejrzeć wszystkie krawędzie, co zajmuje czas O(E).

Złożoność pamięciowa: O(V + E) - lista następników wymaga pamięci na przechowywanie informacji o każdym wierzchołku oraz jego listy następników, co daje złożoność liniową względem liczby wierzchołków i krawędzi.

Podsumowując, dla sortowania topologicznego DFS lista następników ma złożoność czasową O(V + E), co jest bardziej efektywne niż macierz sąsiedztwa o złożoności O(V^2). Złożoność pamięciowa jest również niższa dla listy następników (O(V + E)) w porównaniu do macierzy sąsiedztwa (O(V^2)). Dlatego lista następników jest lepszym sposobem reprezentacji grafu przy implementacji sortowania topologicznego DFS.

**Zależność czasu obliczeń t od liczby n wierzchołków w sortowaniu topologicznym DFS dla różnych reprezentacji maszynowych grafu:**




**Wnioski:**

Reprezentacja grafu ma istotne znaczenie dla złożoności czasowej i pamięciowej algorytmu sortowania topologicznego DFS. W porównaniu między macierzą sąsiedztwa a listą następników, lista następników okazuje się lepszą reprezentacją pod względem efektywności.

Lista następników jest preferowaną reprezentacją grafu do sortowania topologicznego DFS ze względu na jej lepszą złożoność czasową i pamięciową w porównaniu do macierzy sąsiedztwa. Lista następników jest szczególnie przydatna w przypadku dużych grafów, gdzie efektywność obliczeniowa i zużycie pamięci są ważne. Wybór odpowiedniej reprezentacji grafu ma znaczący wpływ na wydajność i skuteczność algorytmu sortowania topologicznego DFS.

**Sortowanie topologiczne BFS:**

**Jak działa:**

Sortowanie topologiczne BFS rozpoczyna się od wybranego wierzchołka źródłowego. Najpierw odwiedza się wszystkich jego sąsiadów, a następnie przechodzi się do kolejnych wierzchołków, które nie zostały jeszcze odwiedzone. W trakcie przeglądania sąsiadów każdego wierzchołka, zlicza się stopnie wejściowe dla wszystkich wierzchołków. Stopień wejściowy wierzchołka oznacza liczbę krawędzi wchodzących do danego wierzchołka. W wyniku sortowania topologicznego BFS wierzchołki zostają uporządkowane w taki sposób, że żaden wierzchołek nie ma krawędzi skierowanej do wierzchołka wcześniejszego w kolejności.

**Złożoności:**

Porównując złożoności czasowe i pamięciowe dla macierzy sąsiedztwa i listy następników w kontekście sortowania topologicznego, można zauważyć następujące różnice:

**Dla macierzy sąsiedztwa:**

Czasowa: O(V^2), gdzie V to liczba wierzchołków. Musimy przejrzeć całą macierz, aby znaleźć wszystkie następniki dla każdego wierzchołka. To prowadzi do złożoności kwadratowej, która staje się problematyczna dla dużych grafów.

Pamięciowa: O(V^2), gdzie V to liczba wierzchołków. Musimy przechowywać macierz o rozmiarze V*V, co wymaga pamięci proporcjonalnej do kwadratu liczby wierzchołków. Jest to szczególnie istotne dla dużych grafów, gdzie zużycie pamięci może być znaczące.

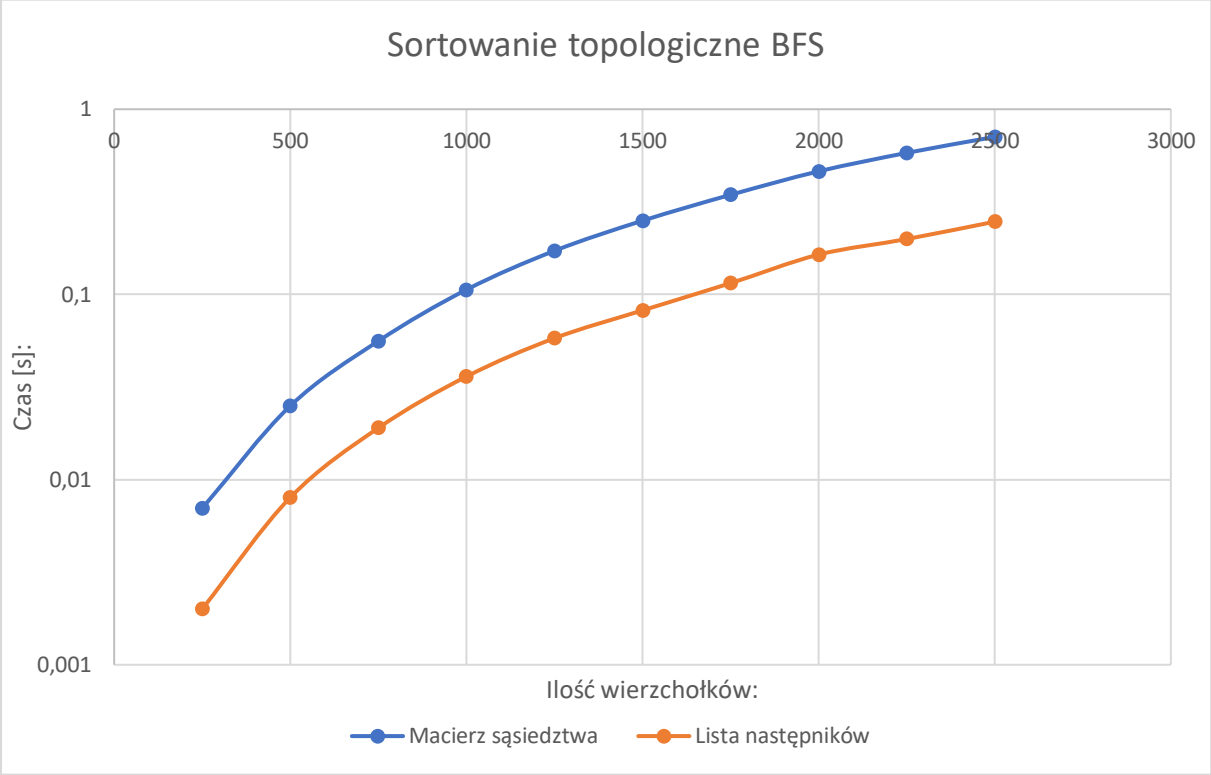
**Dla listy następników:**

Czasowa: O(V + E), gdzie V to liczba wierzchołków, a E to liczba krawędzi. Przechodzimy przez wszystkie wierzchołki i krawędzie, co zajmuje czas proporcjonalny do ich sumy. Ta liniowa złożoność czasowa jest bardziej efektywna dla dużych grafów.

Pamięciowa: O(V + E), gdzie V to liczba wierzchołków, a E to liczba krawędzi. Przechowujemy listy następników dla każdego wierzchołka, co zajmuje pamięć proporcjonalną do sumy liczby wierzchołków i krawędzi. Ta liniowa złożoność pamięciowa jest bardziej optymalna dla dużych grafów.

Podsumowując, lista następników jest lepszą reprezentacją grafu pod względem zarówno złożoności czasowej, jak i pamięciowej dla sortowania topologicznego. Zapewnia ona liniową złożoność czasową i pamięciową (O(V + E)), co jest bardziej efektywne dla dużych grafów. Macierz sąsiedztwa ma złożoność kwadratową zarówno czasową, jak i pamięciową (O(V^2)), co staje się problematyczne dla większych grafów. Dlatego lista następników jest preferowaną reprezentacją grafu przy implementacji sortowania topologicznego BFS.

**Zależność czasu obliczeń t od liczby n wierzchołków w sortowaniu topologicznym BFS dla różnych reprezentacji maszynowych grafu:**

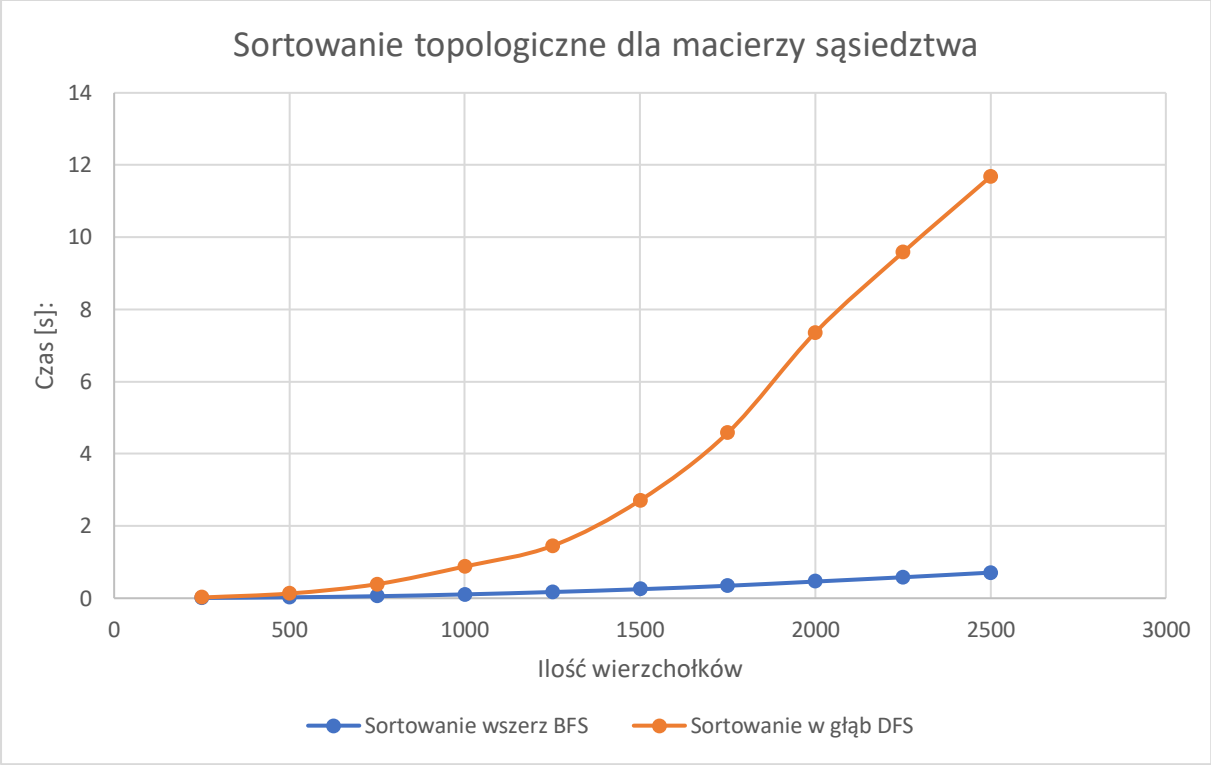



**Wnioski:**

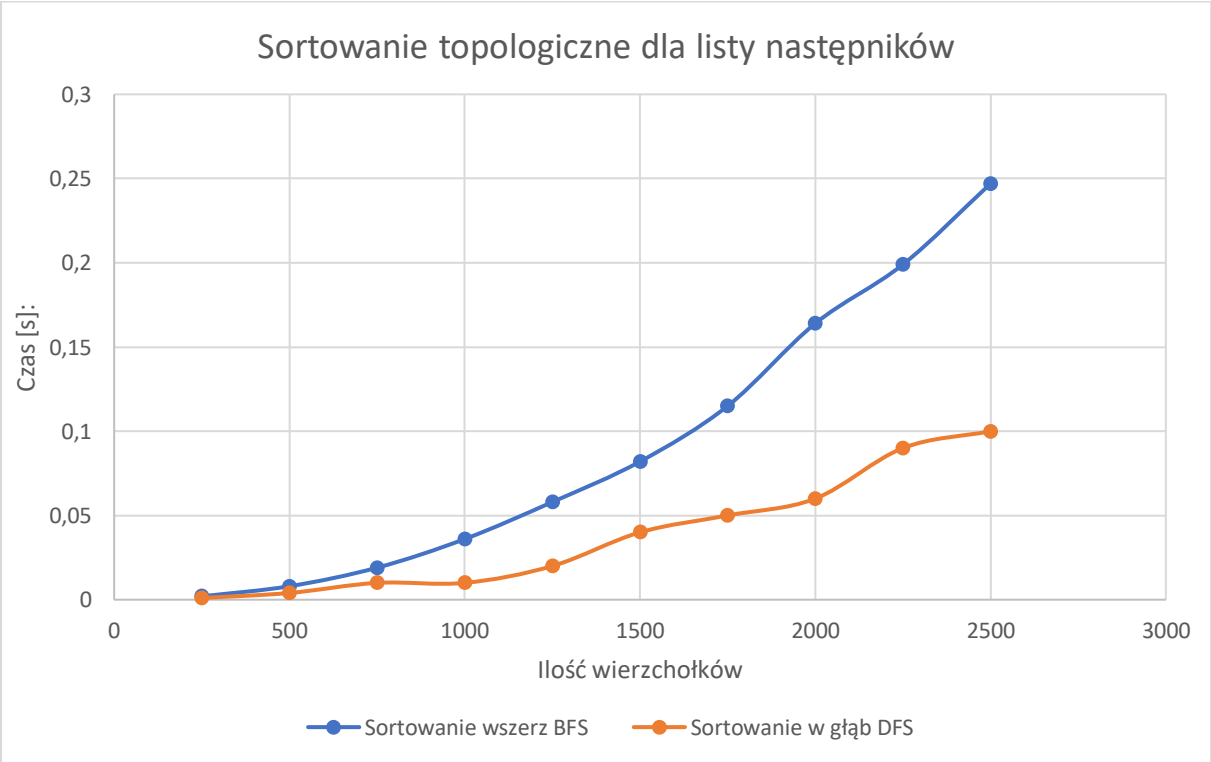
Reprezentacja grafu ma kluczowe znaczenie dla złożoności czasowej i pamięciowej algorytmu sortowania topologicznego BFS. W porównaniu między macierzą sąsiedztwa a listą następników, lista następników jest preferowaną reprezentacją, ze względu na jej lepszą efektywność zarówno czasową, jak i pamięciową. Jest szczególnie korzystna dla dużych grafów, gdzie optymalne wykorzystanie zasobów obliczeniowych i pamięci jest istotne. Wybór właściwej reprezentacji grafu ma istotny wpływ na wydajność i skuteczność algorytmu sortowania topologicznego BFS.

**Porównanie metod sortowania topologicznego w zależności od reprezentacji maszynowej grafu:**

**Macierz sąsiedztwa:**




**Lista następników:**



**Wnioski:**

Porównując metody sortowania topologicznego w zależności od reprezentacji grafu, lista następników jest lepszą opcją niż macierz sąsiedztwa. Dla macierzy sąsiedztwa, BFS działa lepiej niż DFS ze względu na złożoność czasową O(V^2). Dla listy następników, DFS jest bardziej efektywny niż BFS ze względu na złożoność czasową O(V + E). Ogólnie rzecz biorąc, lista następników jest bardziej efektywną reprezentacją grafu dla sortowania topologicznego zarówno przy użyciu DFS, jak i BFS, ze względu na lepszą złożoność czasową i pamięciową.

# Algorytmy z powracaniem

ZNAJDOWANIE CYKLU HAMILTONA I EULERA W GRAFIE

PAWEŁ KOLEC

## Cel:

Celem niniejszego sprawozdania jest przeprowadzenie analizy algorytmów z powracaniem w kontekście poszukiwania cyklu Eulera i cyklu Hamiltona. W szczególności, skoncentruję się na analizie tych algorytmów dla dwóch rodzajów grafów: grafu nieskierowanego z reprezentacją macierzy sąsiedztwa oraz grafu skierowanego z reprezentacją listy następników.

Przedstawię szczegółowo opis algorytmów z powracaniem dla poszukiwania cyklu Eulera i cyklu Hamiltona w grafach nieskierowanych z reprezentacją macierzy sąsiedztwa oraz w grafach skierowanych z reprezentacją listy następników. Przeprowadzę również analizę złożoności czasowej i przestrzennej tych algorytmów oraz porównamy ich efektywność i wydajność dla różnych typów grafów.

Wnioski z przeprowadzonych badań mogą posłużyć do lepszego zrozumienia i wyboru odpowiednich technik rozwiązywania problemów poszukiwania cyklu Eulera i Hamiltona w praktycznych zastosowaniach.

## Jak działa:

Cykl Hamiltona:

Cykl Hamiltona to taki cykl w grafie, który przechodzi przez każdy wierzchołek dokładnie raz i wraca do wierzchołka początkowego. Istnienie cyklu Hamiltona w grafie oznacza, że istnieje zamknięta ścieżka, która odwiedza każdy wierzchołek grafu dokładnie raz.

Algorytm poszukiwania cyklu Hamiltona polega na przeglądaniu wszystkich możliwych permutacji wierzchołków grafu w celu znalezienia cyklu, który spełnia warunek odwiedzenia każdego wierzchołka dokładnie raz. Algorytm wykorzystuje metodę z powracaniem, aby eksplorować różne ścieżki i dokonywać wyborów w przypadku napotkania punktu martwego, czyli sytuacji, gdy dalsze poszukiwanie nie prowadzi do znalezienia rozwiązania.

Cykl Eulera:

Cykl Eulera to taki cykl w grafie, który przechodzi przez każdą krawędź grafu dokładnie raz. W przeciwieństwie do cyklu Hamiltona, cykl Eulera nie musi przechodzić przez każdy wierzchołek grafu.

Algorytm poszukiwania cyklu Eulera również opiera się na metodzie z powracaniem. Początkowo, algorytm wybiera pierwszy wierzchołek grafu jako punkt początkowy. Następnie, wędruje po krawędziach, eliminując je po przejściu. Jeżeli napotka wierzchołek, z którego nie można dalej się poruszać, wraca do poprzedniego wierzchołka, który ma jeszcze dostępne krawędzie, kontynuując eksplorację. Algorytm kończy się, gdy wszystkie krawędzie zostaną odwiedzone.

Podsumowując, poszukiwanie cyklu Hamiltona polega na znajdowaniu zamkniętej ścieżki, która przechodzi przez każdy wierzchołek grafu dokładnie raz, podczas gdy poszukiwanie cyklu Eulera polega na znalezieniu ścieżki, która przechodzi przez każdą krawędź grafu dokładnie raz.

Obie te operacje opierają się na algorytmach z powracaniem, które eksplorują różne ścieżki i wykorzystują informacje o odwiedzonych wierzchołkach lub krawędziach w celu znalezienia rozwiązania.

## Złożoność obliczeniowa:

Klasy złożoności obliczeniowej badanych problemów są związane z liczbą wierzchołków w grafie. Problem poszukiwania cyklu Hamiltona należy do klasy NP-trudnych problemów, co oznacza, że nie istnieje znany algorytm efektywny w czasie wielomianowym dla wszystkich instancji tego problemu.

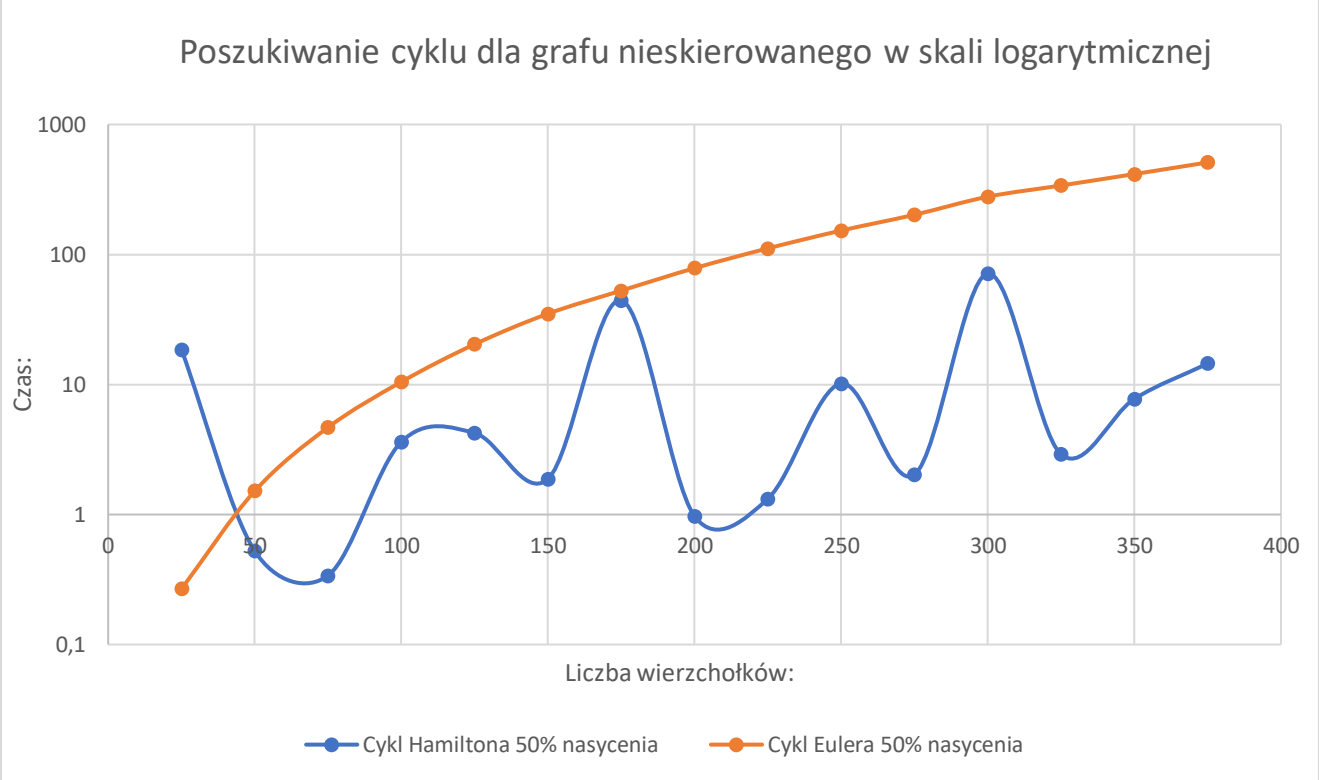
Podobnie jak problem poszukiwania cyklu Hamiltona, problem poszukiwania cyklu Eulera również jest klasyfikowany jako NP-trudny. Oznacza to, że nie istnieje znany algorytm efektywny w czasie wielomianowym dla wszystkich instancji tego problemu.

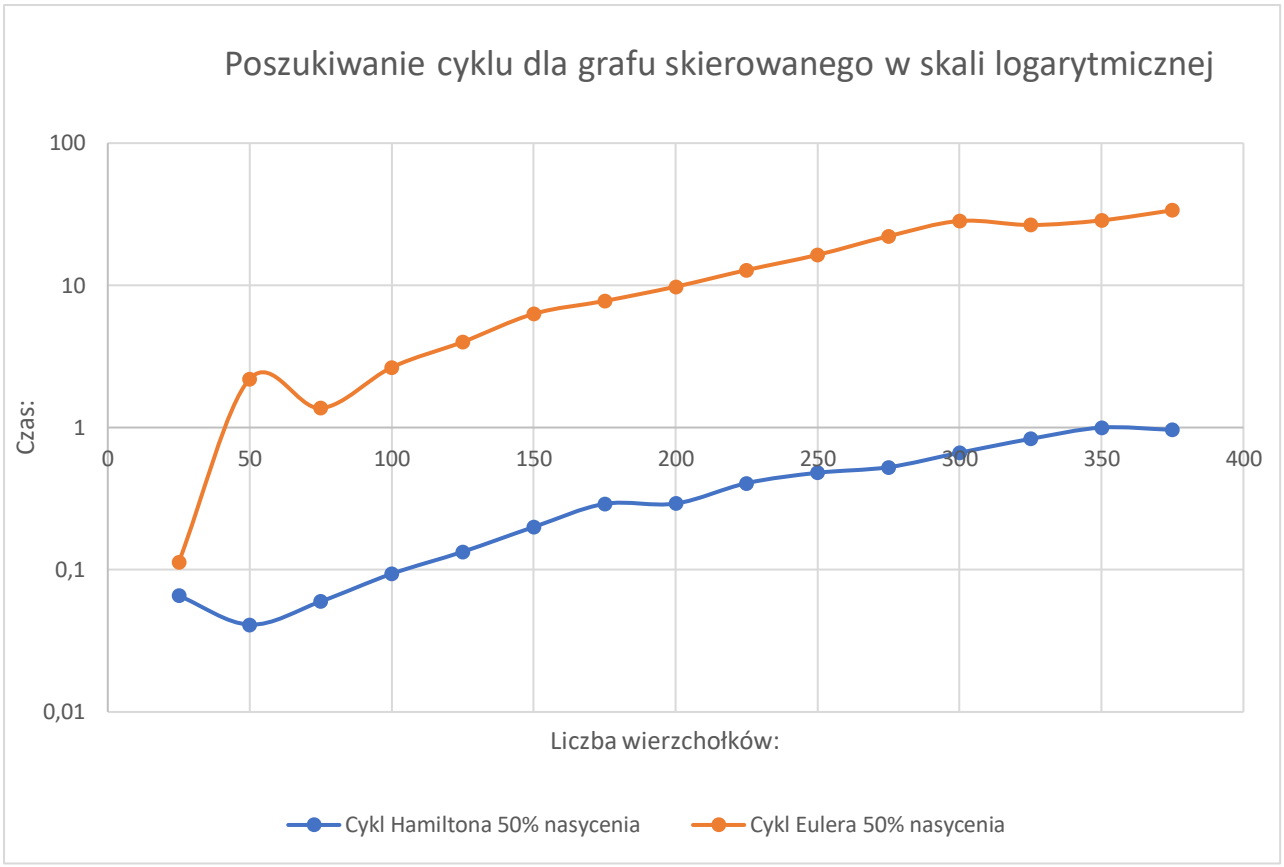
Klasyfikacja NP-trudności oznacza, że nie mamy ogólnie znanego efektywnego sposobu rozwiązania tych problemów, szczególnie dla dużych grafów. Algorytmy z powracaniem są jednym z podejść do rozwiązywania tych problemów, ale złożoność czasowa tych algorytmów rośnie wykładniczo wraz ze wzrostem liczby wierzchołków grafu. Dlatego dla dużych grafów, poszukiwanie cyklu Hamiltona i cyklu Eulera może być praktycznie niemożliwe do wykonania w rozsądnym czasie.

Podsumowując, badane problemy poszukiwania cyklu Hamiltona i cyklu Eulera w grafach, zarówno nieskierowanych jak i skierowanych, są klasyfikowane jako NP-trudne.

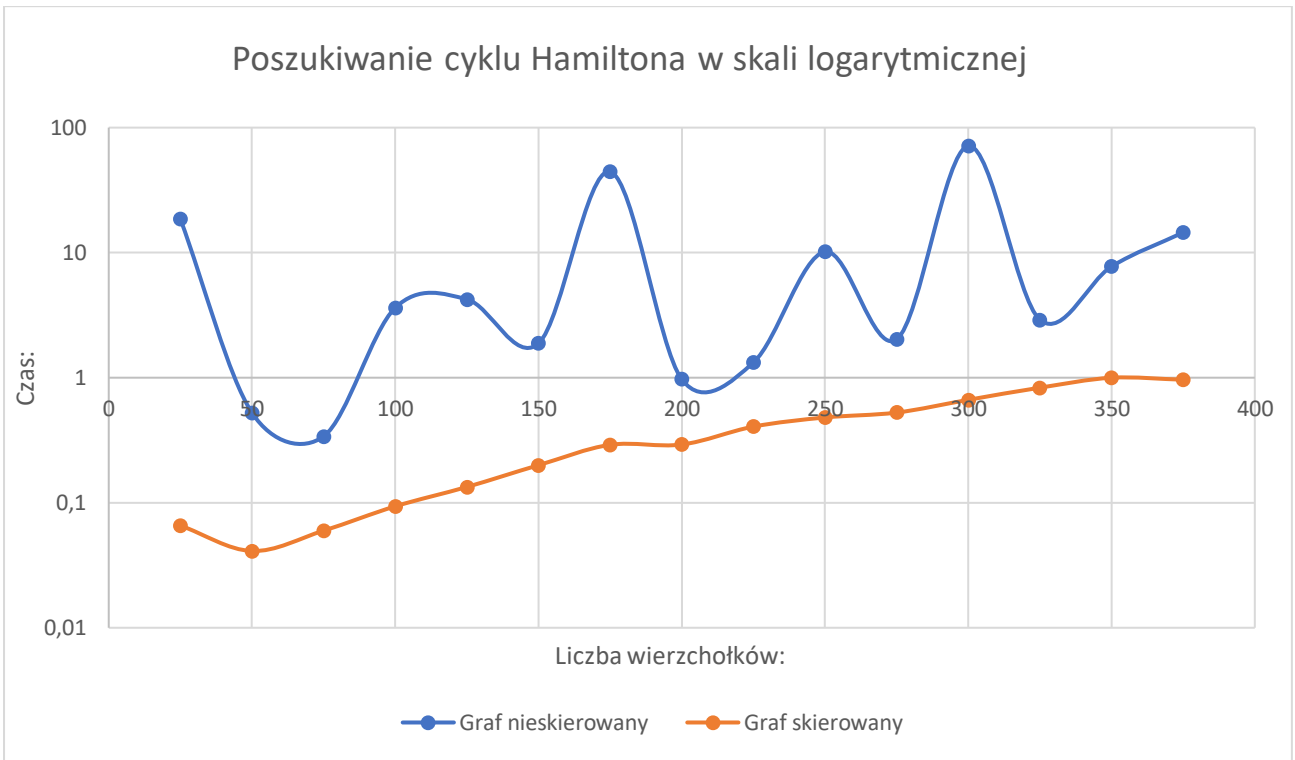
## Wyniki pomiarów:

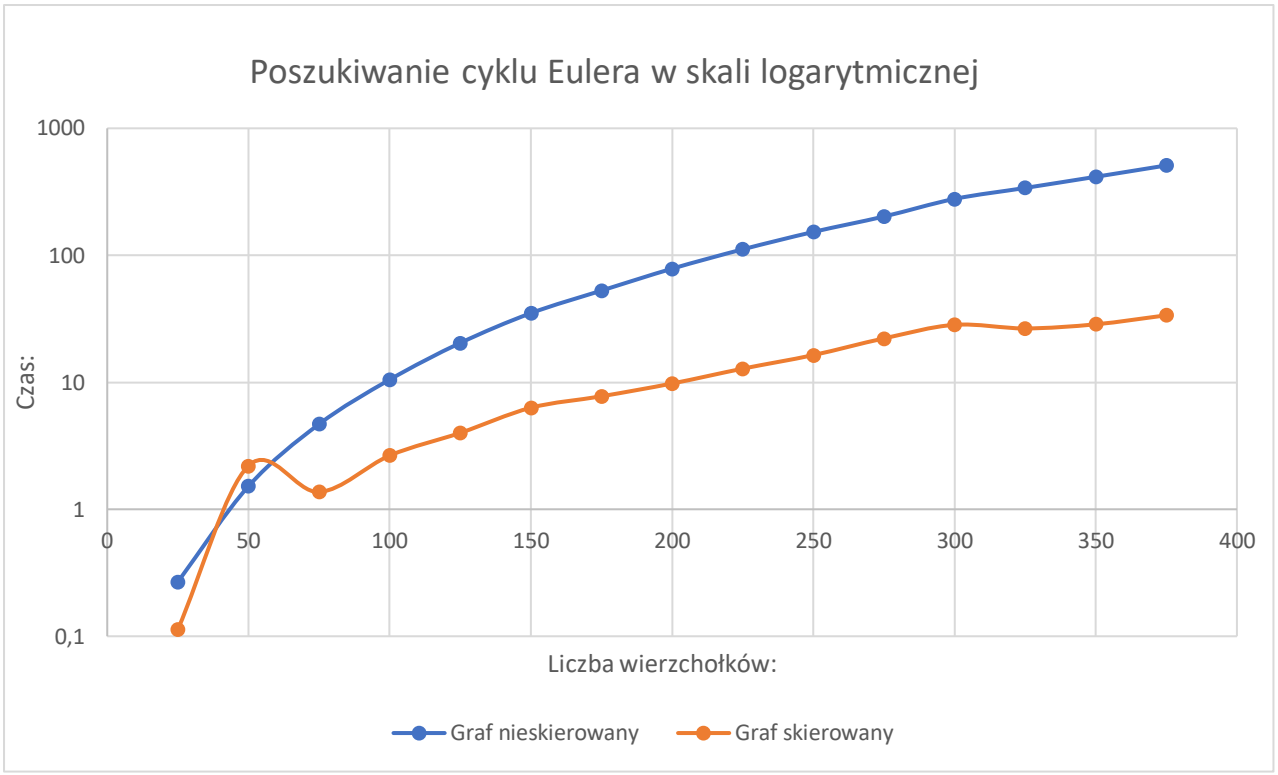
**Poszukiwanie cyklu Hamiltona i Eulera dla grafu nieskierowanego oraz skierowanego dla grafu o nasyceniu 50% w zależności od liczby wierzchołków:**



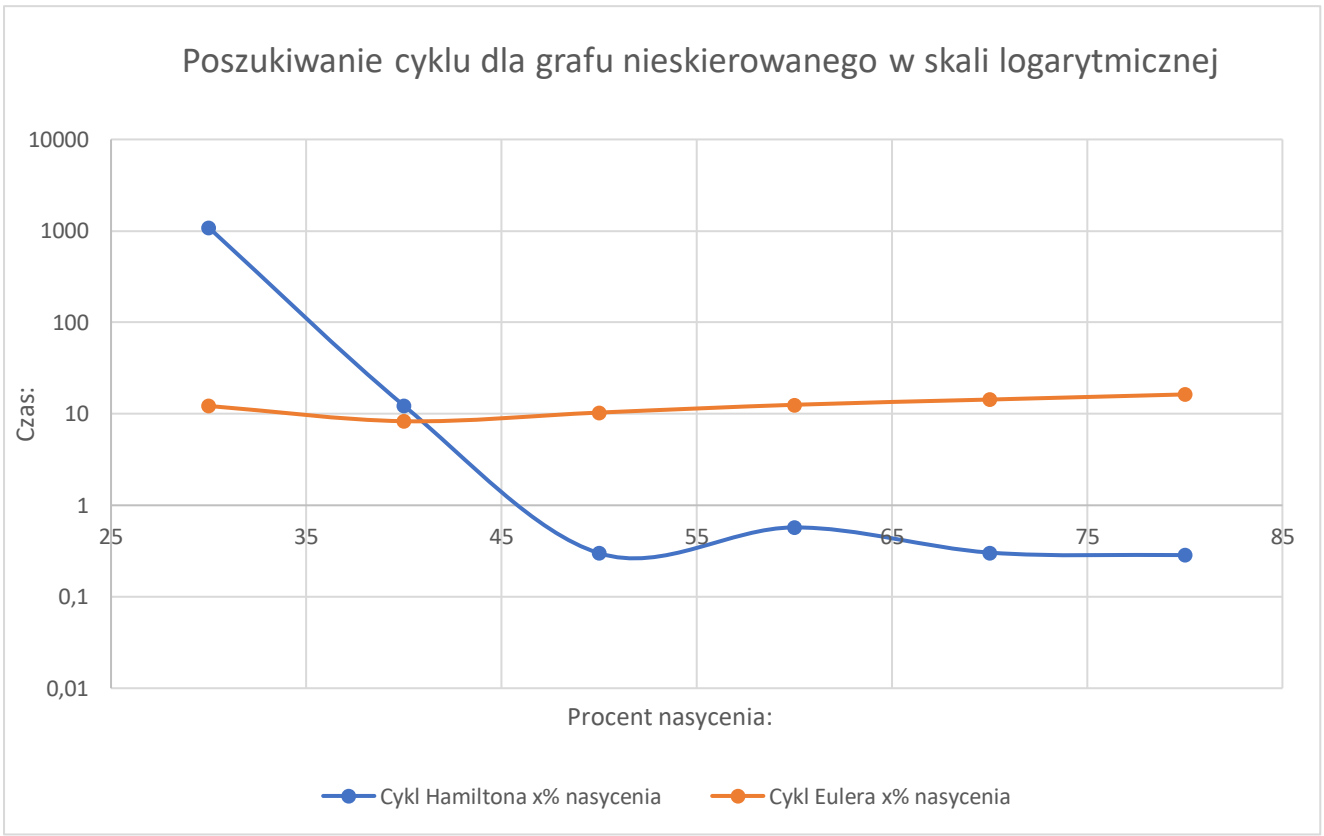


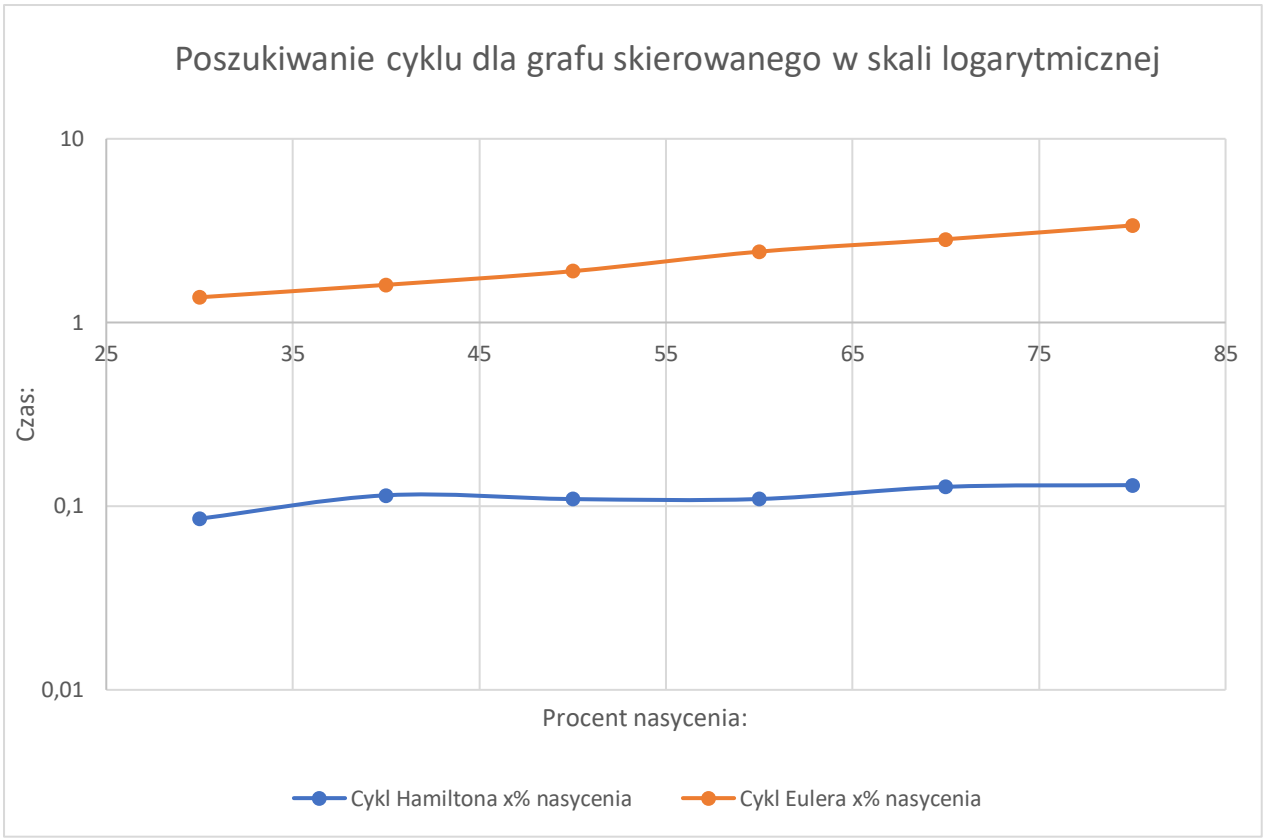

**Poszukiwanie cyklu Hamiltona i Eulera przy stałym nasyceniu w zależności od liczby wierzchołków dla grafu skierowanego i nieskierowanego:**



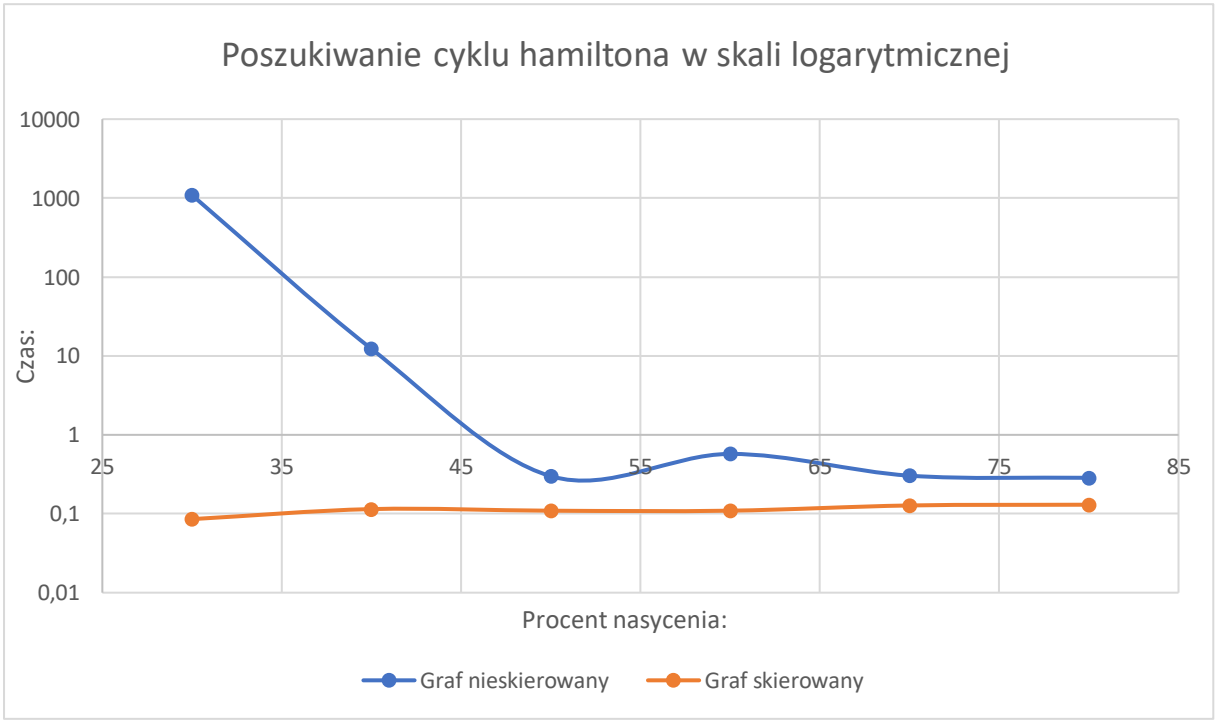


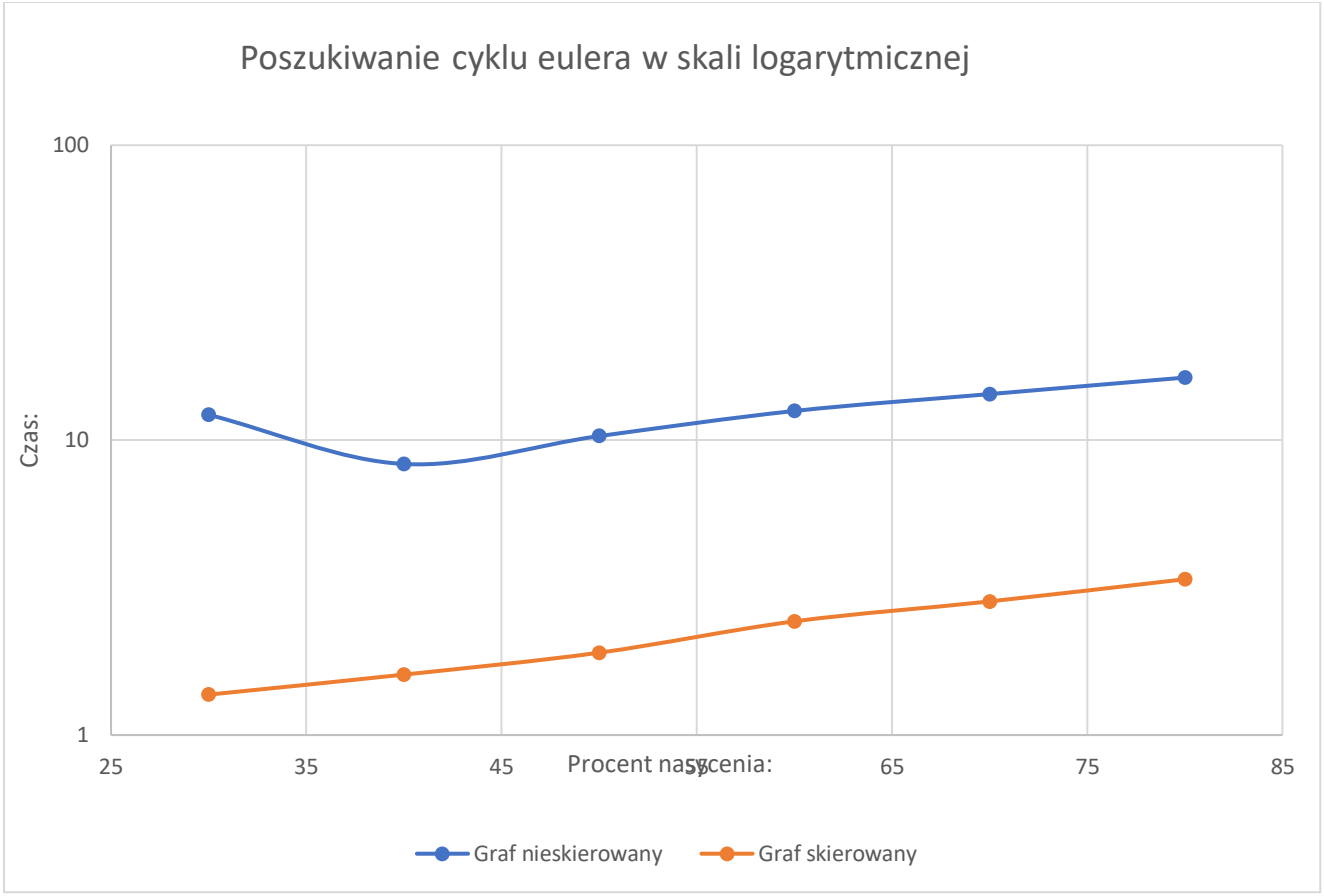

**Poszukiwanie cyklu Hamiltona i Eulera dla grafu nieskierowanego oraz skierowanego w zależności od procentu nasycenia grafu:**



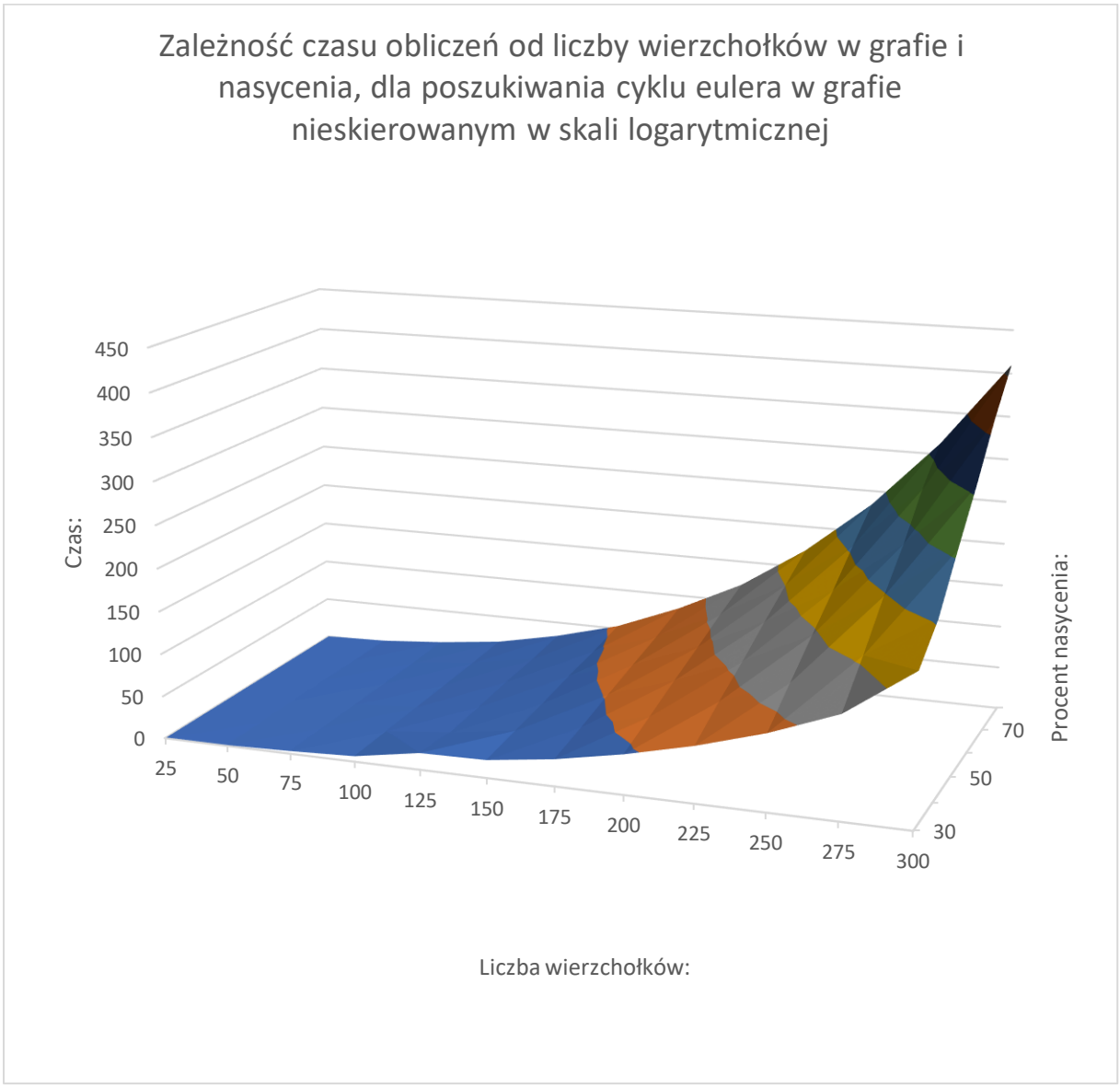


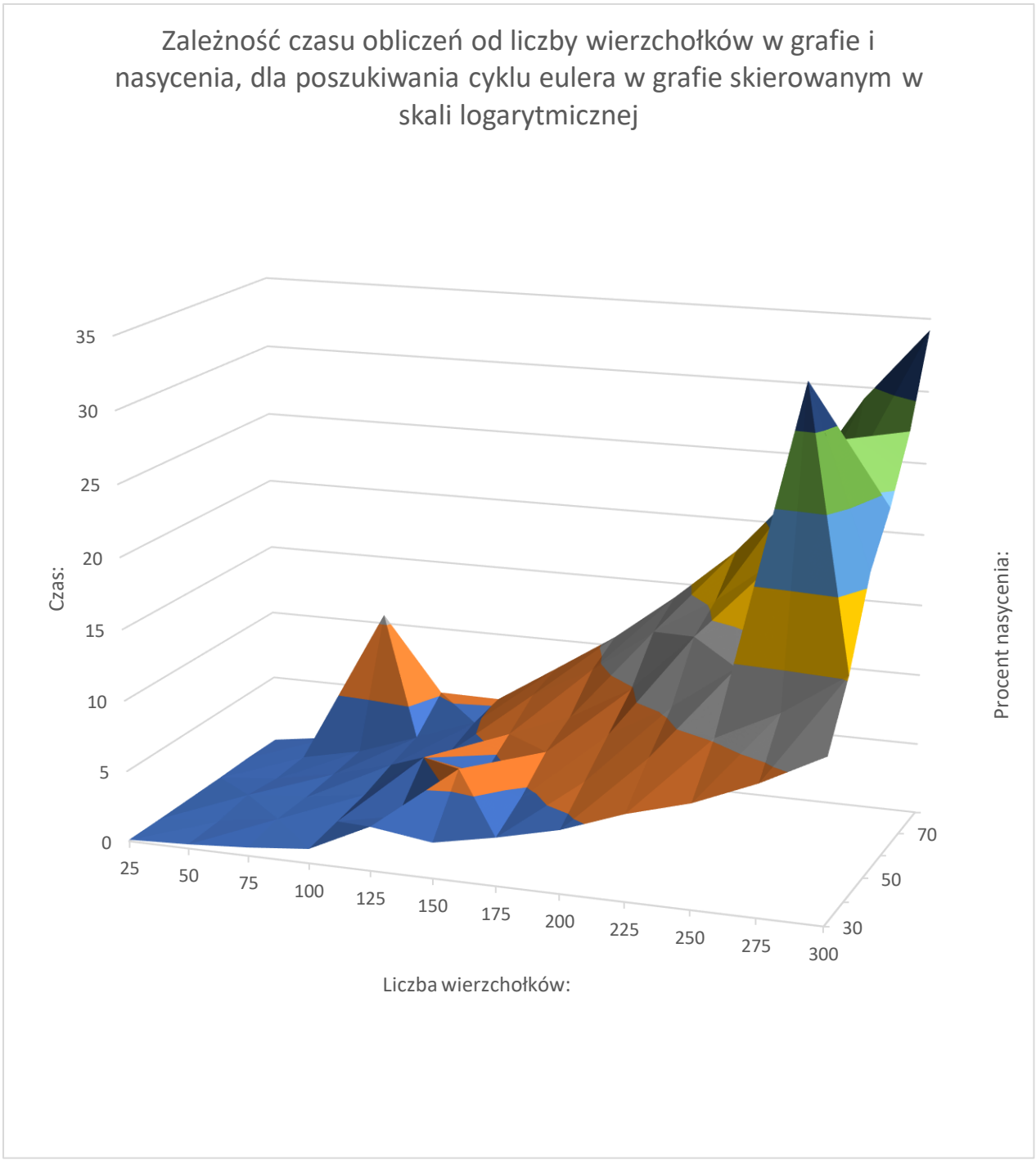

**Poszukiwanie cyklu Hamiltona i Eulera dla grafu nieskierowanego oraz skierowanego w zależności od procentu nasycenia grafu:**

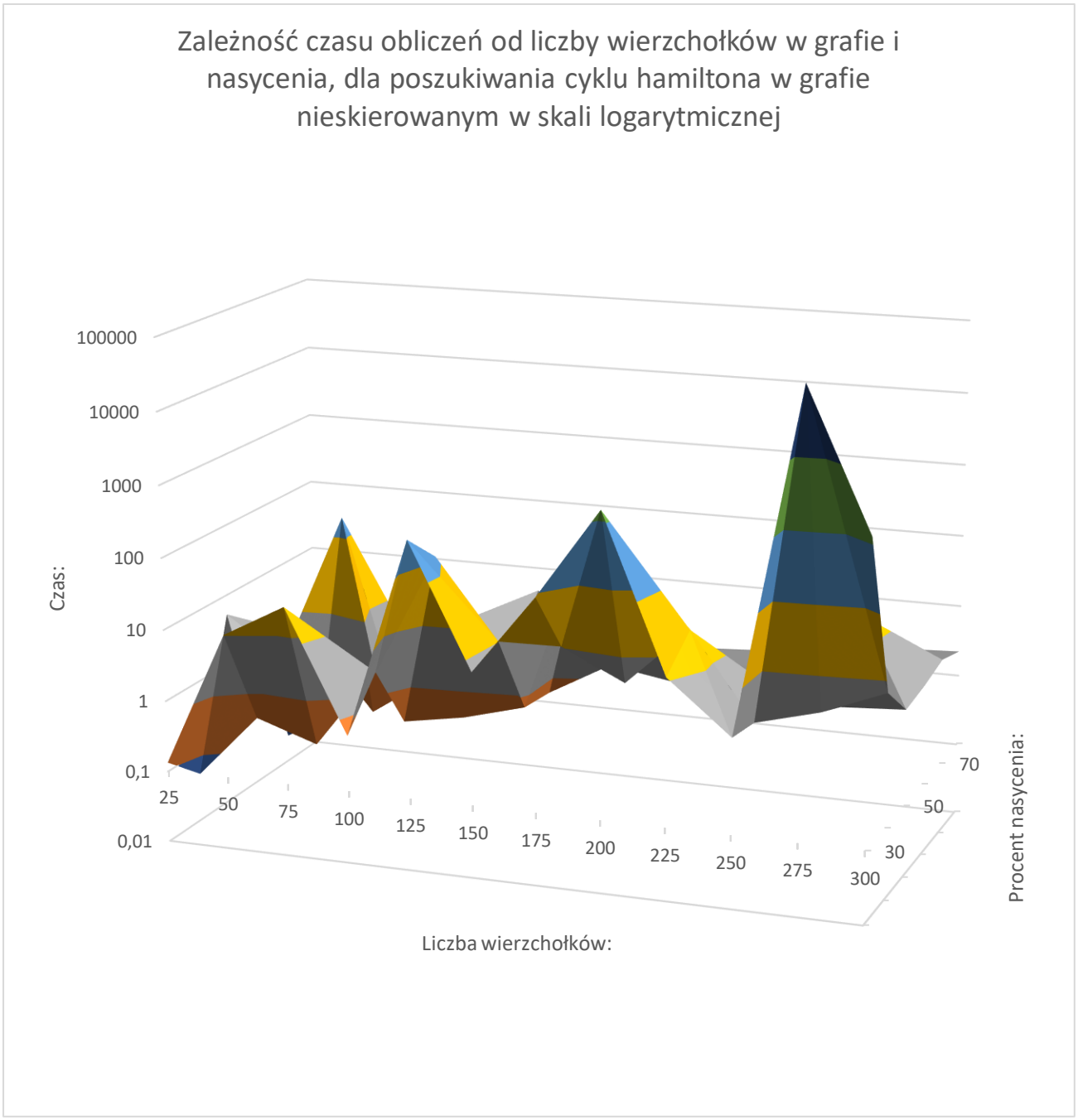



Poszukiwanie cyklu eulera w skali logarytmicznej
100
10
1
Czas:
25
35
45
55
65
75
85
Procent nasycenia:
Graf nieskierowany
Graf skierowany

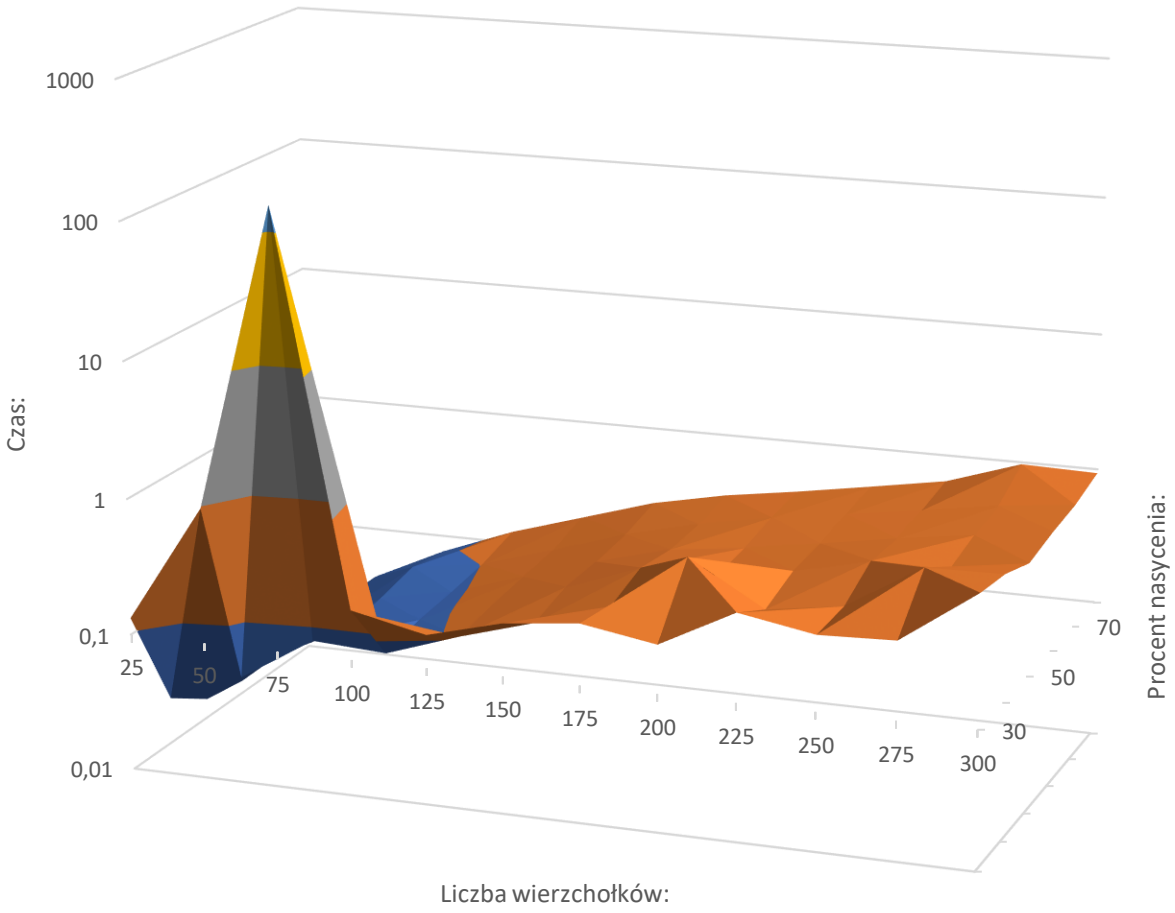
**Zależność czasu obliczeń od liczby wierzchołków w grafie i nasycenia, dla poszukiwania cyklu Eulera i Hamiltona w grafie nieskierowanym i skierowanym:**




Zależność czasu obliczeń od liczby wierzchołków w grafie i nasycenia, dla poszukiwania cyklu eulera w grafie skierowanym w skali logarytmicznej
Czas:
35
30
25
20
15
10
5
0
25
50
75
100
125
150
175
200
225
250
275
300
Liczba wierzchołków:
70
50
30
Procent nasycenia:


Zależność czasu obliczeń od liczby wierzchołków w grafie i nasycenia, dla poszukiwania cyklu hamiltona w grafie nieskierowanym w skali logarytmicznej
Czas:
100000
10000
1000
100
10
1
0,1
0,01
25
50
75
100
125
150
175
200
225
250
275
300
Liczba wierzchołków:
30
50
70
Procent nasycenia:

Zależność czasu obliczeń od liczby wierzchołków w grafie i nasycenia, dla poszukiwania cyklu eulera w grafie skierowanym w skali logarytmicznej



**Wnioski:**

Obserwacje dotyczące działania zaimplementowanych algorytmów dla grafów o różnym nasyceniu wskazują na to, że czas wykonania algorytmów zależy w dużej mierze od struktury i charakterystyki konkretnego grafu. Każdy przypadek i graf mogą mieć różne właściwości, co wpływa na złożoność obliczeniową i czas działania algorytmów.

**Cykl Eulera:**

Algorytm poszukiwania cyklu Eulera oparty na DFS ma złożoność obliczeniową O(m), gdzie m to liczba krawędzi w grafie. Jednak warto zauważyć, że to jest złożoność oczekiwana i może się różnić w praktyce w zależności od grafu.

Wybór reprezentacji maszynowej grafu dla tego algorytmu zależy od charakterystyki grafu.

Ważne jest podkreślenie, że zwiększenie liczby krawędzi w grafie powoduje liniowy wzrost czasu obliczeń (O(m)). Jednakże, to jak szybko algorytm znajdzie cykl zależy od struktury konkretnego grafu. Istnieją grafy o dużej liczbie krawędzi, dla których algorytm znajduje cykl Eulera bardzo szybko, podczas gdy dla innych grafów może to zająć znacznie więcej czasu.

**Cykl Hamiltona:**

Problem znalezienia cyklu Hamiltona należy do klasy problemów silnie NP-trudnych, a złożoność obliczeniowa wynosi O(n!), gdzie n to liczba wierzchołków w grafie. Ta złożoność oznacza, że czas wykonania algorytmu rośnie wykładniczo wraz z rozmiarem grafu.

Znalezienie cyklu Hamiltona wymaga przeglądania ogromnej przestrzeni rozwiązań, co sprawia, że czas wykonania jest bardzo długi.

Wybór odpowiedniej reprezentacji maszynowej grafu dla algorytmu z powracaniem również zależy od struktury grafu

Warto zaznaczyć, że czas wykonania algorytmu poszukiwania cyklu Hamiltona może się znacznie różnić w zależności od grafu. Niektóre grafy mogą mieć łatwo odnajdywalne cykle Hamiltona, podczas gdy inne mogą mieć bardziej skomplikowaną strukturę, co znacznie wydłuża czas wykonania algorytmu. W praktyce istnieją grafy o dużej liczbie wierzchołków, dla których znalezienie cyklu Hamiltona jest bardzo trudne lub nawet niemożliwe do osiągnięcia w rozsądnym czasie.

**Podsumowanie:**

Czas wykonania algorytmów zależy nie tylko od liczby wierzchołków i krawędzi, ale także od struktury grafu. Grafy o prostych i regularnych strukturach mogą być łatwiejsze do przeszukania, podczas gdy grafy o skomplikowanych i gęstych strukturach mogą wymagać znacznie więcej czasu.

Podsumowując, obserwacje wskazują na to, że czas wykonania algorytmów poszukiwania cykli w grafach zależy od wielu czynników, takich jak struktura grafu, liczba wierzchołków, liczba krawędzi oraz wybrana reprezentacja maszynowa. Nie ma jednego rozwiązania pasującego do wszystkich przypadków, dlatego ważne jest uwzględnienie specyfiki konkretnego grafu i dostosowanie strategii poszukiwań odpowiednio do danego problemu.