Informatyka, studia dzienne, inż I st. semestr V

**Sztuczna inteligencja i systemy ekspertowe 2021/2022**

|  |  |  |  |
| --- | --- | --- | --- |
| Prowadzący: dr. Inż. Krzysztof Lichy | |  | Czwartek, 15.45 |
|  |  |  |  |
| Data oddania: | 02.12.2021 | Ocena: |  |

Maciej Kolibabski 229916

Kacper Świąder 230023

Zadanie 1 : Piętnastka

1. Cel

Celem zadania było zaimplementowanie programu rozwiązującego grę „piętnastkę” używającego różnych metod rozwiązywania, a także porównania statystyk tych metod.

1. Wprowadzenie

Gra „15puzzle” potocznie zwana „piętnastką” to plansza o wymiarach 4x4 na której znajduje się 15 ruchomych kwadratowych kwadratów ponumerowanych od 1 do 15. Jedno miejsce na planszy zawsze pozostaje puste by móc umożliwić przemieszczanie 15 ruchomym kwadratom, by móc ułożyć je w kolejności 1-15.

Rozwiązywanie tej łamigłówki pod względem implementacji programu to przeszukiwanie drzewa w którym wierzchołkami są aktualne kombinacje ułożenia puzzli, a wykonywane ruchy są krawędziami drzewa.

Znalezienie rozwiązania można wykonać na przykład używając strategii:

- BFS – przeszukiwanie wszerz

- DFS – przeszukiwanie w głąb

- Astr – Algorytm z heurystykami Hamminga i Manhattan

DFS i BFS:

Procedury przeglądania grafu w głąb (DFS) i wszerz (BFS) są bardzo często wykorzystywane.  
W strategii DFS wybrany wierzchołek należy umieścić na stosie, zaznaczyć jako odwiedzony a następnie przejść do jego następnika. Następnik również umieszczamy na stosie, zaznaczamy jako odwiedzony przechodzimy do jego następnika. Jeśli dojdziemy do takiego wierzchołka, że nie ma on krawędzi z nieodwiedzonymi wierzchołkami, należy usunąć go ze stosu i pobrać ze stosu kolejny wierzchołek do przeszukania. . Przeszukiwanie DFS wykorzystuje się do badania spójności grafu. Jeśli procedura wywołana dla pierwszego wierzchołka "dotrze" do wszystkich wierzchołków grafu to graf jest spójny.   
Aby przeszukać graf wszerz (BFS) należy zamiast stosu wykorzystać kolejkę do przechowywania wierzchołków a kolejnych nieodwiedzonych następników szukać od początku macierzy.

Algorytm ASTR od wierzchołka układu początkowego tworzy ścieżkę, za każdym razem wybierając wierzchołek x z dostępnych w danym kroku niezbędnych wierzchołków tak, by minimalizować funkcję f(x) zdefiniowaną

F(x) = g(x) + h(x)

Gdzie

- g(x) – droga pomiędzy wierzchołkiem początkowym a x.

- h(x) – przewidywana przez heurystykę droga od x do wierzchołka docelowego

Zostały wykorzystane dwie metryki:

* Hamminga - obliczane jest ile klocków nie znajduję się na właściwym miejscu
* Manhattan - obliczane jest ile ruchów jest potrzebne do osiągniecia docelowych miejsc

1. Opis implementacji

Wszystkie algorytmy zostały zaimplementowane przy użyciu języka programowania – Python.

Aplikacja jest konsolowa. Komenda wywołująca uruchomienie programu wygląda w następujący sposób:

*python [nazwa] [strategia] [argument] [plik z rozwiązaniem] [plik ze statystykami ]*

gdzie:

**[nazwa]** : to nazwa naszego programu np. main.py

**[strategia]:** BFS lub DFS lub ASTR

**[*argument*]** : w przypadku algorytmów BFS i DFS jest to permutacja liter LURD, natomiast w przypadku

wywołania algorytmu ASTR w jest to nazwa heurystyki MANH lub HAMM

**[plik z rozwiązaniem]:** nazwa pliku z rozwiązaniem np. 4x4\_00001\_bfs\_rdul\_sol.txt

**[plik ze statystykami]:** nazwa pliku ze statystykami np. 4x4\_00001\_bfs\_rdul\_stats.txt

W programie poza funkcjami samych trzech algorytmów (DFS, BFS, ASTR) zostało stworzone wiele dodatkowych funkcji pomocniczych między innymi takich jak :

* Funkcja wyznaczająca możliwe ruchy w taki sposób aby element nie mógł być przesunięty poza wymiary planszy, oraz nie mógł cofnąć się w to samo miejsce z którego został przesunięty
* Funkcja sprawdzająca czy układ jest rozwiązany
* Funkcja która lokalizuje na planszy puste pole
* Funkcje wykonujące zapis do plików
* Funkcje wykonujące odczyt z plików

Opisy implementacji funkcji BFS, DFS, ASTR:

BFS

Algorytm BFS został zaimplementowany iteracyjnie. Działa on z zastosowaniem kolejki FIFO

Przechodzenie grafu rozpoczyna się od zadanego wierzchołka i polega na odwiedzeniu wszystkich osiągalnych z niego wierzchołków, oraz usuwaniu z kolejki wierzchołków od których zaczynaliśmy znajdować osiągalne z niego wierzchołki. Odwiedzenie wszystkich wierzchołków zakończy się w momencie, gdy kolejka będzie pusta.

DFS

Algorytm DFS został zaimplementowany rekurencyjnie. Wywołania rekurencyjne na węzłach potomnych przez algorytm będą wywoływane do momentu osiągnięcia maksymalnej głębokości, w przypadku naszego programu – 20

Każdorazowe przejście do wierzchołka odznaczamy ustawiając flagę jako odwiedzony. Gdy skończą nam się wierzchołki, cofamy się do wierzchołka, z którego ostatnio przyszliśmy i wchodzimy z tego miejsca do następnego nieodwiedzonego

ASTR

1. Materiały i metody

Do wygenerowania wszystkich rozwiązań (plików solutions oraz plików stats ) został użyty skrypt umieszony na platformie WIKAMP (tzw. uruchamiacz przeszukiwań w trybie wsadowym ).

Należało przebadać wszystkie układy początkowe układanki w odległościach 1-7 od układu wzorcowego ( czyli 413 układów).

Dla algorytmów DFS oraz BFS należało użyć 8 następujących porządków przeszukiwania sąsiedztwa:

* prawo-dół-góra-lewo;
* prawo-dół-lewo-góra;
* dół-prawo-góra-lewo;
* dół-prawo-lewo-góra.
* lewo-góra-dół-prawo.
* lewo-góra-prawo-dół;
* góra-lewo-dół-prawo;
* góra-lewo-prawo-dół;

Natomiast dla algorytmu A\* należało użyć dwóch heurystyk – Manhattan i Hamminga

Program wygenerował łącznie kilkanaście tysięcy plików ponieważ :

**Algorytm DSF:**

413 (układów) \* 8 (porządków przeszukiwania sąsiedztwa) \* 2 (plik solution + plik stat ) = 6608 plików

**Algorytm BFS**

413 (układów) \* 8 (porządków przeszukiwania sąsiedztwa) \* 2 (plik solution + plik stat ) = 6608 plików

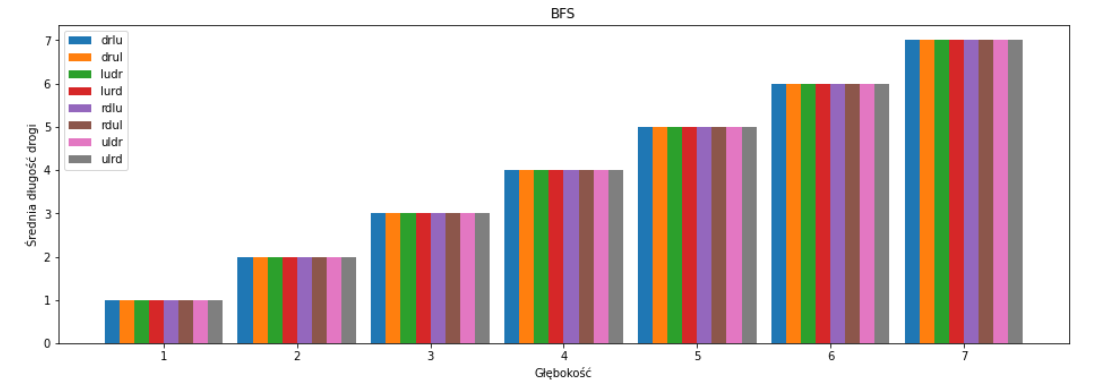
**Algorytm A\* :**

413 (układów) \* 2 (plik solution + plik stat ) \* 2 (metryka Hamminga + Manhattan) = 1652 plików

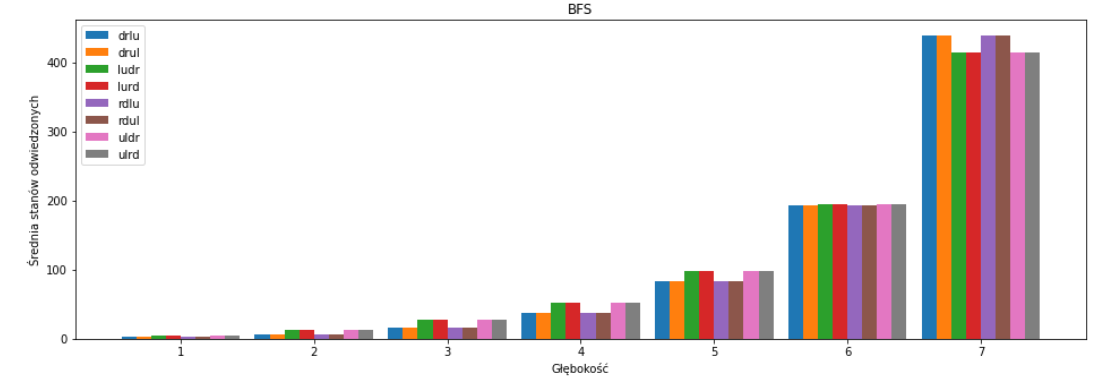
Po wykonaniu programu utworzone pliki zostały odpowiednio pogrupowane oraz oddane analizie na podstawie której zostały wygenerowane wykresy.

1. Wyniki

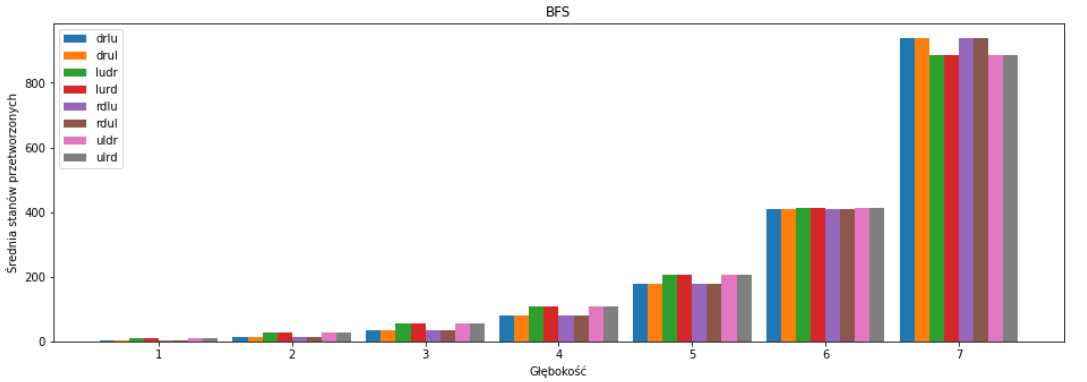
BFS



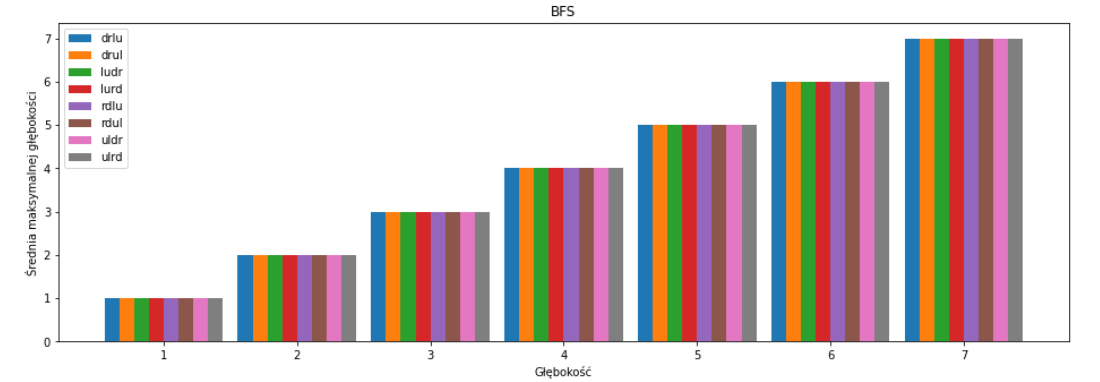
Rysunek 1



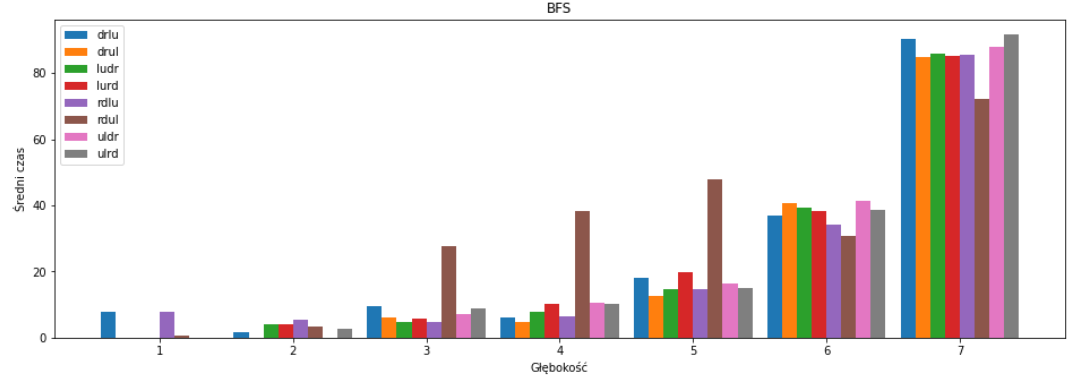
Rysunek 2



Rysunek 3

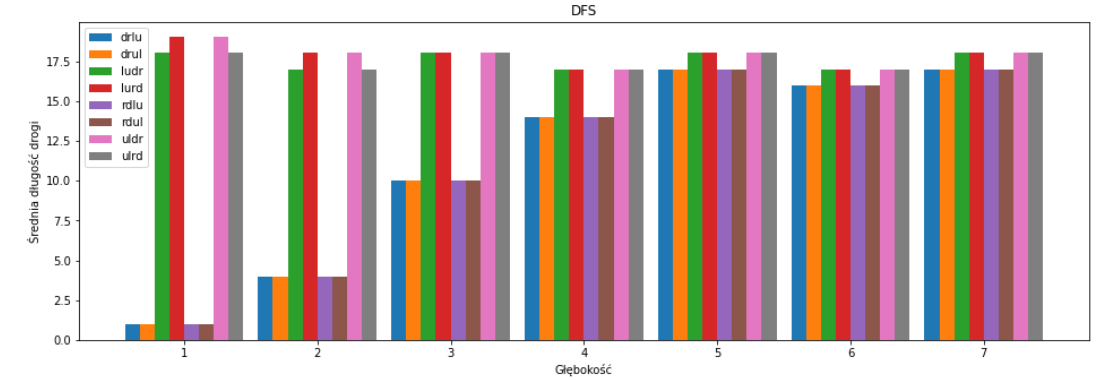


Rysunek 4

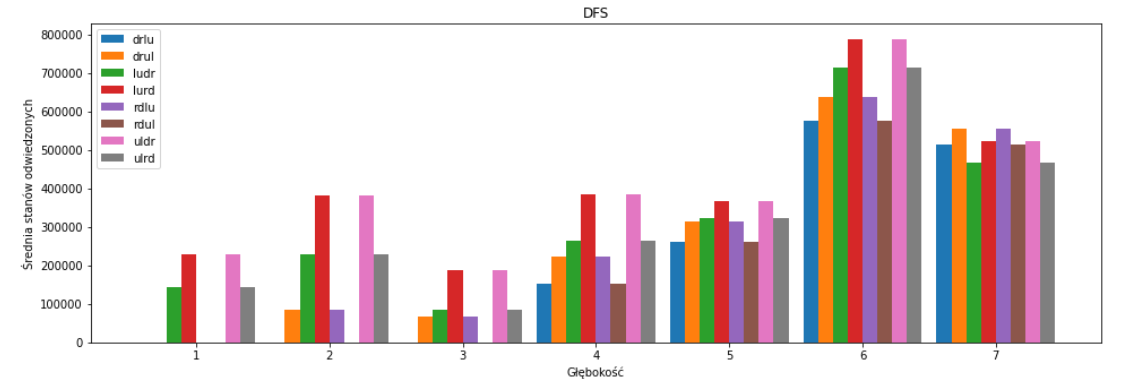


Rysunek 5

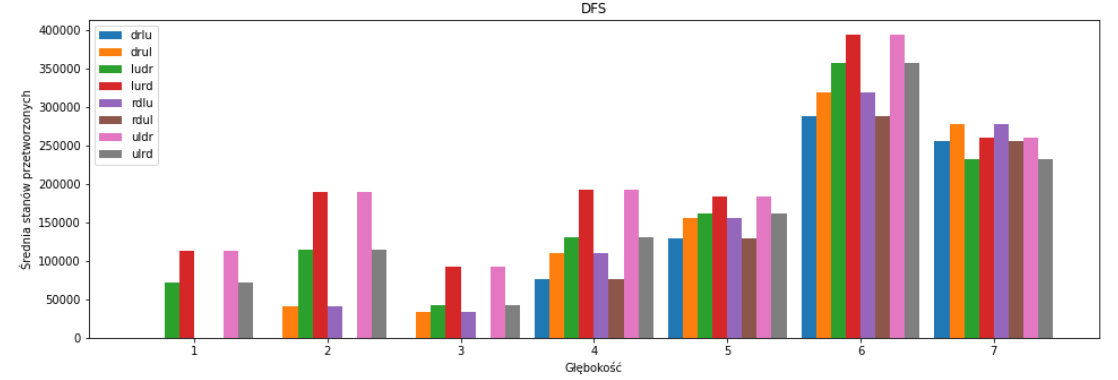
DFS



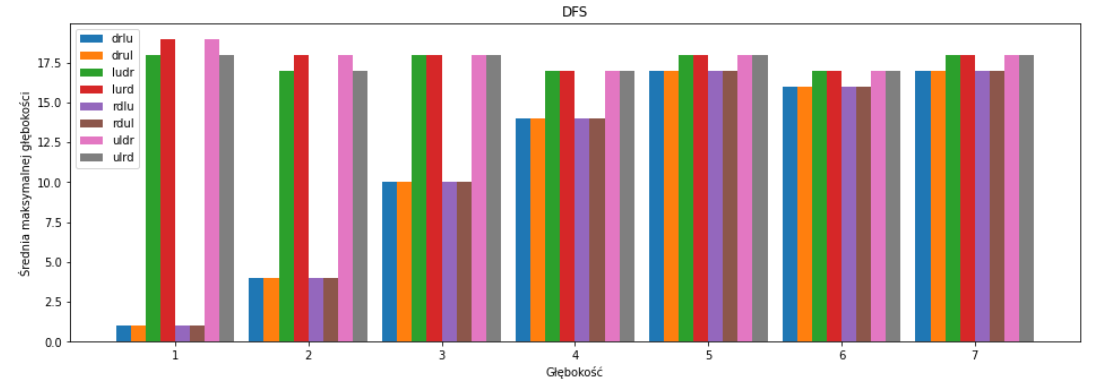
Rysunek 6



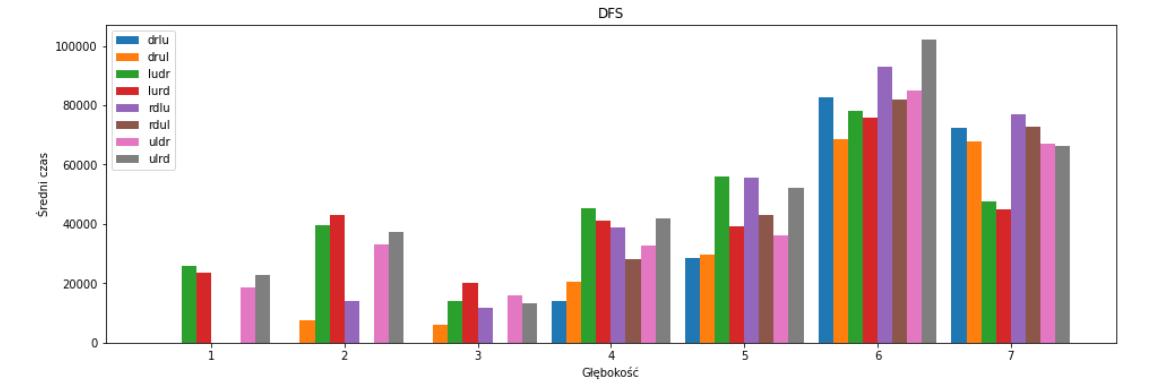
Rysunek 7



Rysunek 8

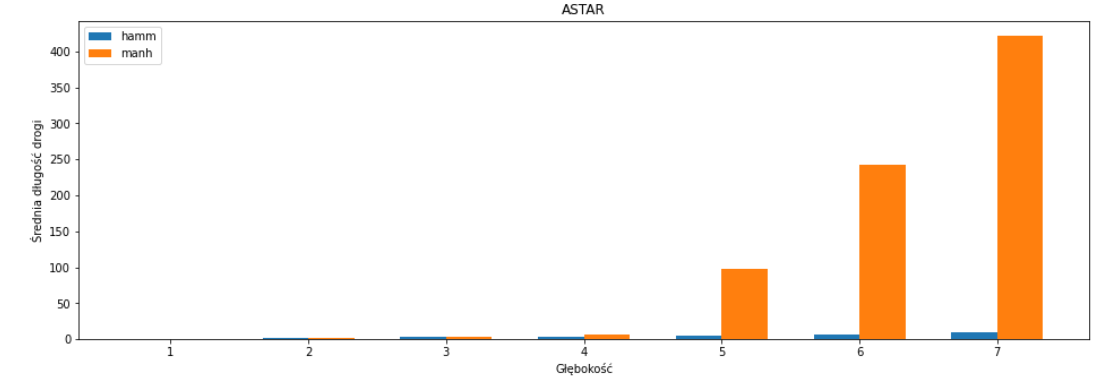


Rysunek 9

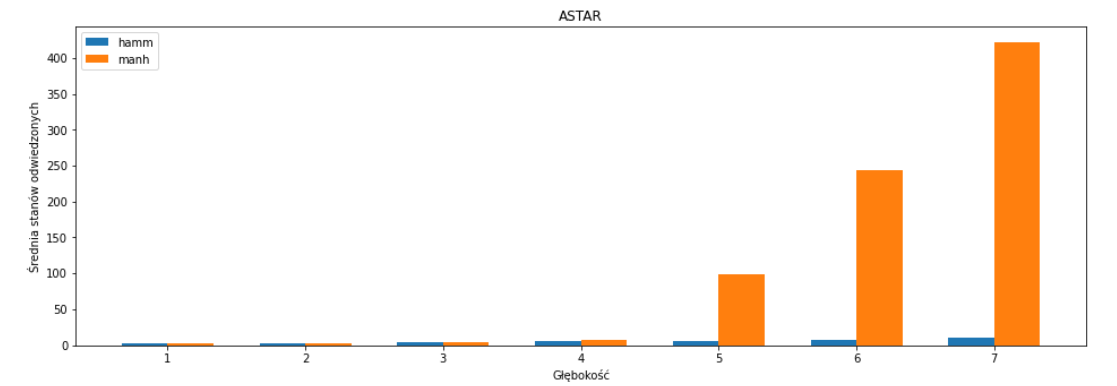


Rysunek 10

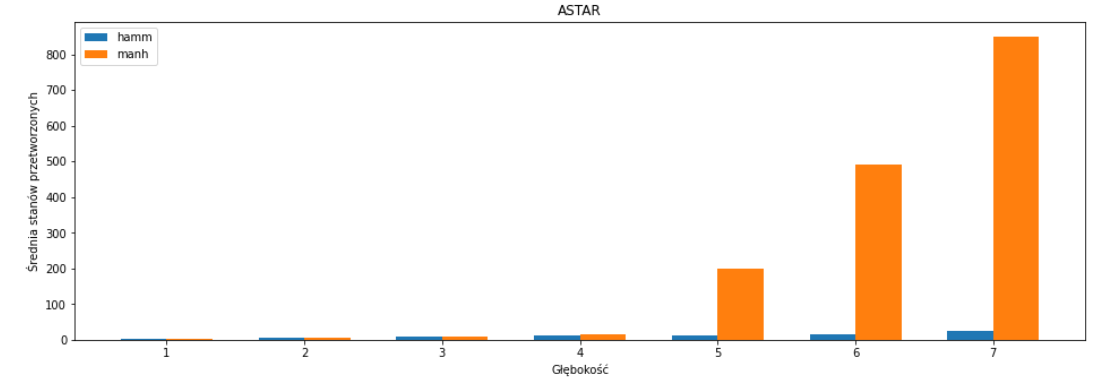
ASTR



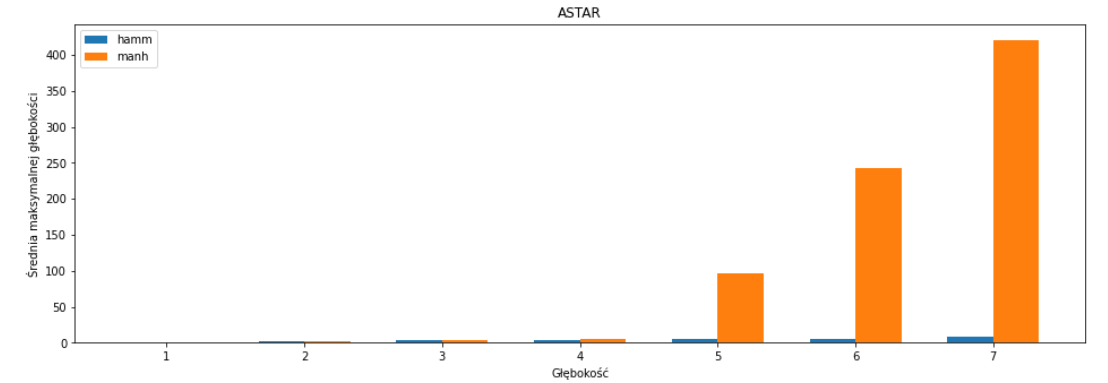
Rysunek 11



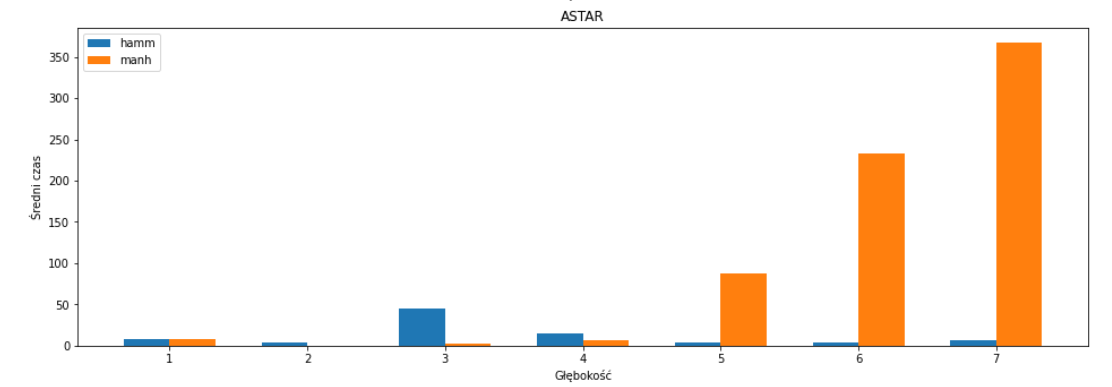
Rysunek 12



RRysunek 13

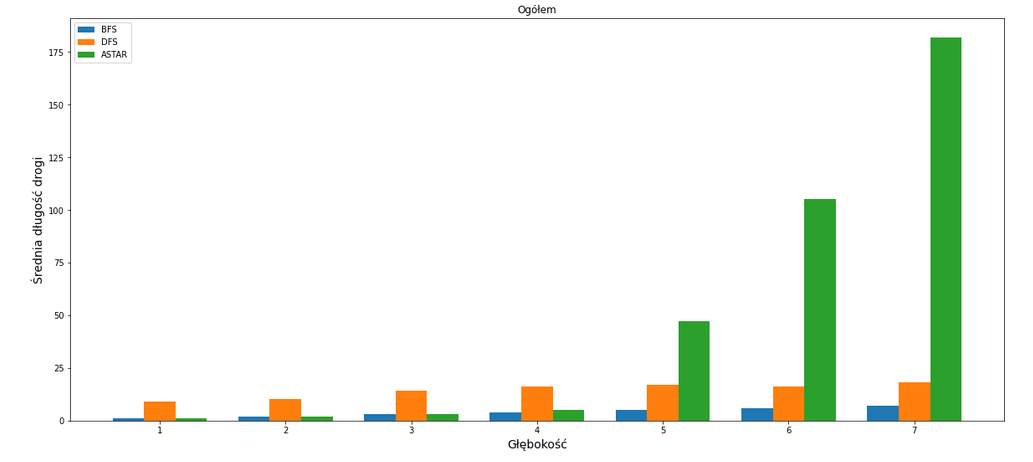


Rysunek 14

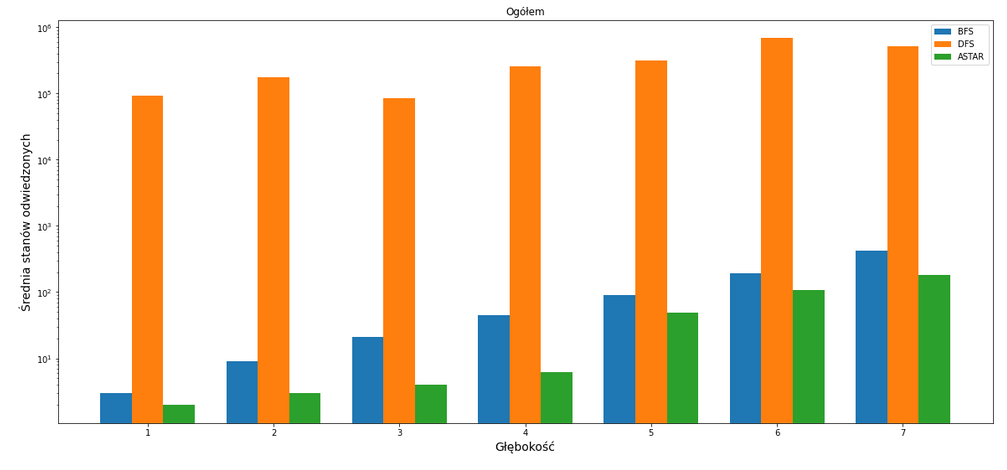


Rysunek 15

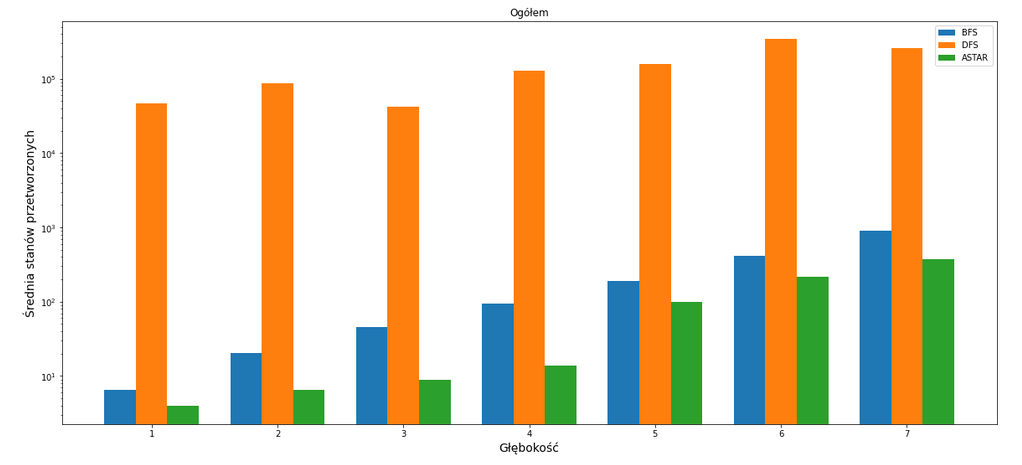
Ogółem



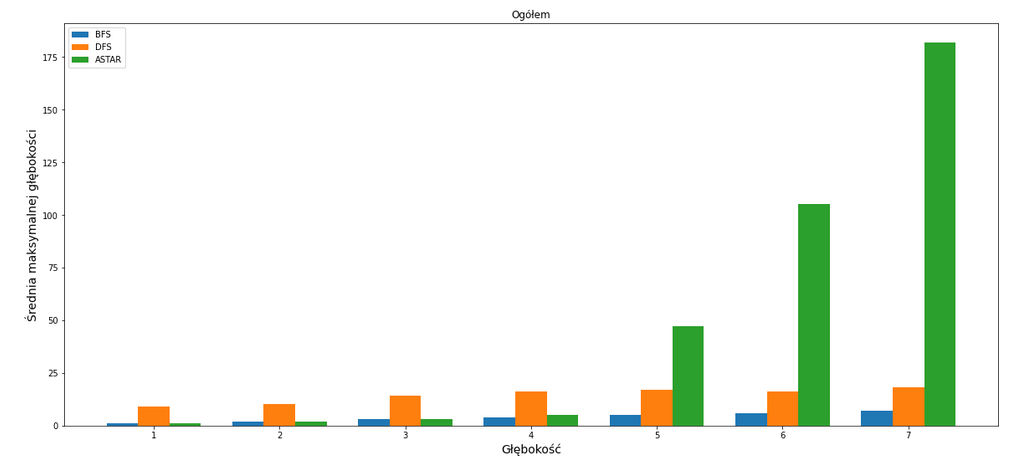
Rysunek 16



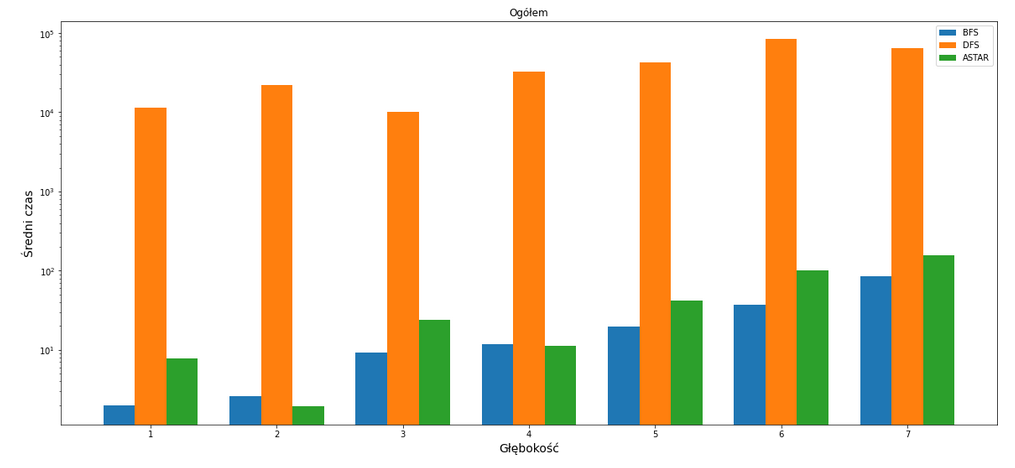
Rysunek 17



Rysunek 18



Rysunek 19



Rysunek 20

1. Dyskusja
2. Wnioski

Literatura

http://www.algorytm.org

https://pl.wikipedia.org