Informatyka, studia niestacjonarne, inż I st. semestr VI

Sztuczna inteligencja i systemy ekspertowe 2013/2014

Prowadzący: dr inż. Krzysztof Lichy sobota, 15:30

Data oddania: _____ Ocena: _____

Łukasz Ochmański 183566

Przemysław Szwajkowski 173524

Zadanie 1 - przeszukiwanie przestrzeni stanów*

1. Wprowadzenie

Celem niniejszego zadania jest napisanie dwóch programów. Pierwszy z nich ma za zadanie odnaleźć rozwiązanie łamigłówki zwanej "Piętnastką", a drugi ma na celu wizualizację rozwiązywania łamigłówki.

2. Uruchamianie programu

Program można uruchomić z lini poleceń w systemie z zainstalowaną wirtualną maszyną Java'y wersji 7 lub nowszej. Program przyjmuje jeden parametr:

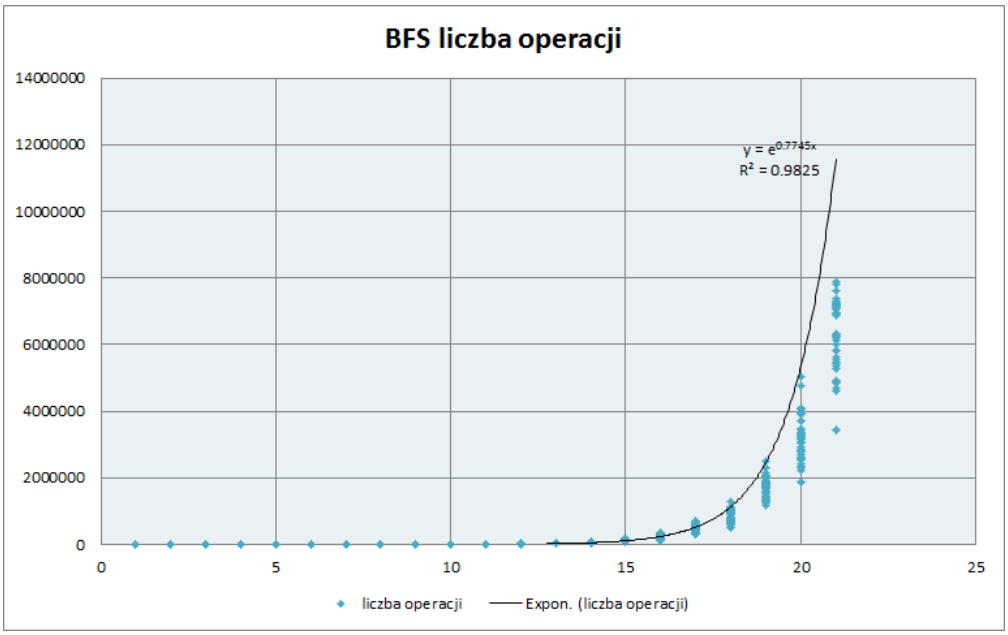
— `alogorytm`
a do wyboru: *dfs, bfs, a1, a2, a3*

Przed uruchomieniem należy spakować projekt wraz z bibliotekami do formatu *.jar. Metoda main() znajduje się w pliku Solver.java

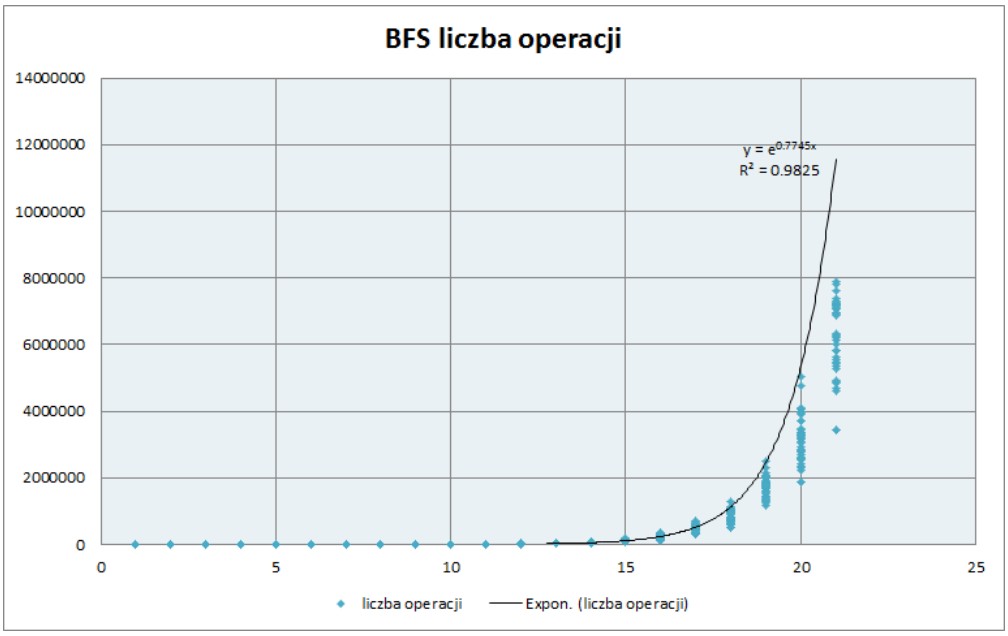
Następnie uruchomić polecenie:

```
java -jar Zadanie1.jar bfs 0 2 3 4 1 6 7 8 5 13 10 11 14 9 15 12
```

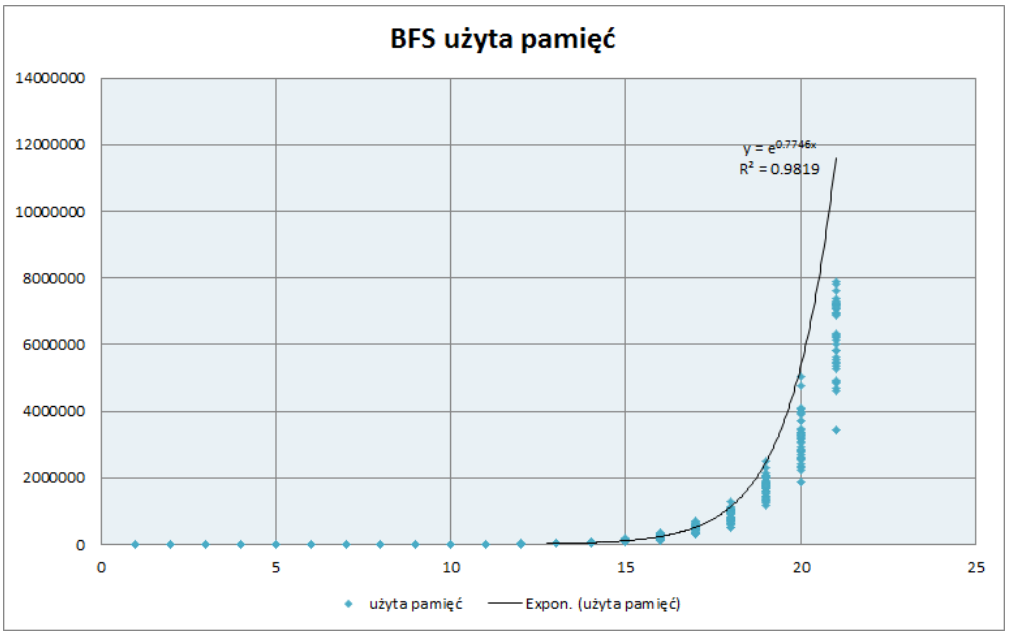
* SVN: `https://sise-lukasz-ochmanski.googlecode.com/svn/trunk/`



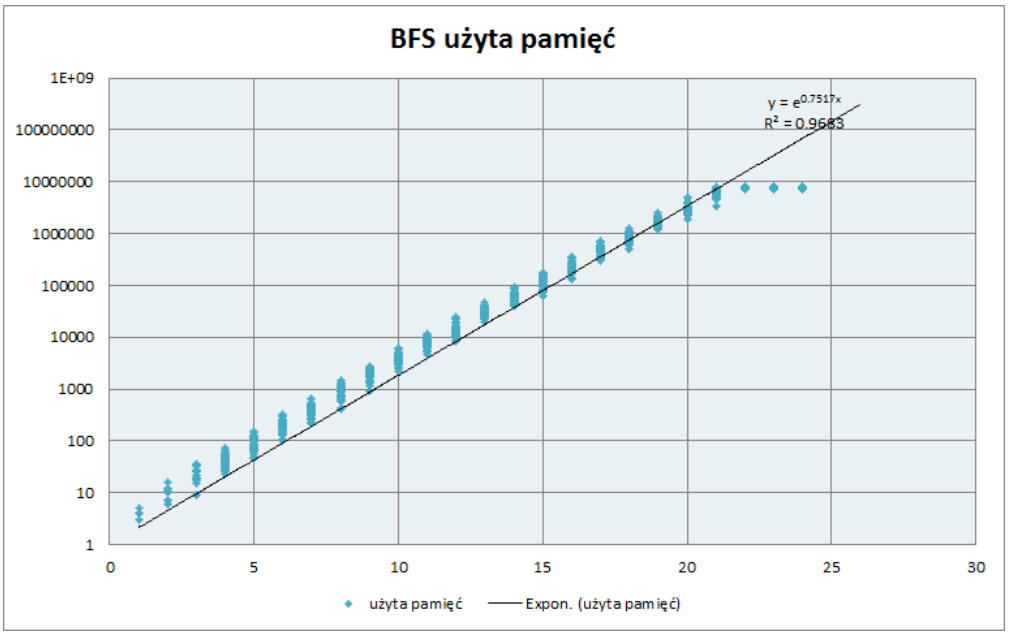
Rysunek 1. Breadth-First Search



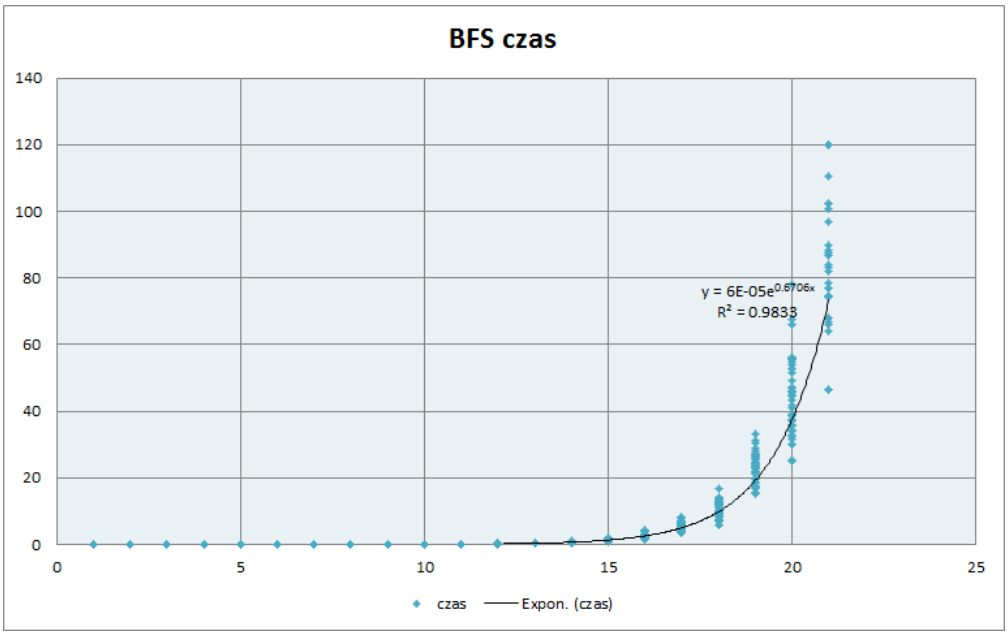
Rysunek 2. Breadth-First Search



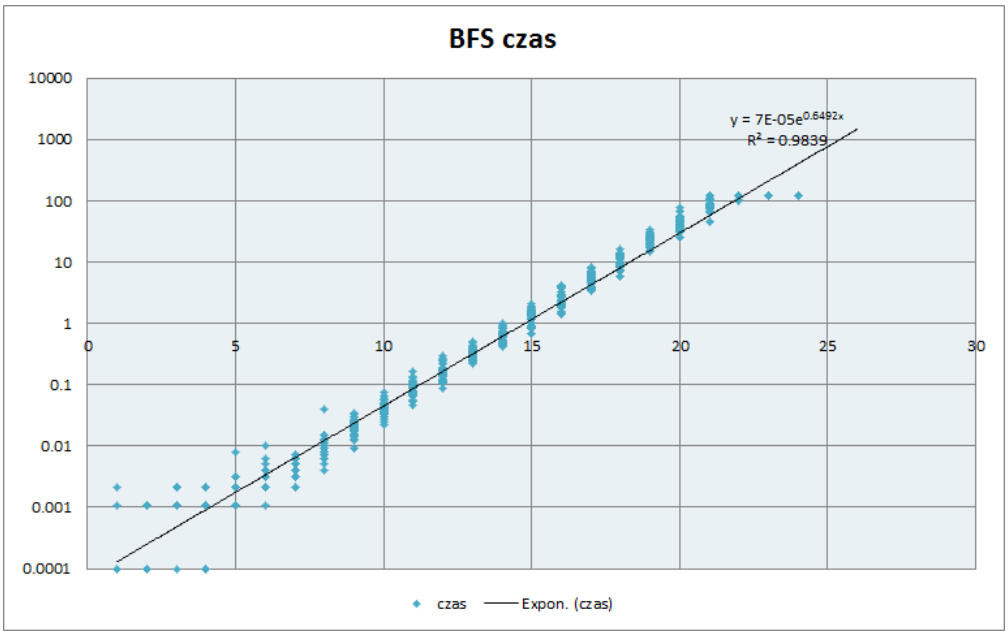
Rysunek 3. Breadth-First Search



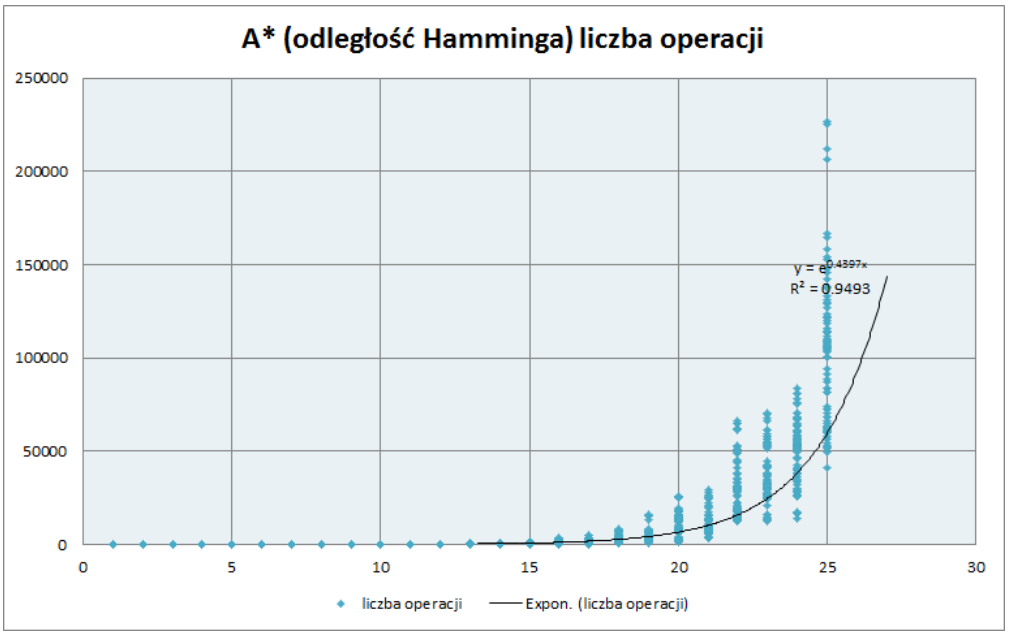
Rysunek 4. Breadth-First Search



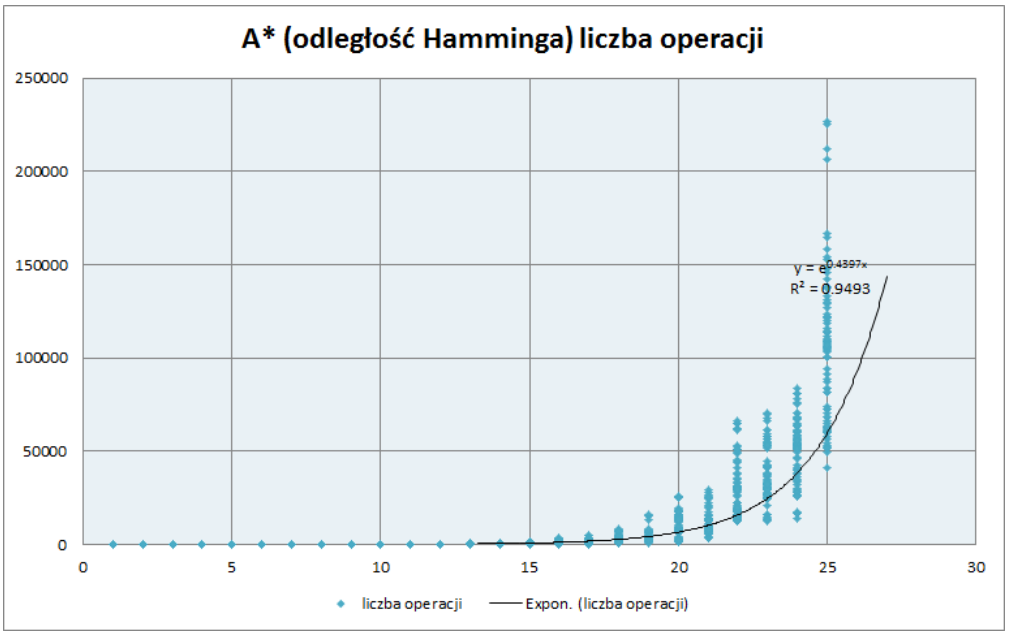
Rysunek 5. Breadth-First Search



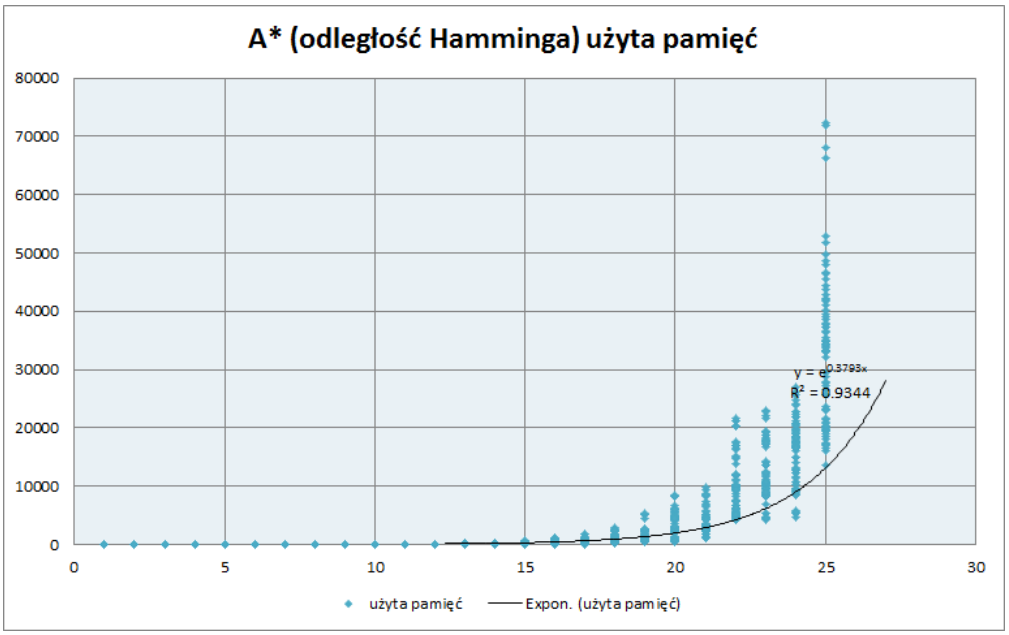
Rysunek 6. Breadth-First Search



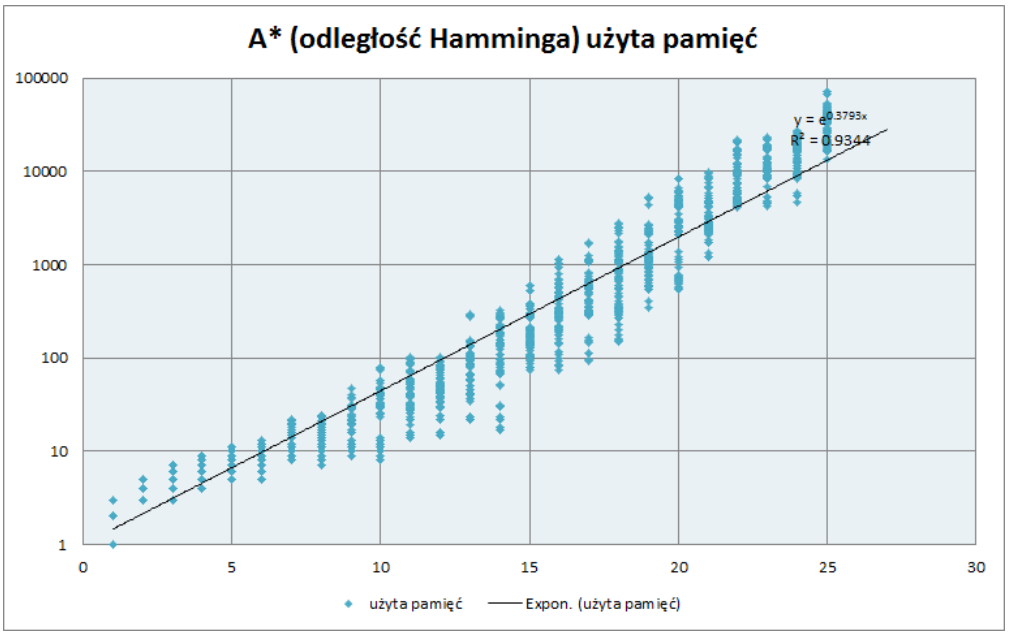
Rysunek 7. A* Odleglosc Hamminga



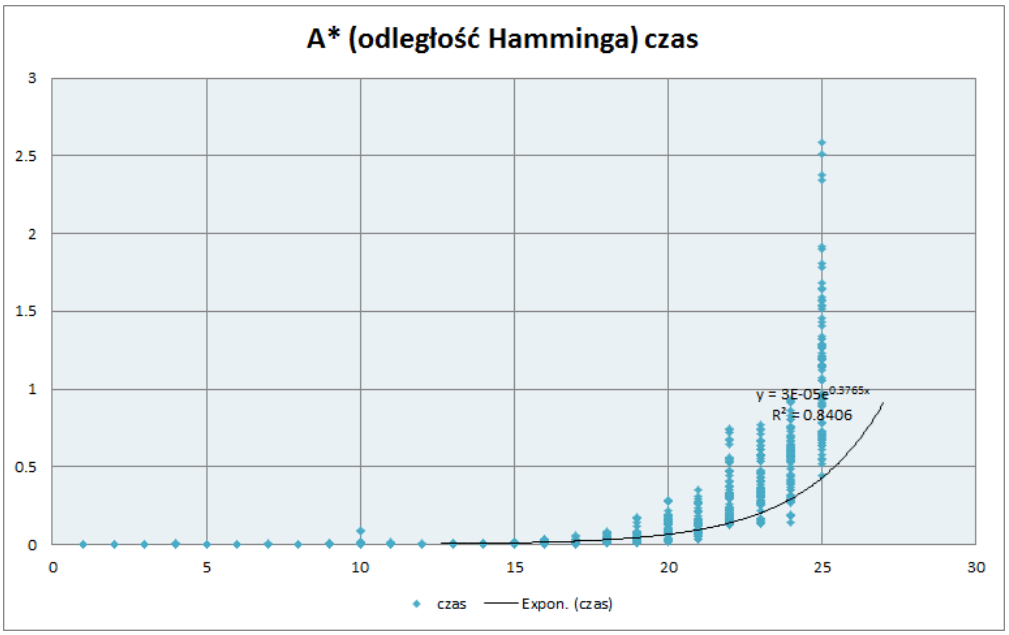
Rysunek 8. A* Odleglosc Hamminga



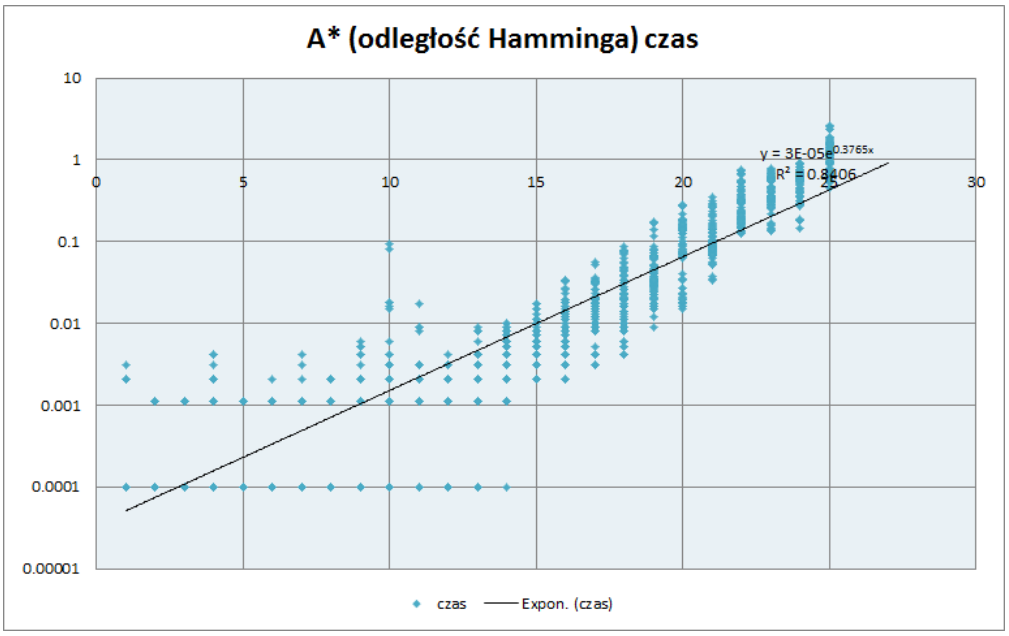
Rysunek 9. A* Odleglosc Hamminga



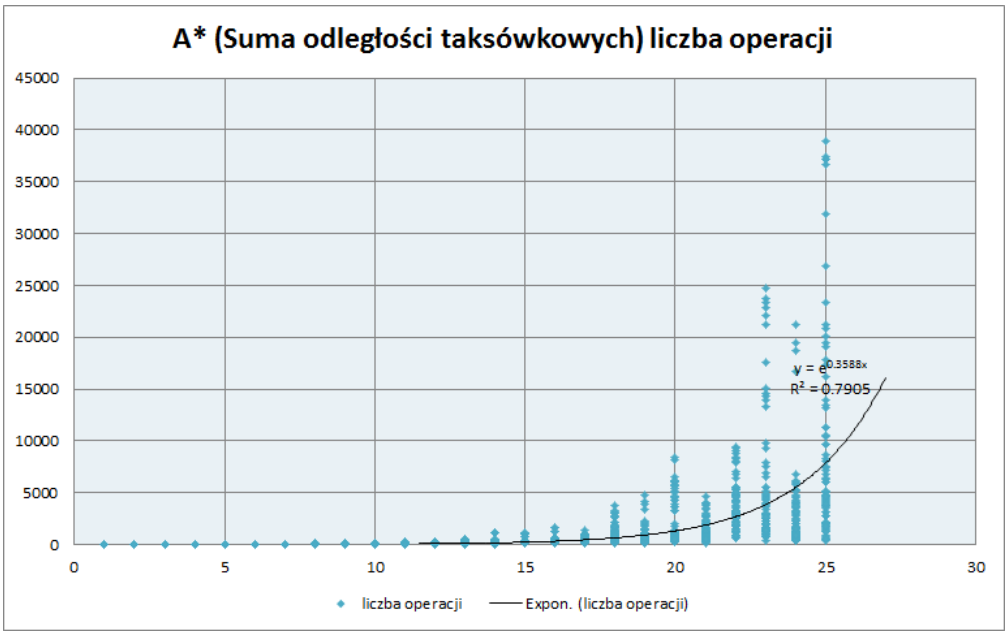
Rysunek 10. A* Odleglosc Hamminga



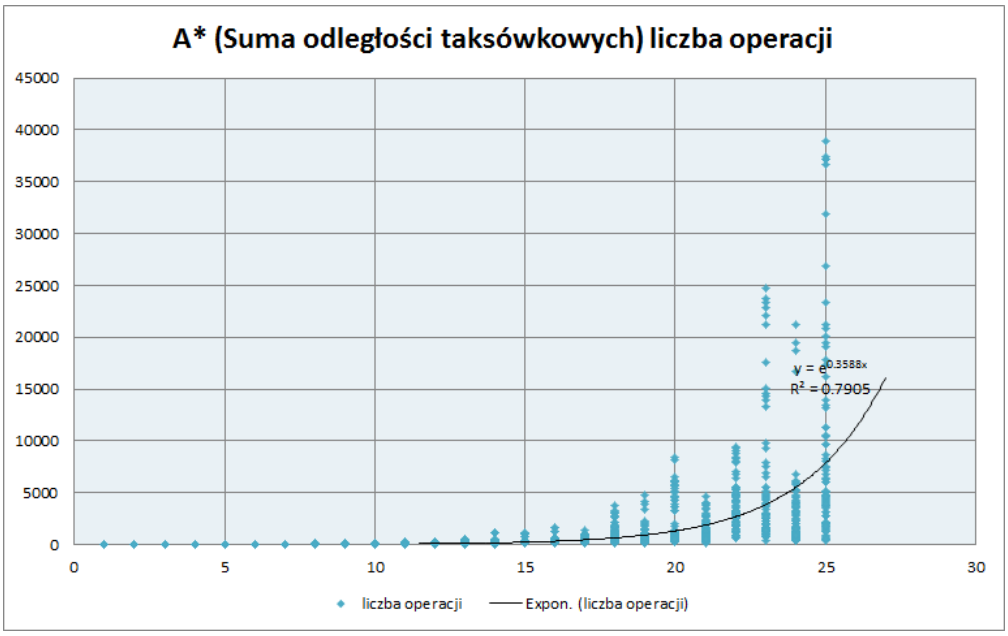
Rysunek 11. A* Odleglosc Hamminga



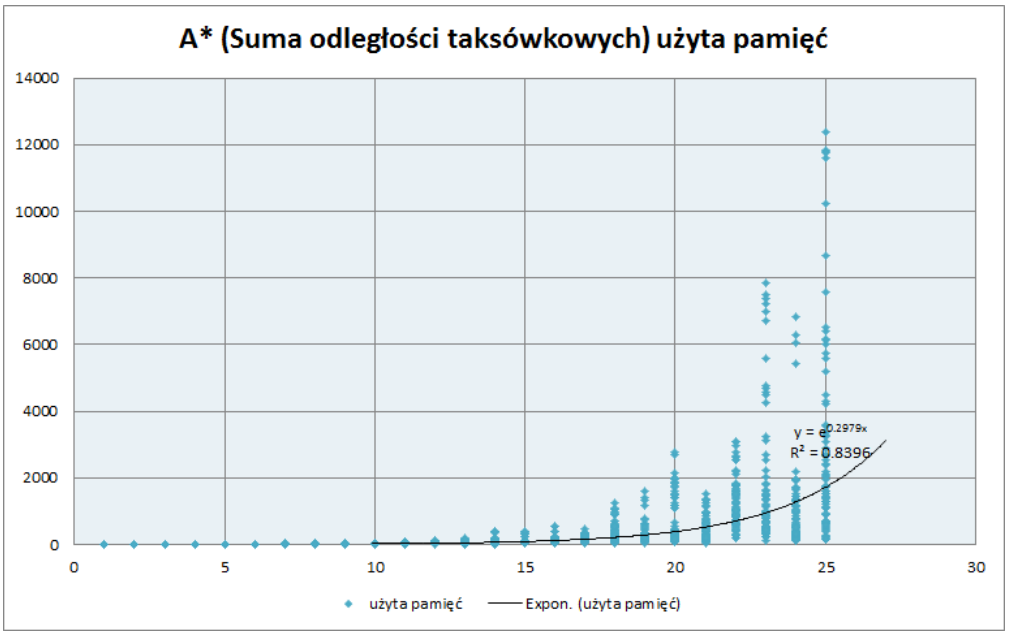
Rysunek 12. A* Odleglosc Hamminga



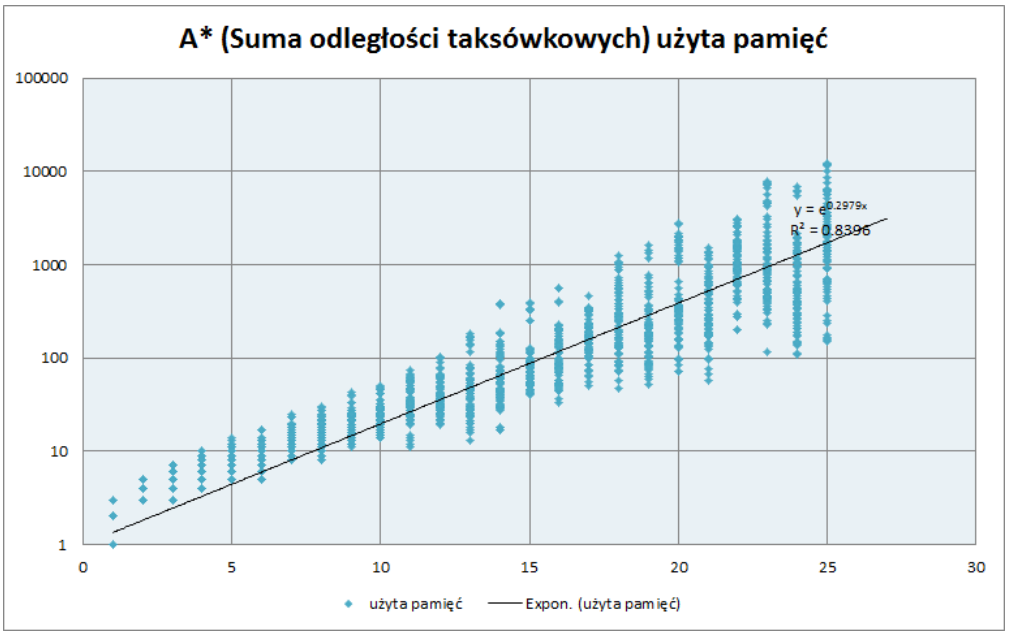
Rysunek 13. A* Suma odleglosci taksowkowych



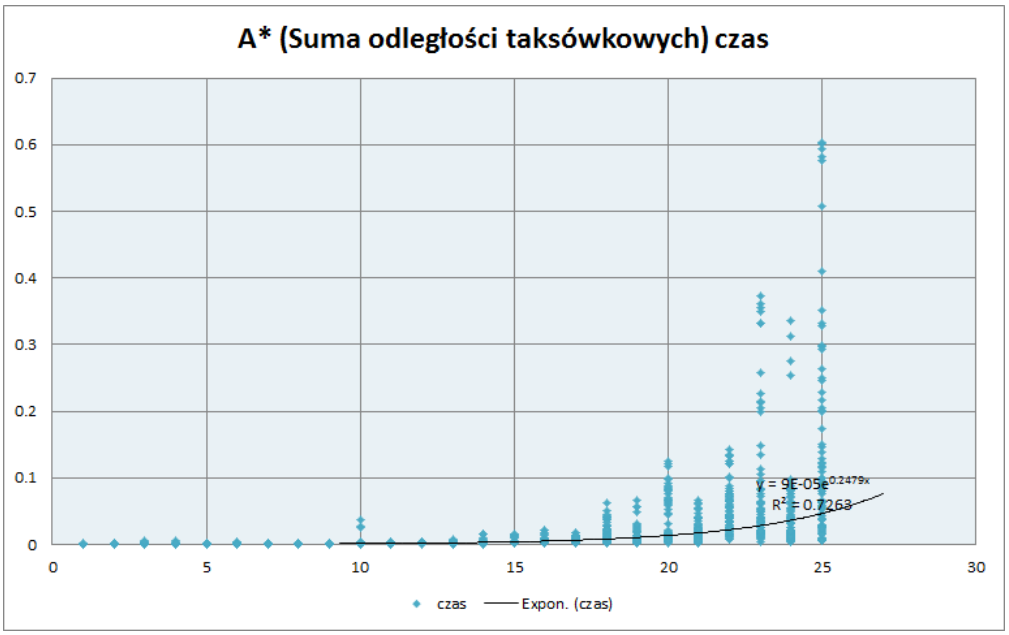
Rysunek 14. A* Suma odleglosci taksowkowych



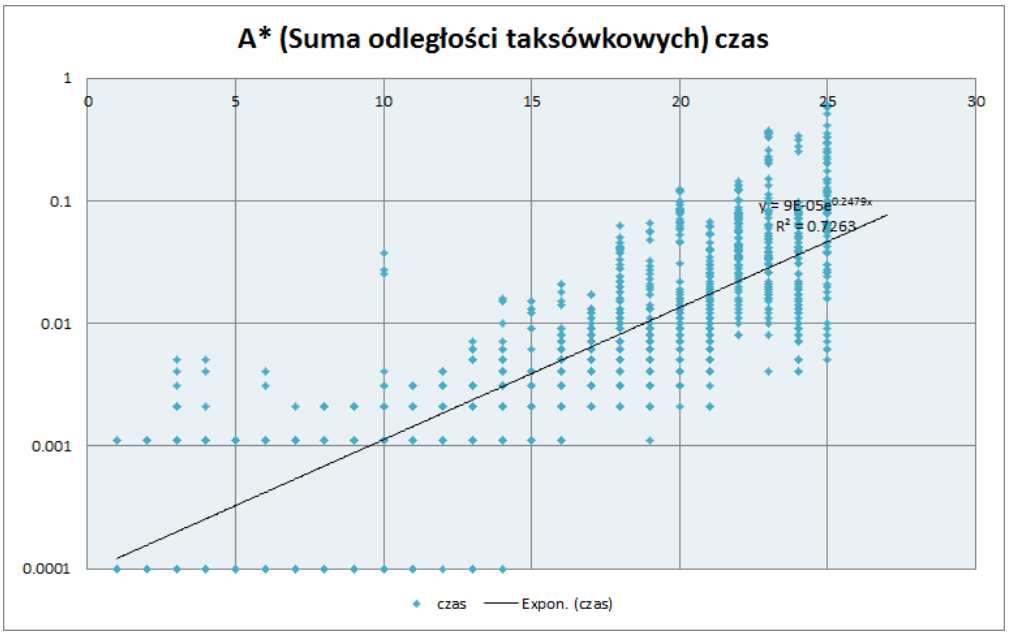
Rysunek 15. A* Suma odleglosci taksowkowych



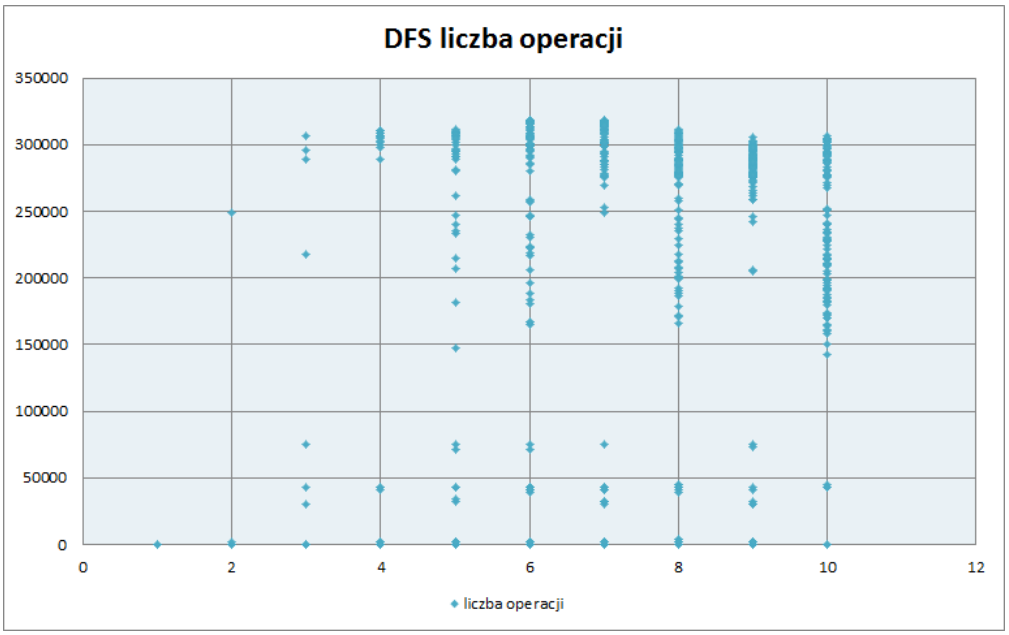
Rysunek 16. A* Suma odleglosci taksowkowych



Rysunek 17. A* Suma odleglosci taksowkowych

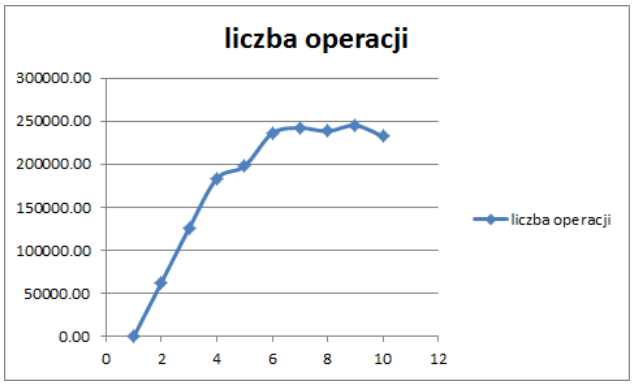


Rysunek 18. A* Suma odleglosci taksowkowych

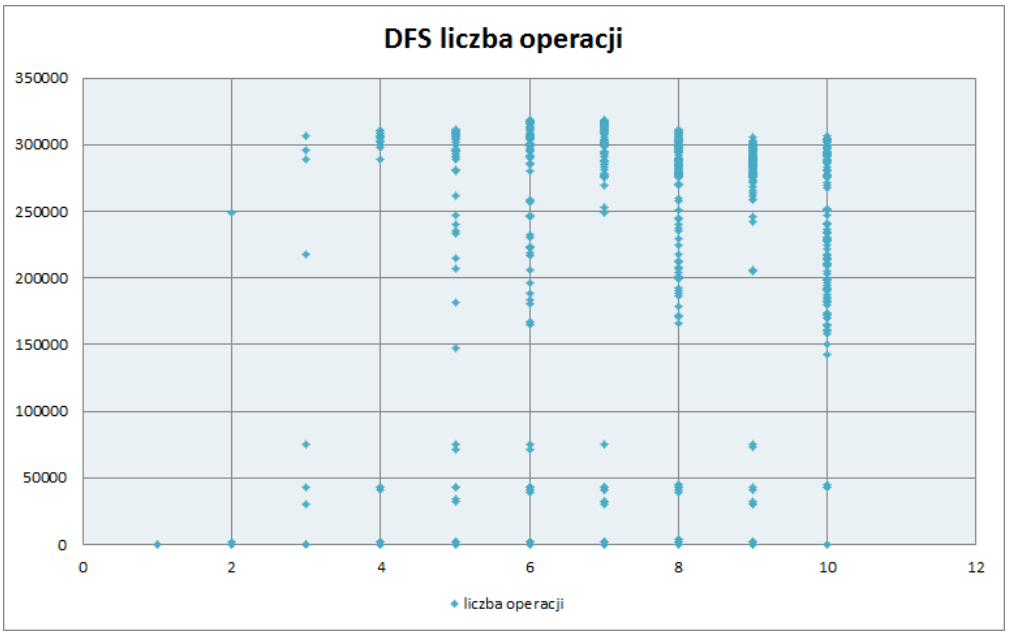


Rysunek 19. Depth-First Search

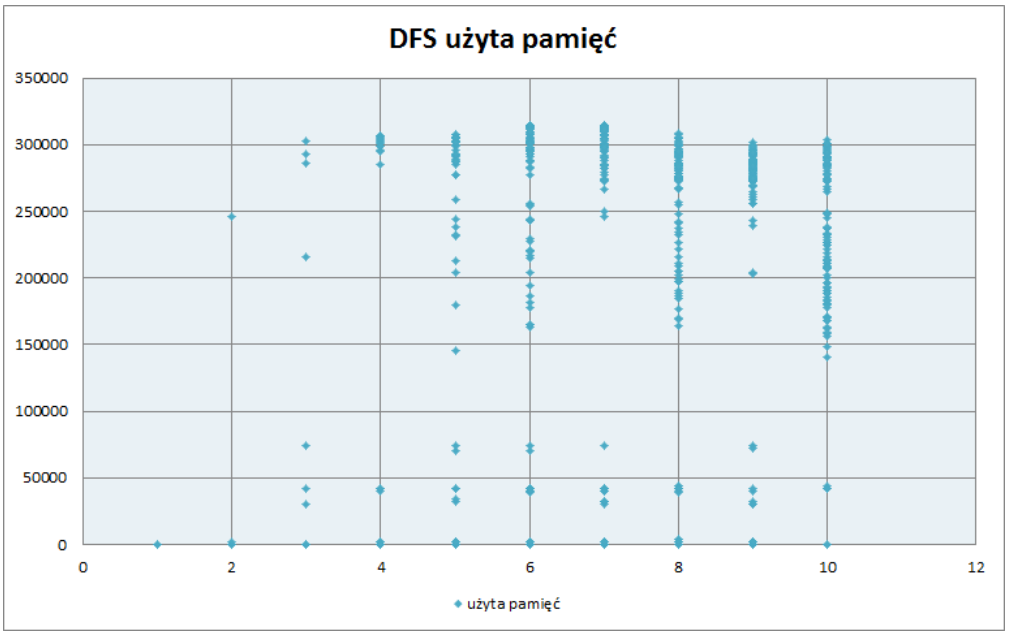
Po wyciągnięciu średniej z każdego przedziału otrzymujemy:



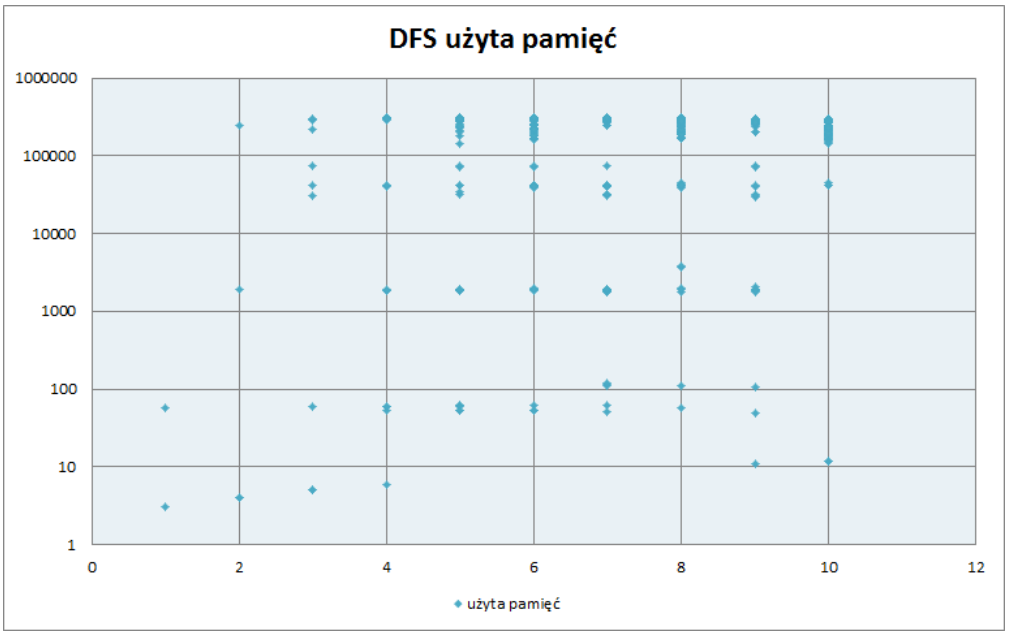
Rysunek 20. Depth-First Search



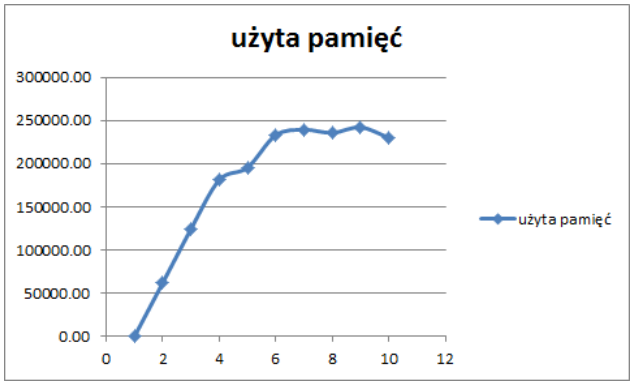
Rysunek 21. Depth-First Search



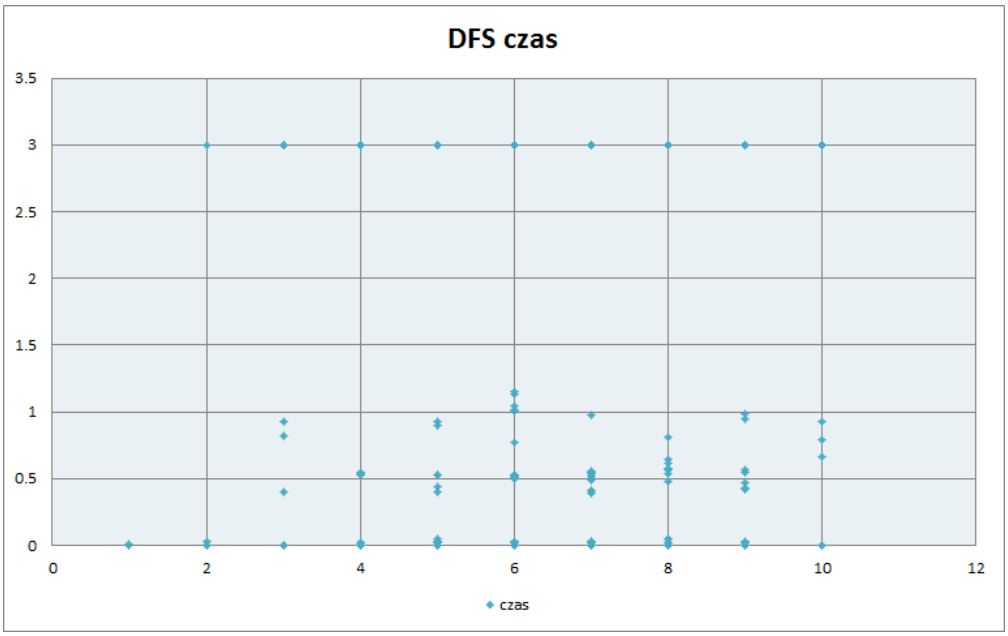
Rysunek 22. Depth-First Search



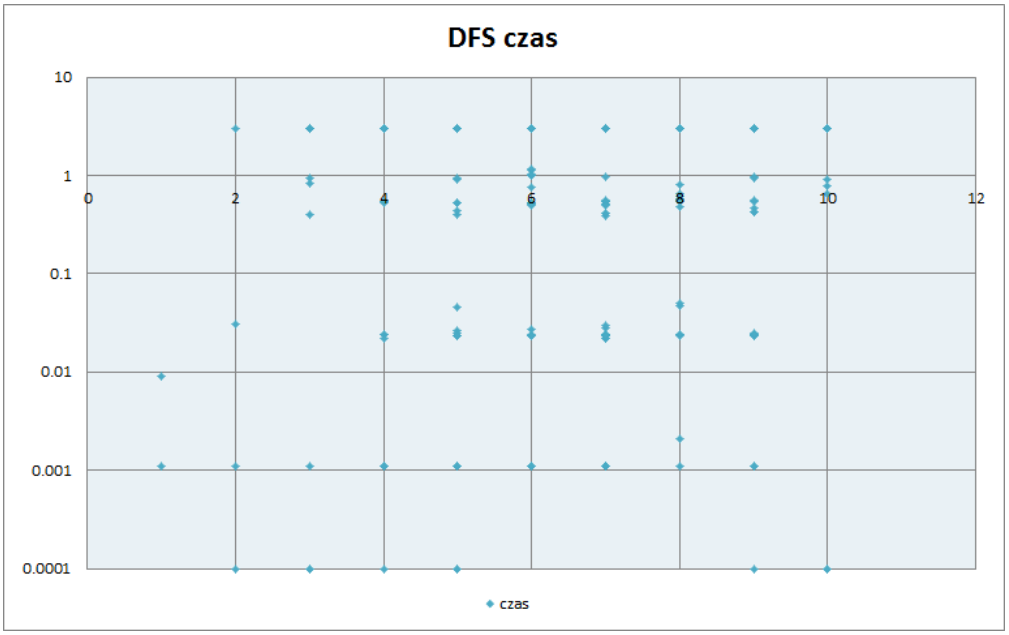
Rysunek 23. Depth-First Search



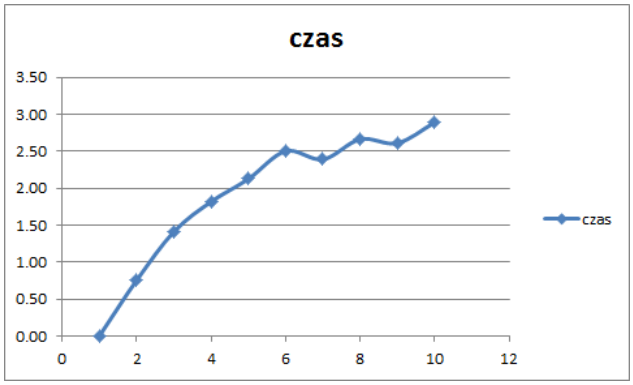
Rysunek 24. Depth-First Search



Rysunek 25. Depth-First Search



Rysunek 26. Depth-First Search



Rysunek 27. Depth-First Search

3. Wnioski

Najwydajniejszymi algorytmami okazały się, zgodnie z oczekiwaniami, algorytmy wykorzystujące heurystykę. Spowodowane jest to tym, że wybierane są tutaj drogi, które są najbliżej rozwiązania porzucając pozostałe. Funkcjonalności tej pozbawione są algorytmy nie wykorzystujące heurystyk, przez co rośnie czas poszukiwania. Najmniej wydajnym algorytmem okazał się DFS, ponieważ liczba iteracji jest tutaj największa.

Literatura

[1] T. Oetiker, H. Partl, I. Hyna, E. Schlegl. *Nie za krótkie wprowadzenie do systemu LaTeX2e*, 2007, dostępny online.

[2] Przemysław Klęsk. *Algorytmy przeszukiwania grafów i drzew dla gier i łamigłówek*, `http://wikizmsi.zut.edu.pl/uploads/b/be/2_search.pdf`

[3] Wikipedia, wolna encykolpedia *Breadth-first search*, `http://en.wikipedia.org/wiki/Breadth-first_search`

[4] Wikipedia, wolna encyklopedia *Algorytm A**, `http://pl.wikipedia.org/wiki/Algorytm_A*`