Sprawozdanie z Laboratorium

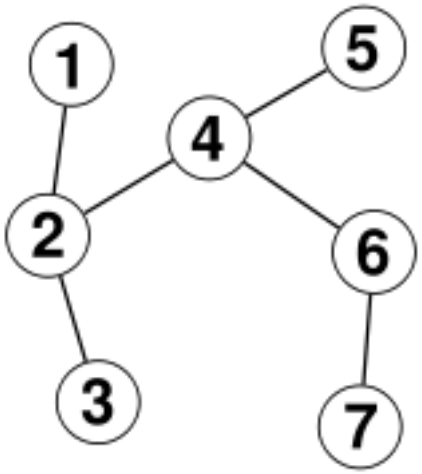
Monika Gollnik

June 3, 2014

numer albumu 200470
termin: środa 8:00-10:15
Prowadzący: mgr inż. Andrzej Wytyczak-Partyka

GRAF jest to zbiór wierzchołków, które mogą być połączone krawędziami w taki sposób, że każda krawędź kończy się i zaczyna w którymś z wierzchołków. Wierzchołki grafu zwykle są numerowane i czasem stanowią reprezentację jakichś obiektów, natomiast krawędzie mogą wówczas obrazować relacje między takimi obiektami. Krawędzie mogą mieć wyznaczony kierunek, a graf zawierający takie krawędzie nazywany jest grafem skierowanym.

Rysunek przedstawiający przykładowy graf:



Depth_first: Czyli przeszukiwanie w głab, w skrócie DFS. Polega na badaniu wszystkich krawędzi wychodzących z danego wierzchołka. Po zbadaniu wszystkich krawędzi, które wychodzą z tego wierzchołka algorytm powraca do wierzchołka, z którego dany wierzchołek został odwiedzony.

1. Złożoność pamięciowa przeszukiwania w głąb jest mała, gdyż algorytm w każdym momencie wymaga zapamiętania tylko ścieżki od korzenia do bieżącego węzła.

2. Złożoność czasowa algorytmu jest zależna od liczby wierzchołków oraz liczby krawędzi. Algorytm musi odwiedzić wszystkie wierzchołki oraz wszystkie krawędzie, co oznacza, że złożoność wynosi O(|V|+|E|).

3. Zupełność algorytm jest zupełny (czyli znajduje rozwiązanie lub informuje, że ono nie istnieje) dla drzew skończonych. Grafy skończone wymagają oznaczania już odwiedzonych wierzchołków. Dla grafów nieskończonych nie jest zupełny.

Breadth_first: Czyli przeszukiwanie w głąb, w skrócie BFS. Jeden z najprostszych algorytmów przeszukiwania grafu. Przechodzenie grafu rozpoczyna się od zadanego wierzchołka p i polega na odwiedzeniu wszystkich osiągalnych z niego wierzchołków. Wykorzystywany jest do odnajdywania najkrótszej drogi w grafie. Wynikiem działania algorytmu jest także drzewo przeszukiwania wszerz o korzeniu w p, zawierające wszystkie wierzchołki do których prowadzi droga z p.

1. Złożoność pamięciowa algorytmu uzależniona jest od tego w jaki sposób reprezentowany jest graf wejściowy. W przypadku listy sąsiedztwa dla każdego wierzchołka przechowywana jest lista wierzchołków osiągalnych bezpośrednio z niego. W tym wypadku złożoność pamięciowa wynosi O(|V|+|E|), gdzie |V| to liczba węzłów, a |E| to liczba krawędzi w grafie, odpowiadająca sumie wierzchołków znajdujących się na listach sąsiedztwa. Zaś w przypadku macierzy sąsiedztwa wymagane jest przechowywanie macierzy o wymiarach |V|x|V|, czyli potrzebne jest O(|V|2) pamięci.

2. Złożoność czasowa w przypadku przeszukiwanie wszerz musi przebyć wszystkie krawędzie prowadzące do wszystkich węzłów, złożoność czasowa przeszukiwania wszerz wynosi O(|V|+|E|), gdzie |V| to liczba węzłów, a |E| to liczba krawędzi w grafie.

3. Przeszukiwanie wszerz jest kompletne, to znaczy że gdy istnieje rozwiązanie, przeszukiwanie wszerz odnajdzie je niezależnie od grafu.

Algorytm A* – algorytm heurystyczny znajdowania najkrótszej ścieżki w grafie z dowolnego wierzchołka do wierzchołka spełniającego określony warunek zwany testem celu. Algorytm jest zupełny i optymalny, w tym sensie, że znajduje ścieżkę, jeśli tylko taka istnieje, i przy tym jest to ścieżka najkrótsza.

1. Złożoność czasowa algorytmu A* zależy od zastosowanej heurystyki. W najgorszym przypadku liczba przeszukanych węzłów rośnie wykładniczo w stosunku do długości rozwiązania, natomiast rośnie już tylko wielomianowo, jeśli funkcja heurystyki h spełnia następujący warunek: |h(x) - h^*(x)| = O(\log h^*(x)).

Przykładowe testy sprawdzające poprawność działania przeszukiwań grafów:
Wyniki testu I dla grafu:
Dla Depth_first:
0->1->2->3->4->5
czas wyszukiwania to: 0.000153s
Dla Breadth_first
0->2->3->5
czas wyszukiwania to: 1.8e-05s
Dla A*:
0->2->3->5
czas wyszukiwania to: 0.000114s
-
Wyniki testu II dla grafu:
Dla Depth_first:
0->1->2->3->4->5
czas wyszukiwania to: 0.000169s
Dla Breadth_first
0->2->3->5
czas wyszukiwania to: 1.9e-05s
Dla A*:
0->2->3->5
czas wyszukiwania to: 0.000115s

Wnioski:

1. Przy małych ilościach danych (krawędziach, wierzchołkach) bardziej efektywniejsze jest przeszukiwanie Depth first i Breadth_first, niż A*.

2. Jak widać algorytm A* znajduję najkrótszą ścieżkę zatem funkcja heurystyczna zwraca 0.

3. Algorytm A* w porównaniu z Depth first i Breadth_first działa znacznie szybciej przy dużej ilości krawędzi i wierzchołków.

4. W obu testach wyszukiwania poszczególnych grafów przebiegły po tych samych ścieżkach.

5. Algorytm A* oraz Breadth_first znajdują najkrótszą ścieżkę, choć czas wyszukiwania w obu testach dla A* jest znacznie krótszy.

6. Najmniejsza różnica w czasie, w obu testach jest w przypadku wyszukiwania grafu metodą A*.