Algorytm Dijkstry

Jest to algorytm służący do znajdowania najkrótszych ścieżek między węzłami w grafie. Został opracowany przez informatyka Edsgera W. Dijkstrę w 1956 roku i opublikowany trzy lata później. Dla danego węzła źródłowego w grafie algorytm wyznacza najkrótszą ścieżkę między tym węzłem, a pojedynczym węzłem docelowym w grafie, poprzez zatrzymanie algorytmu po określeniu najkrótszej ścieżki do węzła docelowego. Algorytm na początku swojego działania oznacza wszystkie węzły jako nieodwiedzone. Następnie uzupełnia odległość do każdego węzła. Wszystkie odległości są początkowo ustawione na nieskończoność, z wyjątkiem węzła początkowego, którego odległość zostaje zapisana jako zero. Wstępną odległość węzła v to długość najkrótszej odkrytej dotychczas ścieżki między węzłem v, a węzłem początkowym. Następnie algorytm dla bieżącego węzła rozważa jego wszystkich nieodwiedzonych sąsiadów i oblicza odległość do nich. Algorytm następnie porównuje nowo obliczone wartość z bieżąco wpisanymi i wybiera mniejszą. Kiedy zakończone zostanie rozpatrywanie wszystkich sąsiadów bieżącego węzła, zostaje on oznaczony jako odwiedzony i usunięty ze zbioru punktów nieodwiedzonych. Taki węzeł nie rozstanie już ponownie sprawdzony. Algorytm przechodzi tak przez wszystkie nieodwiedzone punkty, a następnie kończy swoją pracę.

Dane wejściowe:

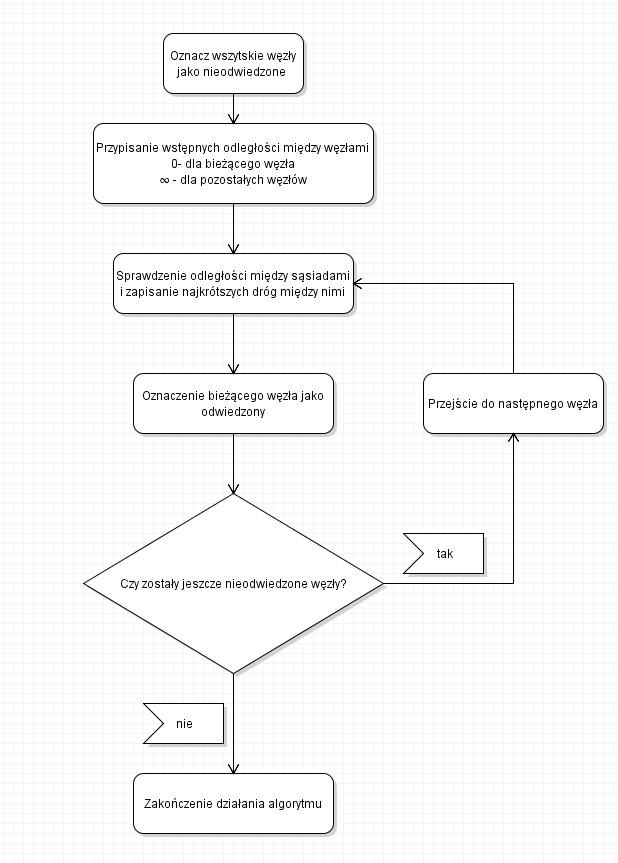
-macierz koincydencji

Dane wyjściowe:

-najkrótsze odległości między węzłami

Złożoność czasowa: Θ(nlog n)

Złożoność pamięciowa: Θ(n+nlog n)



Algorytm najbliższego sąsiada

Ten algorytm wykorzystuje strategie zachłanną. Rozpoczyna swoją pracę od wierzchołka oznaczonego jako startowy. Ten wierzchołek zostaje oznaczony jako odwiedzony i aktualnie odwiedzany. Następnie algorytm sprawdza, który z pozostałych nieodwiedzonych punktów zaznaczonych przez użytkownika jest najbliżej za pomocą algorytmu Dijkstry. Po wyborze najkrótszej drogi zostaje ona dołączona do rozwiązania, a punkt do którego prowadzi ta droga zostaje zaznaczony jako odwiedzony i jest on aktualną pozycją wyjściową do szukania następnej drogi do punktów nieodwiedzonych. Algorytm powtarza swoje działanie do momentu, aż wszystkie punkty zaznaczone przez użytkowania zostaną odwiedzone. Gdy dojdzie do tego momentu wyznaczona zostaje trasa od ostatnio odwiedzonego punktu do punktu startowego. Po dodaniu tej drogi, algorytm zwraca pełną marszrutę.

Dane wejściowe:

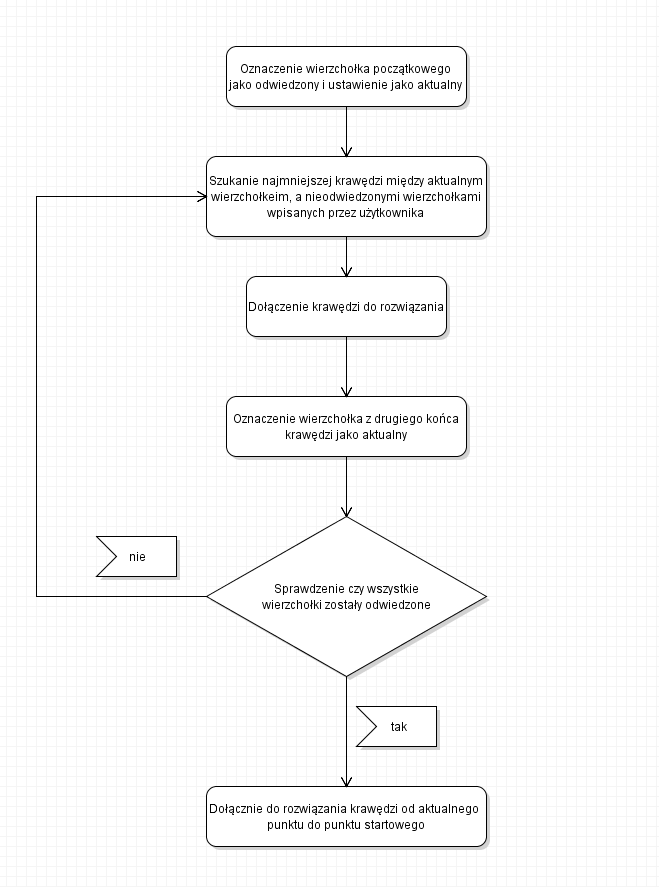
-macierz koincydencji

Dane wyjściowe:

-Marszruta stworzona z punktów docelowych

Złożoność czasowa: Θ(N2)

Złożoność pamięciowa: Θ(N)



Algorytm brute force (algorytm dokładny)

Jest to algorytm szukający wszystkich możliwych permutacji zbioru punktów początkowych. Algorytm w trakcie szukania rozwiązań korzysta z indeksów punktów docelowych, zapisując ich każda kombinację. Następnie na początku i na końcu każdej permutacji dodawany jest indeks punktu początkowego, aby uzyskać pełną trasę. Kolejnym krokiem algorytmu jest porównanie długości każdej drogi w celu wybrania najkrótszej z nich. Algorytm do porównania długości marszrut korzysta z algorytmu Dijkstry, który zwraca najkrótszą drogę między 2 punktami. Na początku porównywania długości algorytm wybiera pierwszą trasę jako najkrótszą, a następnie porównuje do niej kolejne drogi. Jeśli dana trasa jest krótsza od aktualnie oznaczonej jako najkrótsza zastępuje jej miejsce, a algorytm przechodzi tak przez dalsze drogi. Po wybraniu najkrótszej marszruty otrzymujemy ją w postaci listy punktów.

Obraz zawierający akcesorium, parasol

Opis wygenerowany automatycznie

Zakładając wierzchołek z numerem 1 jako początkowy z powyższego grafu możemy wyszczególnić marszruty:

-1,2,3,4

-1,2,4,3

-1,3,2,4

-1,3,4,2

-1,4,2,3

-1,4,3,2

Dane wejściowe:

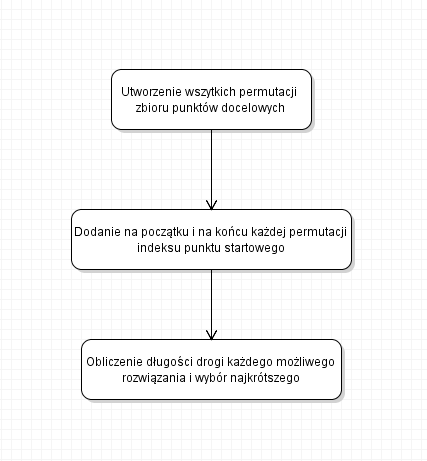
-macierz koincydencji

Dane wyjściowe:

-Marszruta stworzona z punktów docelowych

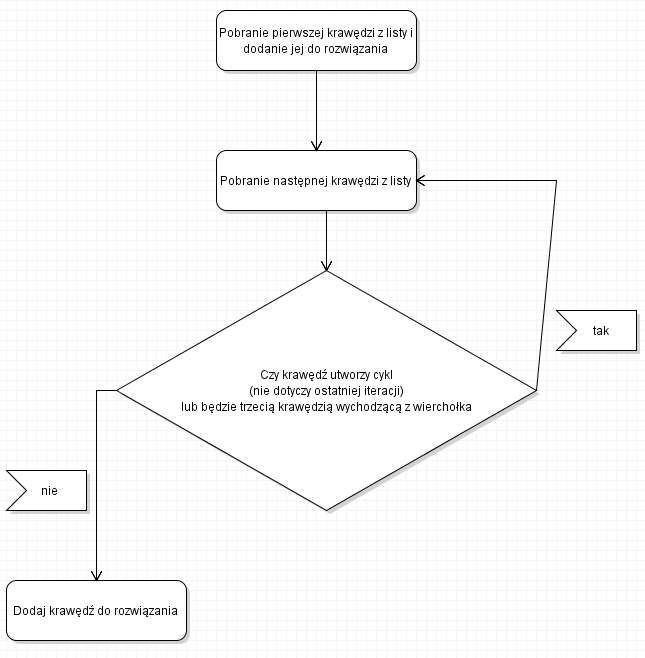
Złożoność czasowa: Θ(N!)

Złożoność pamięciowa: Θ(N!)

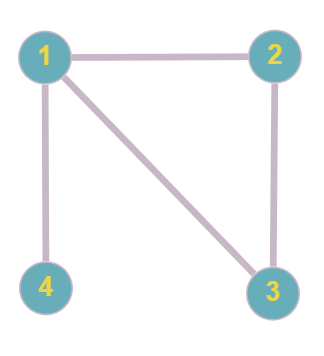
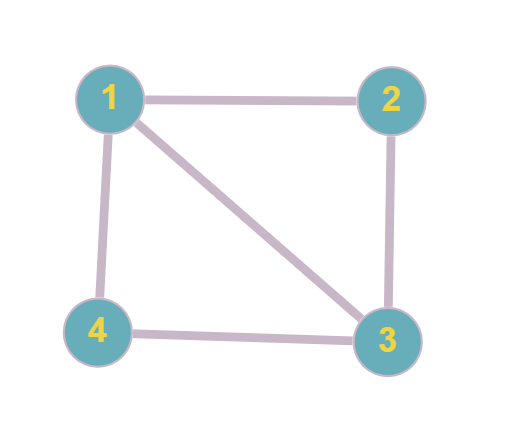


Algorytm najmniejszej krawędzi

Ten algorytm polega na kolejnym dołączaniu do rozwiązania najkrótszych spośród dopuszczalny krawędzi. Początkowo algorytm zapisuje wszystkie krawędzie z ich długościami między wszystkimi punktami docelowymi w liście, a następnie sortuje je rosnąco względem długości. Algorytm wybiera kolejne krawędzi z listy i próbuje dołączyć je do rozwiązania. Podczas próby dołączenia krawędzi do rozwiązania algorytm sprawdza czy nie spowoduje ona utworzenia cyklu (nie dotyczy ostatniej iteracji) lub powstania wierzchołka, z którego wychodzą trzy krawędzie, jeśli dana krawędź spełnia chociaż jeden z tych warunków nie będzie brana pod uwagę w rozwiązaniu, a algorytm przejdzie do następnej krawędzi z listy. Po utworzeniu połączeń między punktami docelowymi algorytm ustawia je tak, aby pierwszym punktem na liście wynikowej był punkt początkowy, a następnie kolejne punkty które łączą krawędzie. Po wykonaniu tego wszystkiego algorytm zwraca marszrutę stworzoną z punktów docelowych.

Schemat krokowy:

Niedopuszczlane sytuacje:

Obraz zawierający tekst, zegar

Opis wygenerowany automatycznie

Dane wejściowe:

-macierz koincydencji

Dane wyjściowe:

-Marszruta stworzona z punktów docelowych

Złożoność czasowa: Θ(n2log n)

Złożoność pamięciowa: Θ(n2log n)