5b. Algorytmy znajdowania najkrótszych dróg

Grzegorz Kosiorowski

Uniwersytet Ekonomiczny w Krakowie

Motywacja

Typowym zagadnieniem związanym z grafami (zwłaszcza skierowanymi i z wagami) jest znalezienie najkrótszej drogi z jednego wierzchołka do drugiego.

Motywacja

Typowym zagadnieniem związanym z grafami (zwłaszcza skierowanymi i z wagami) jest znalezienie najkrótszej drogi z jednego wierzchołka do drugiego.

- Nawigacja samochodowa.

Motywacja

Typowym zagadnieniem związanym z grafami (zwłaszcza skierowanymi i z wagami) jest znalezienie najkrótszej drogi z jednego wierzchołka do drugiego.

- Nawigacja samochodowa.
- Gry komputerowe: zarządzanie poruszaniem się przeciwników gracza.

Motywacja

Typowym zagadnieniem związanym z grafami (zwłaszcza skierowanymi i z wagami) jest znalezienie najkrótszej drogi z jednego wierzchołka do drugiego.

- Nawigacja samochodowa.
- Gry komputerowe: zarządzanie poruszaniem się przeciwników gracza.
- Procesy produkcyjne: w grafie, który przedstawia alternatywne sposoby dotarcia do wyznaczonego celu możemy poszukiwać np. ścieżki o najmniejszym koszcie.

Algorytm przeszukiwania wszerz - założenia

Najpierw rozważmy przypadek prostszy: jak wyznaczyć najkrótszą drogę z wybranego wierzchołka do każdego innego w spójnym grafie skierowanym lub nieskierowanym bez wag? Zazwyczaj rozwiązuje się to zagadnienie przy pomocy tak zwanego **algorytmu przeszukiwania wszerz** (formalnie można dodać: ze wskaźnikami).

Algorytm przeszukiwania wszerz - założenia

Najpierw rozważmy przypadek prostszy: jak wyznaczyć najkrótszą drogę z wybranego wierzchołka do każdego innego w spójnym grafie skierowanym lub nieskierowanym bez wag? Zazwyczaj rozwiązuje się to zagadnienie przy pomocy tak zwanego **algorytmu przeszukiwania wszerz** (formalnie można dodać: ze wskaźnikami). Dla wygody, zaczynamy od ponumerowania w dowolny sposób wierzchołków takiego grafu $V(G) = \{1, \ldots, n\}$. Poszukujemy najkrótszej drogi z wierzchołka 1 do każdego innego wierzchołka j.

Algorytm przeszukiwania wszerz - założenia

Najpierw rozważmy przypadek prostszy: jak wyznaczyć najkrótszą drogę z wybranego wierzchołka do każdego innego w spójnym grafie skierowanym lub nieskierowanym bez wag? Zazwyczaj rozwiązuje się to zagadnienie przy pomocy tak zwanego **algorytmu przeszukiwania wszerz** (formalnie można dodać: ze wskaźnikami). Dla wygody, zaczynamy od ponumerowania w dowolny sposób wierzchołków takiego grafu $V(G) = \{1, \ldots, n\}$. Poszukujemy najkrótszej drogi z wierzchołka 1 do każdego innego wierzchołka j. Uwaga! Pozorne uproszczenie zadania do „znajdź najkrótszą drogę z zadanego wierzchołka do innego zadanego wierzchołka" (a nie do wszystkich innych wierzchołków) nie ułatwia rozwiązania - najprościej jest użyć algorytmu przeszukiwania wszerz i przerwać go, gdy znajdziemy szukaną drogę.

Algorytm przeszukiwania wszerz ze wskaźnikami

Algorytm przeszukiwania wszerz

Dane: Graf spójny $G = (V(G), E(G))$, ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$.
Zmienne: L, S i V - zbiory wierzchołków, d, p - funkcje, i, j - liczby pomocnicze.

Algorytm przeszukiwania wszerz ze wskaźnikami

Algorytm przeszukiwania wszerz

Dane: Graf spójny $G = (V(G), E(G))$, ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$.
Zmienne: L, S i V - zbiory wierzchołków, d, p - funkcje, i, j - liczby pomocnicze.

- I. $L := \{1\}$, $V := \{2, \ldots n\}$. $d(1) = 0$, $p(1) = \emptyset$, $S := \emptyset$.

Algorytm przeszukiwania wszerz ze wskaźnikami

Algorytm przeszukiwania wszerz

Dane: Graf spójny $G = (V(G), E(G))$, ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$.
Zmienne: L, S i V - zbiory wierzchołków, d, p - funkcje, i, j - liczby pomocnicze.

- I. $L := \{1\}$, $V := \{2, \ldots n\}$. $d(1) = 0$, $p(1) = \emptyset$, $S := \emptyset$.
- II. Dopóki $L \neq V(G)$ wykonuj:

Algorytm przeszukiwania wszerz ze wskaźnikami

Algorytm przeszukiwania wszerz

Dane: Graf spójny $G = (V(G), E(G))$, ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$.
Zmienne: L, S i V - zbiory wierzchołków, d, p - funkcje, i, j - liczby pomocnicze.

- I. $L := \{1\}$, $V := \{2, \ldots n\}$. $d(1) = 0$, $p(1) = \emptyset$, $S := \emptyset$.
- II. Dopóki $L \neq V(G)$ wykonuj:
- II.1 Dla $j \in V \setminus L$ wykonuj: jeśli istnieje $i \in L$ takie, że $ij \in E(G)$, dołącz j do S i zdefiniuj $d(j) = d(i) + 1$ oraz $p(j) = i$.

Algorytm przeszukiwania wszerz ze wskaźnikami

Algorytm przeszukiwania wszerz

Dane: Graf spójny $G = (V(G), E(G))$, ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$.
Zmienne: L, S i V - zbiory wierzchołków, d, p - funkcje, i, j - liczby pomocnicze.

- I. $L := \{1\}$, $V := \{2, \ldots n\}$. $d(1) = 0$, $p(1) = \emptyset$, $S := \emptyset$.
- II. Dopóki $L \neq V(G)$ wykonuj:
- II.1 Dla $j \in V \setminus L$ wykonuj: jeśli istnieje $i \in L$ takie, że $ij \in E(G)$, dołącz j do S i zdefiniuj $d(j) = d(i) + 1$ oraz $p(j) = i$.
- II.2 $L := L \cup S$, $S := \emptyset$.

Algorytm przeszukiwania wszerz ze wskaźnikami

Algorytm przeszukiwania wszerz

Dane: Graf spójny $G = (V(G), E(G))$, ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$.
Zmienne: L, S i V - zbiory wierzchołków, d, p - funkcje, i, j - liczby pomocnicze.

- I. $L := \{1\}$, $V := \{2, \ldots n\}$. $d(1) = 0$, $p(1) = \emptyset$, $S := \emptyset$.
- II. Dopóki $L \neq V(G)$ wykonuj:
- II.1 Dla $j \in V \setminus L$ wykonuj: jeśli istnieje $i \in L$ takie, że $ij \in E(G)$, dołącz j do S i zdefiniuj $d(j) = d(i) + 1$ oraz $p(j) = i$.
- II.2 $L := L \cup S$, $S := \emptyset$.
- **Rezultat**: $d(j)$ to minimalna długość drogi z 1 do j, a $p(j)$ to wskaźnik, który pokazuje poprzedni etap drogi z 1 do j.

Algorytm wyszukiwania wszerz - uwagi.

Minimalną długość drogi otrzymujemy od razu, natomiast nad samą drogą trzeba jeszcze chwilę popracować. W ciągu j, $p(j)$, $p(p(j))$, ... występują wierzchołki drogi minimalnej od 1 do j w odwrotnej kolejności - dlatego wystarczy ten ciąg wypisać od tyłu i już droga (jako ciąg wierzchołków) jest znaleziona.

Algorytm wyszukiwania wszerz - uwagi.

Minimalną długość drogi otrzymujemy od razu, natomiast nad samą drogą trzeba jeszcze chwilę popracować. W ciągu j, $p(j)$, $p(p(j))$, ... występują wierzchołki drogi minimalnej od 1 do j w odwrotnej kolejności - dlatego wystarczy ten ciąg wypisać od tyłu i już droga (jako ciąg wierzchołków) jest znaleziona.
Tłumacząc algorytm na ludzki: Szukamy najpierw wierzchołków sąsiadujących ze startowym i przypisujemy im odległość 1.

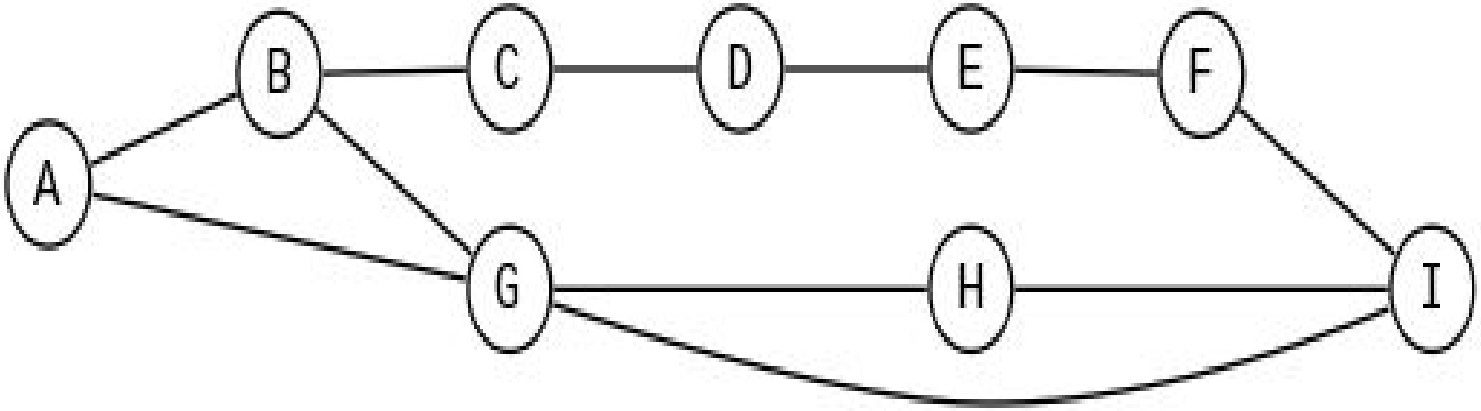
Algorytm wyszukiwania wszerz - uwagi.

Minimalną długość drogi otrzymujemy od razu, natomiast nad samą drogą trzeba jeszcze chwilę popracować. W ciągu j, $p(j)$, $p(p(j))$, ... występują wierzchołki drogi minimalnej od 1 do j w odwrotnej kolejności - dlatego wystarczy ten ciąg wypisać od tyłu i już droga (jako ciąg wierzchołków) jest znaleziona.
Tłumacząc algorytm na ludzki: Szukamy najpierw wierzchołków sąsiadujących ze startowym i przypisujemy im odległość 1. Następnie szukamy wierzchołków, których odległość nie została jeszcze wyznaczona, a sąsiadują z wierzchołkami o odległości 1 i przypisujemy im odległość 2.

Algorytm wyszukiwania wszerz - uwagi.

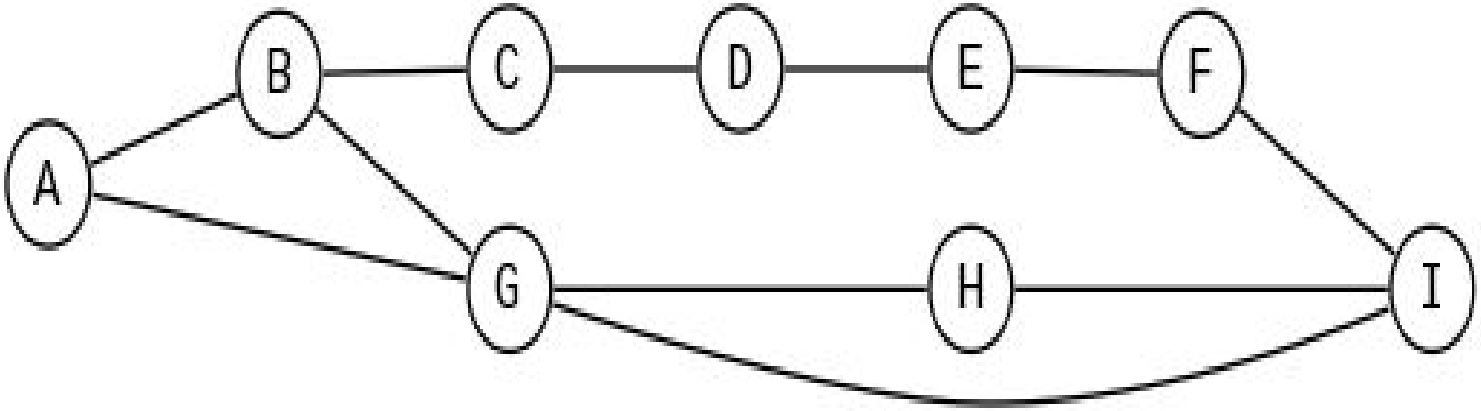
Minimalną długość drogi otrzymujemy od razu, natomiast nad samą drogą trzeba jeszcze chwilę popracować. W ciągu j, $p(j)$, $p(p(j))$, ... występują wierzchołki drogi minimalnej od 1 do j w odwrotnej kolejności - dlatego wystarczy ten ciąg wypisać od tyłu i już droga (jako ciąg wierzchołków) jest znaleziona.
Tłumacząc algorytm na ludzki: Szukamy najpierw wierzchołków sąsiadujących ze startowym i przypisujemy im odległość 1. Następnie szukamy wierzchołków, których odległość nie została jeszcze wyznaczona, a sąsiadują z wierzchołkami o odległości 1 i przypisujemy im odległość 2. Następnie szukamy wierzchołków, których odległość nie została jeszcze wyznaczona, a sąsiadują z wierzchołkami o odległości 2... i tak dalej, aż przejdziemy cały graf. Za każdym razem, gdy przypisujemy wierzchołkowi jego odległość, zapisujemy z jakim wierzchołkiem o odległości o jeden mniejszej sąsiaduje - i to jest jego wskaźnik.

Algorytm przeszukiwania wszerz - przykład



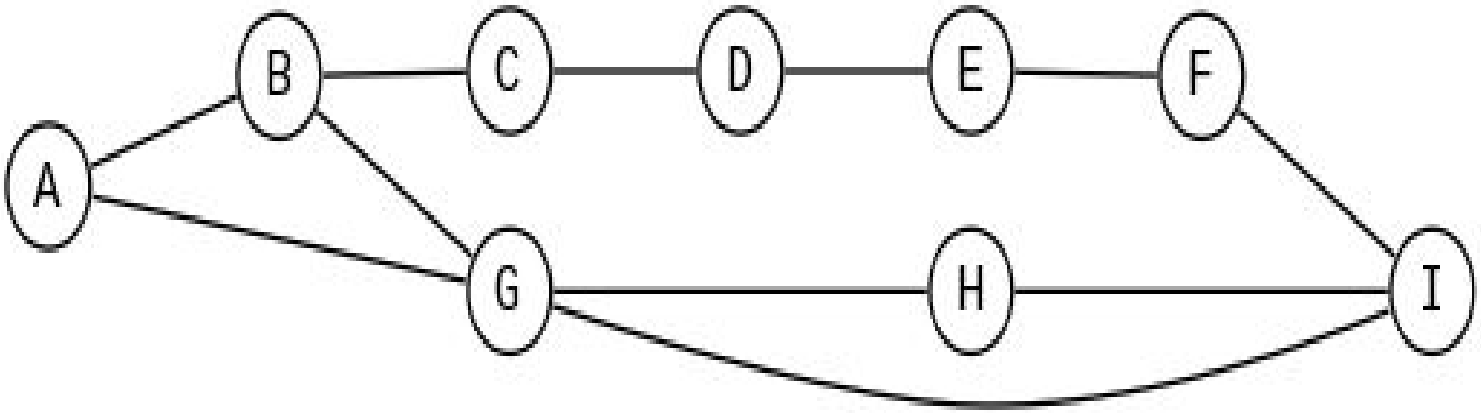
Spróbujemy zastosować algorytm przeszukiwania wszerz dla powyższego grafu nieskierowanego. Przedstawię tutaj sposób postępowania przyjęty w ramach tego kursu - obowiązujący na sprawdzianie/egzaminie.

Algorytm przeszukiwania wszerz - przykład



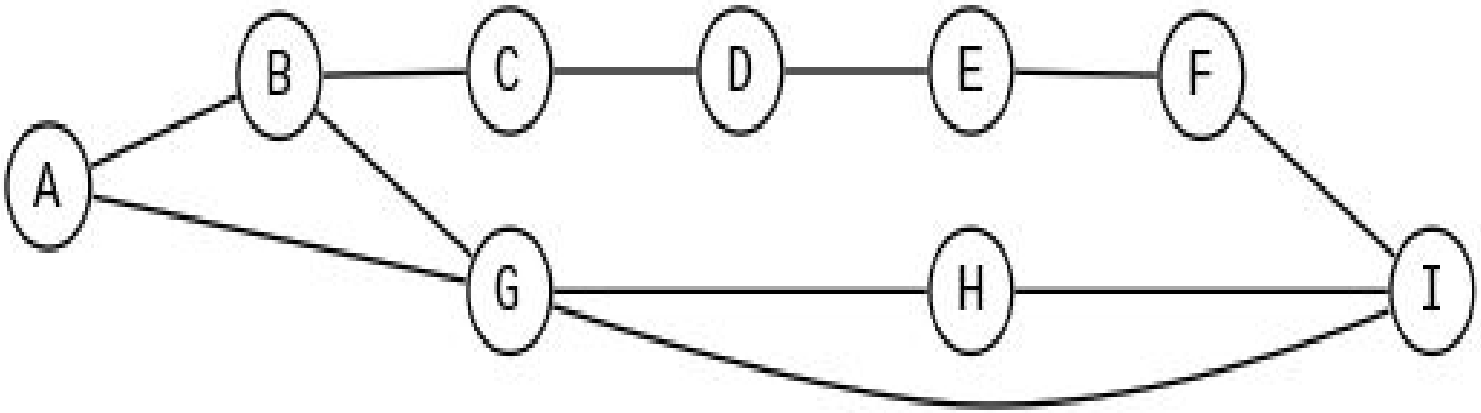
Spróbujemy zastosować algorytm przeszukiwania wszerz dla powyższego grafu nieskierowanego. Przedstawię tutaj sposób postępowania przyjęty w ramach tego kursu - obowiązujący na sprawdzianie/egzaminie.
Graf ten ewidentnie jest spójny, więc algorytm zadziała. Algorytm zaczynamy od ponumerowania wierzchołków w kolejności alfabetycznej: A-1, B-2 itd... Będziemy obliczać minimalną długość drogi wszystkich wierzchołków od A.

Algorytm przeszukiwania wszerz - przykład



Odległość danego wierzchołka X od A oznaczamy $d(X)$, a wskaźnik wierzchołka to $p(X)$. Zbiór L (zielone) to zbiór elementów, których odległości od A mamy policzone. Zbiór S (czerwone) to zbiór sąsiadów tych wierzchołków w danym etapie, których odległości obliczamy.

Algorytm przeszukiwania wszerz - przykład

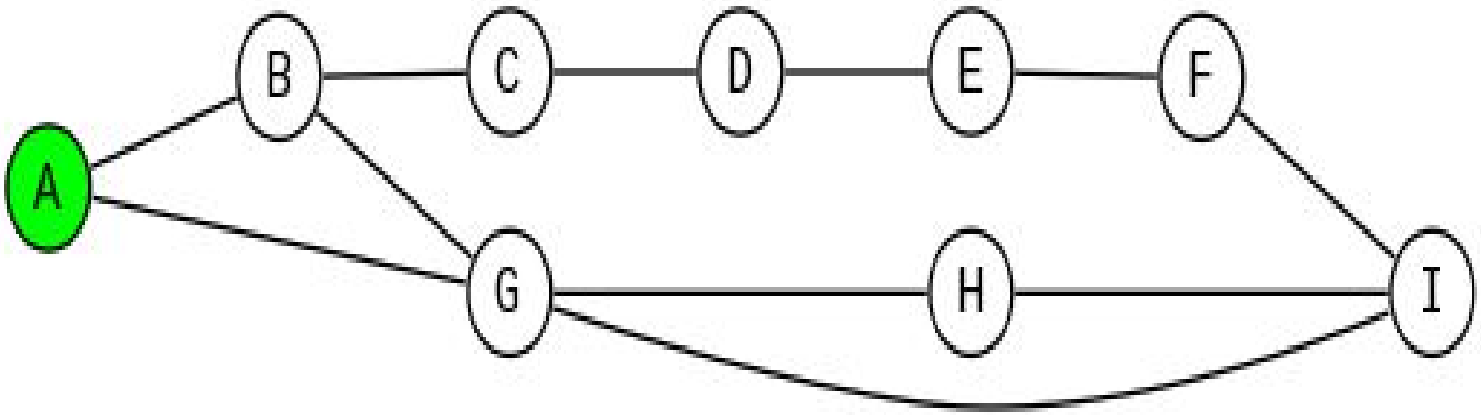


Odległość danego wierzchołka X od A oznaczamy $d(X)$, a wskaźnik wierzchołka to $p(X)$. Zbiór L (zielone) to zbiór elementów, których odległości od A mamy policzone. Zbiór S (czerwone) to zbiór sąsiadów tych wierzchołków w danym etapie, których odległości obliczamy.

Działanie algorytmu zapisujemy w takiej tabeli:

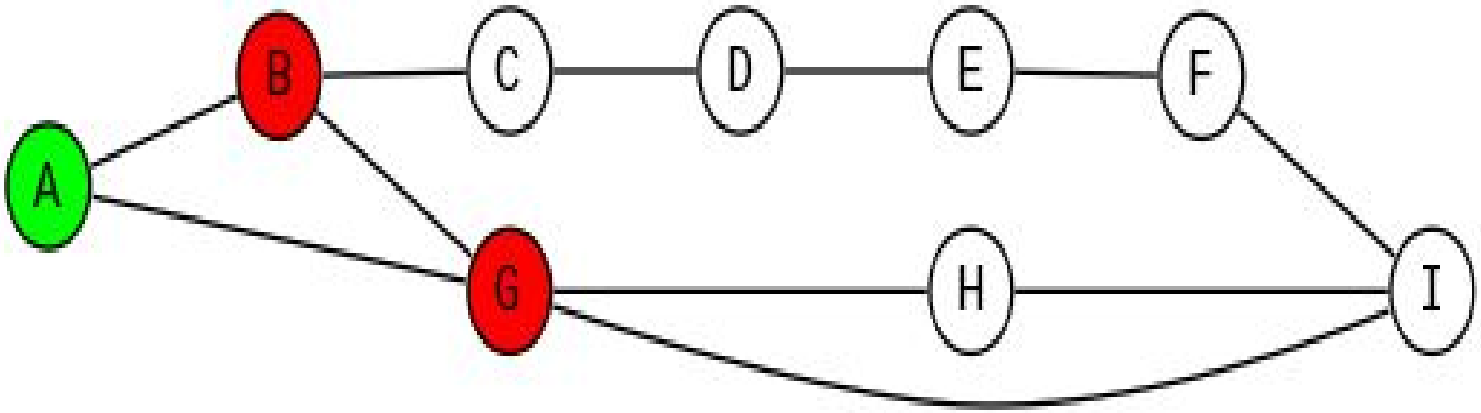
Nr etapu	zbiór L	zbiór S	uzyskane odległości	wskaźniki
1				
2				

Algorytm przeszukiwania wszerz - krok1



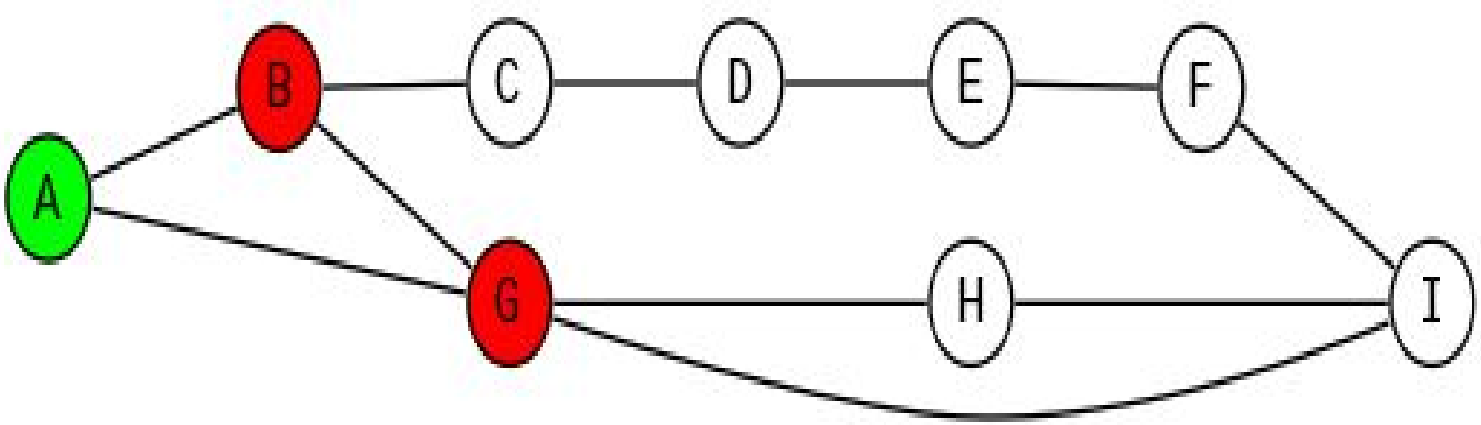
Na początku w zbiorze L jest tylko wierzchołek A. Szukamy sąsiadów wierzchołka A, którzy nie są w zbiorze L i przesuwamy ich do zbioru S.

Algorytm przeszukiwania wszerz - krok1



Skoro to pierwszy krok, obydwu sąsiadom A przypisujemy odległość 1. Ich sąsiadem będącym
w zbiorze L jest A, więc ich wskaźnikiem jest A.

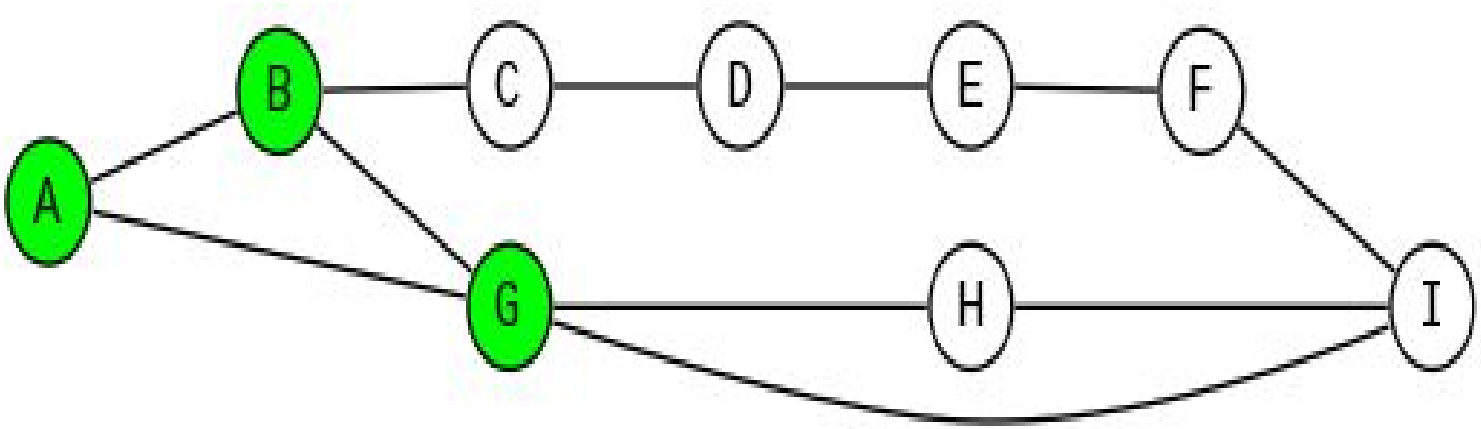
Algorytm przeszukiwania wszerz - krok1



Skoro to pierwszy krok, obydwu sąsiadom A przypisujemy odległość 1. Ich sąsiadem będącym w zbiorze L jest A, więc ich wskaźnikiem jest A. Zapisujemy to w tabeli:

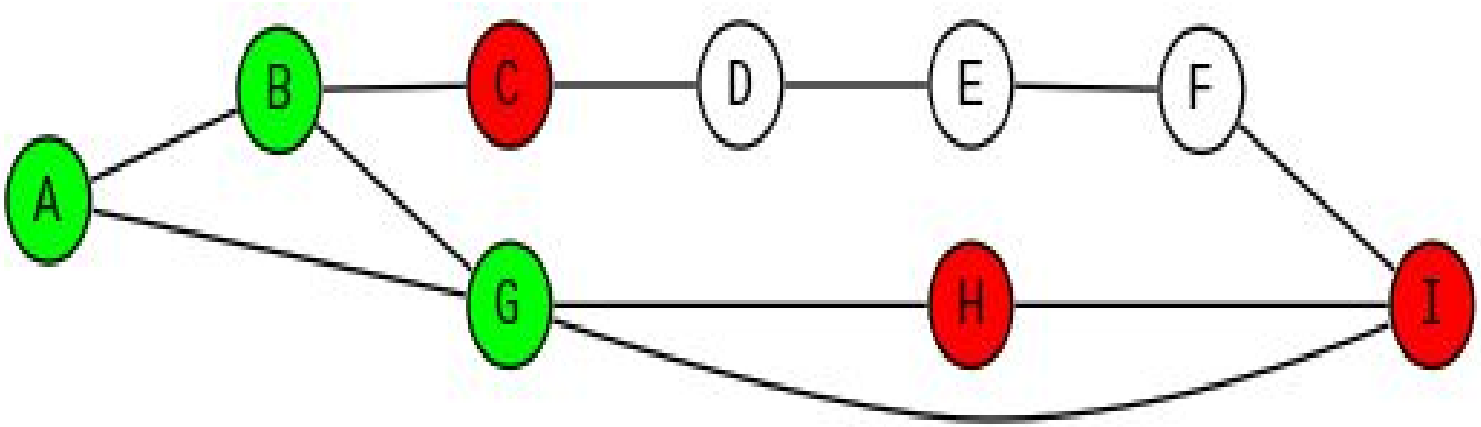
Nr etapu	zbiór L	zbiór S	uzyskane odległości	wskaźniki
1	A	B,G	$d(B) = d(G) = 1$	$p(B) = p(G) = A$

Algorytm przeszukiwania wszerz - krok2



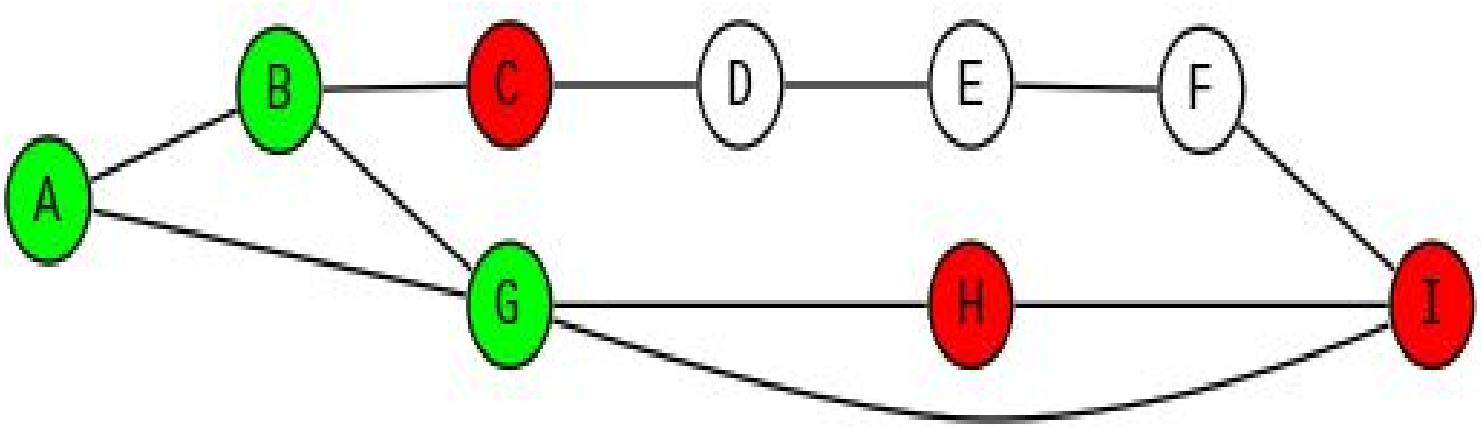
Dołączamy wierzchołki z S do zbioru L i szukamy ich sąsiadów, którzy nie są w zbiorze L. Utworzą oni zbiór S.

Algorytm przeszukiwania wszerz - krok2



Skoro to drugi krok, wierzchołkom z S przypisujemy odległość 2. Ich wskaźnikami są ich sąsiedzi ze zbioru L.

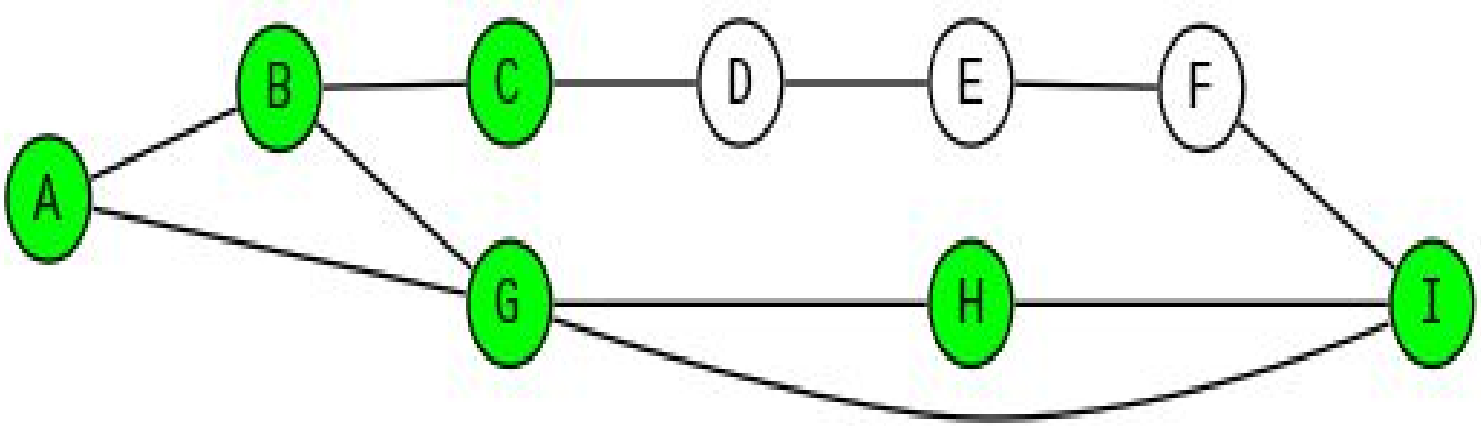
Algorytm przeszukiwania wszerz - krok2



Skoro to drugi krok, wierzchołkom z S przypisujemy odległość 2. Ich wskaźnikami są ich sąsiedzi ze zbioru L.

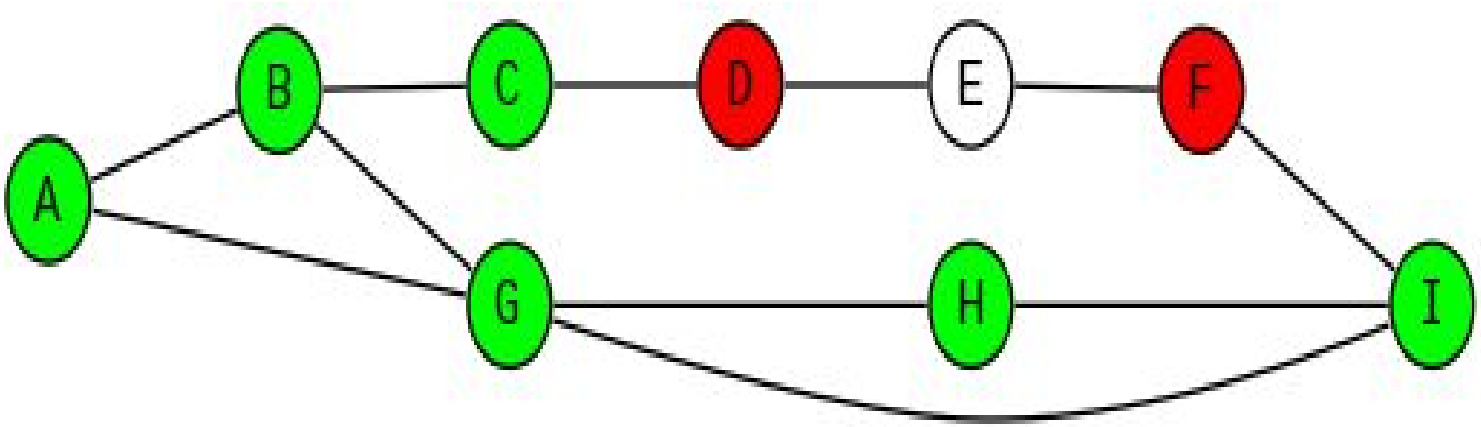
Nr	zbiór L	zbiór S	uzyskane odległości	wskaźniki
2	A,B,G	C,H,I	$d(C) = d(H) = d(I) = 2$	$p(C) = B$, $p(H) = p(I) = G$

Algorytm przeszukiwania wszerz - krok3



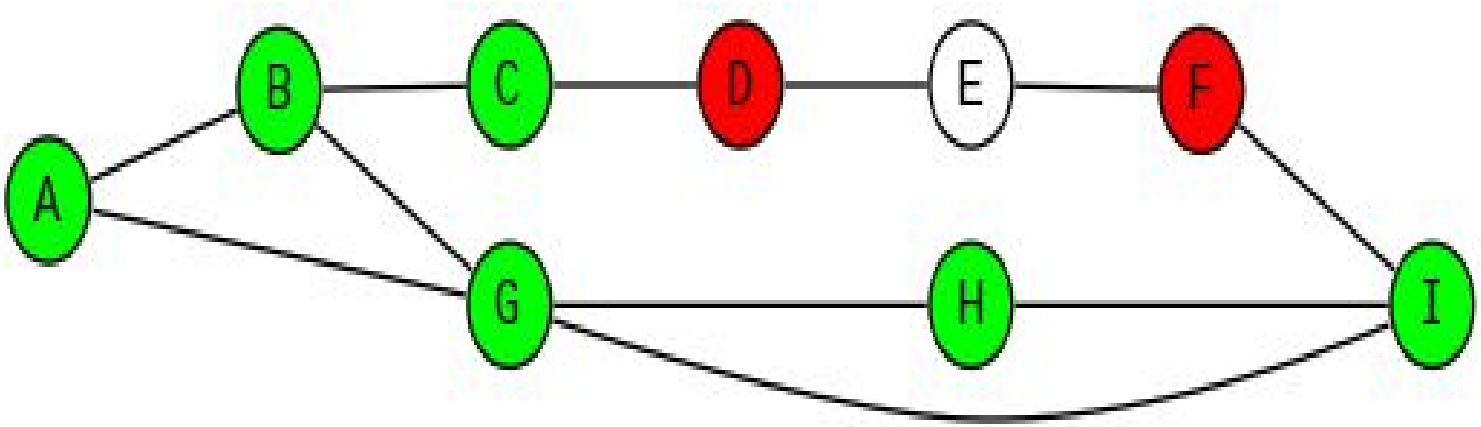
Dołączamy wierzchołki z S do zbioru L i szukamy ich sąsiadów, którzy nie są w zbiorze L. Utworzą oni zbiór S.

Algorytm przeszukiwania wszerz - krok3



Skoro to trzeci obieg pętli, wierzchołkom z S przypisujemy odległość 3. Ich wskaźnikami są ich sąsiedzi ze zbioru L.

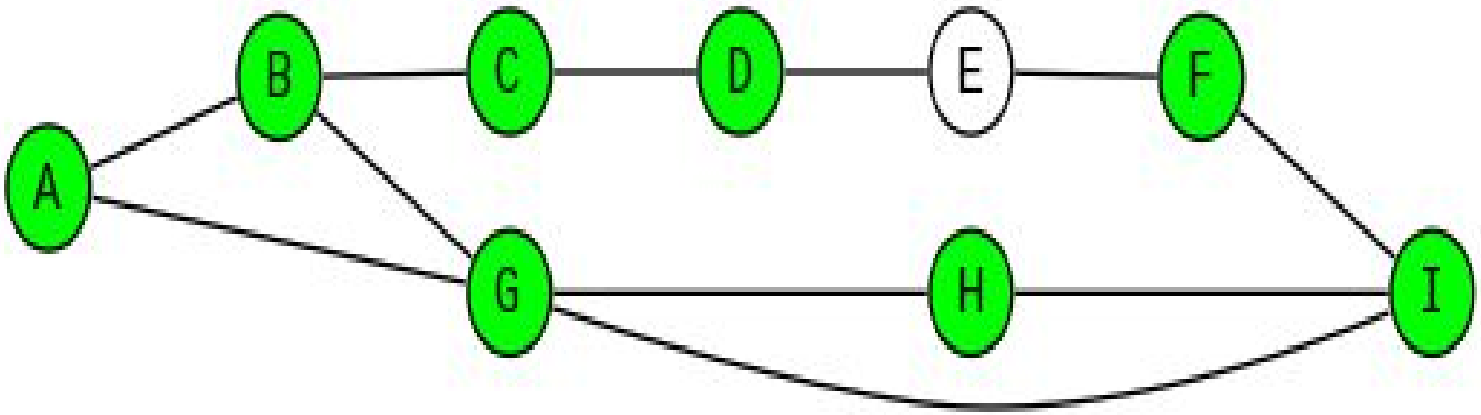
Algorytm przeszukiwania wszerz - krok3



Skoro to trzeci obieg pętli, wierzchołkom z S przypisujemy odległość 3. Ich wskaźnikami są ich sąsiedzi ze zbioru L.

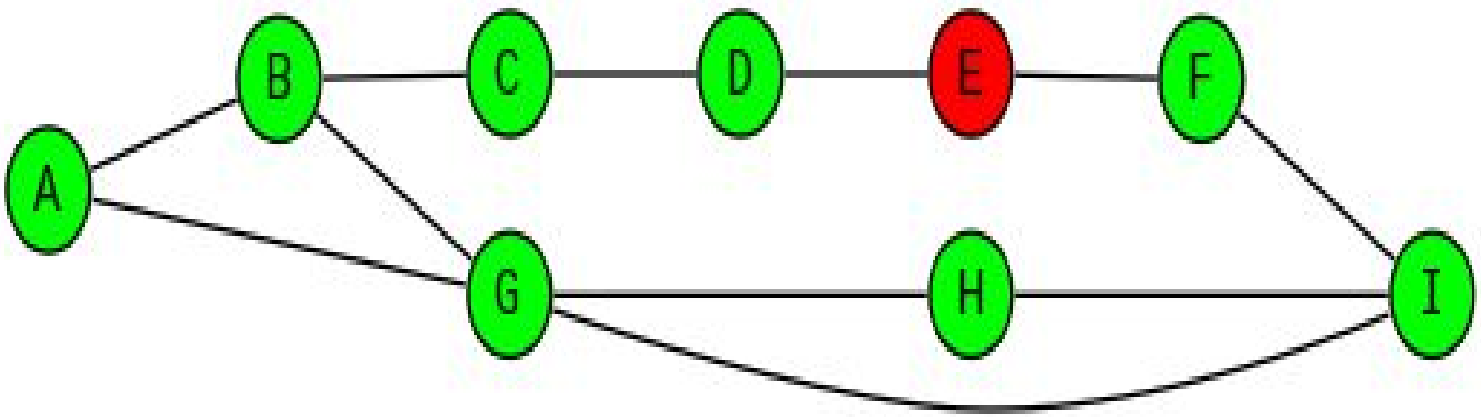
Nr	zbiór L	zbiór S	uzyskane odległości	wskaźniki
3	A,B,C,G,H,I	D, F	$d(D) = d(F) = 3$	$p(D) = C$, $p(F) = I$

Algorytm przeszukiwania wszerz - krok4



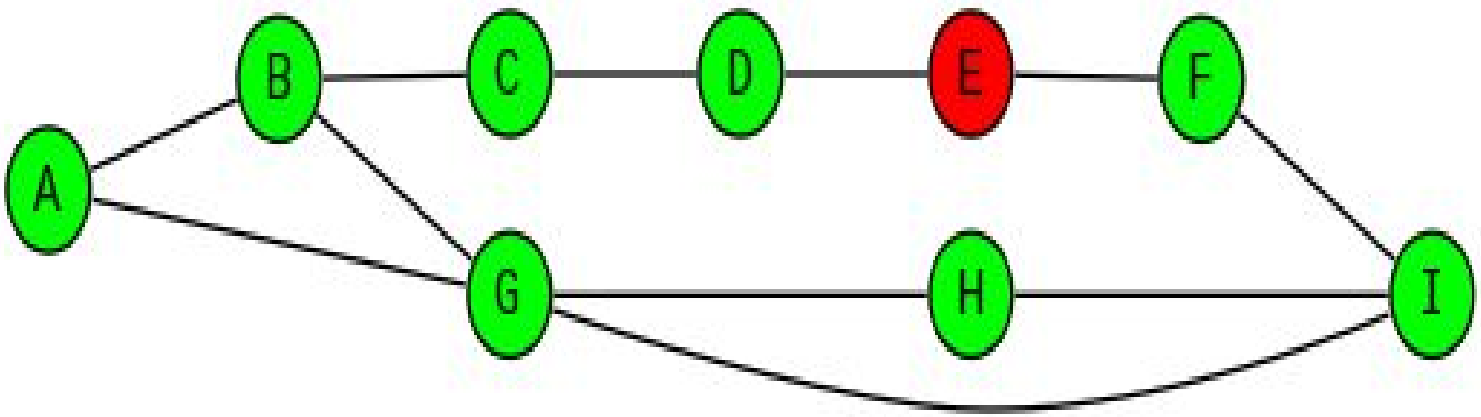
Dołączamy wierzchołki z S do zbioru L i szukamy ich sąsiadów, którzy nie są w zbiorze L. Został tylko jeden taki wierzchołek, który utworzy zbiór S.

Algorytm przeszukiwania wszerz - krok4



Skoro to czwarty obieg pętli, wierzchołkowi z S przypisujemy odległość 4. Ma więcej niż jednego sąsiada w L, więc wybór wskaźnika nie jest jednoznaczny. Zgodnie z treścią algorytmu wybieramy ten o najmniejszym numerze (czyli D przed F).

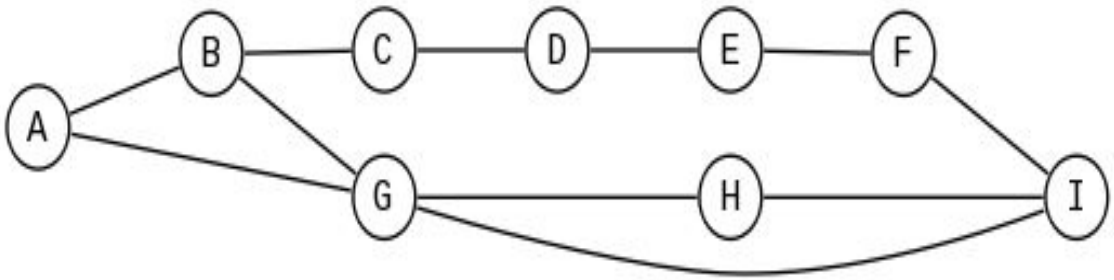
Algorytm przeszukiwania wszerz - krok4



Skoro to czwarty obieg pętli, wierzchołkowi z S przypisujemy odległość 4. Ma więcej niż jednego sąsiada w L, więc wybór wskaźnika nie jest jednoznaczny. Zgodnie z treścią algorytmu wybieramy ten o najmniejszym numerze (czyli D przed F).

Nr	zbiór L	zbiór S	uzyskane odległości	wskaźniki
4	A,B,C,D,F,G,H,I	E	$d(E) = 4$	$p(E) = D$

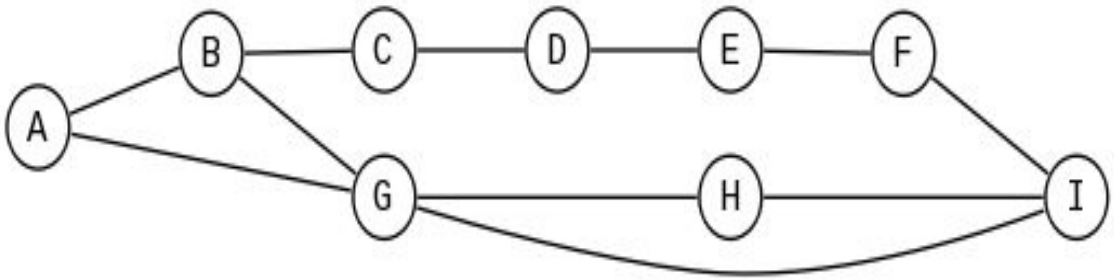
Algorytm przeszukiwania wszerz - tabela



Poniższa tabela jest wynikiem działania algorytmu:

Nr	zbiór L	zbiór S	obliczone odległości	wskaźniki
1	A	B,G	$d(B) = d(G) = 1$	$p(B) = p(G) = A$
2	A,B,G	C,H,I	$d(C) = d(H) = d(I) = 2$	$p(C) = B$, $p(H) = p(I) = G$
3	A,B,C,G,H,I	D, F	$d(D) = d(F) = 3$	$p(D) = C$, $p(F) = I$
4	A,B,C,D,F,G,H,I	E	$d(E) = 4$	$p(E) = D$

Algorytm przeszukiwania wszerz - tabela

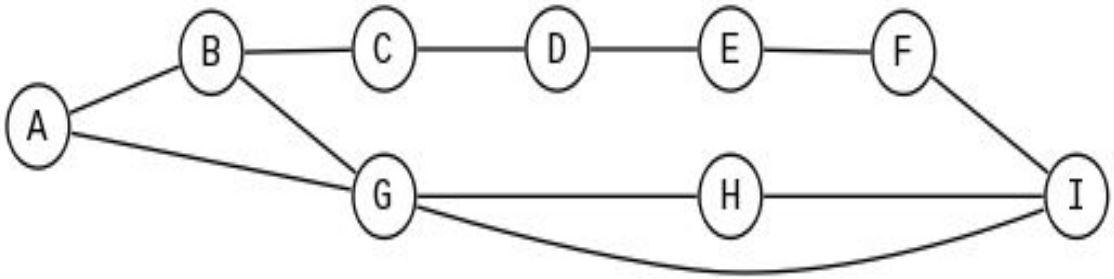


Poniższa tabela jest wynikiem działania algorytmu:

Nr	zbiór L	zbiór S	obliczone odległości	wskaźniki
1	A	B,G	$d(B) = d(G) = 1$	$p(B) = p(G) = A$
2	A,B,G	C,H,I	$d(C) = d(H) = d(I) = 2$	$p(C) = B$, $p(H) = p(I) = G$
3	A,B,C,G,H,I	D, F	$d(D) = d(F) = 3$	$p(D) = C$, $p(F) = I$
4	A,B,C,D,F,G,H,I	E	$d(E) = 4$	$p(E) = D$

Jeśli chcemy znaleźć najkrótszą drogę np. z A do F, musimy zacząć od F, sprawdzić $p(F) = I$,

Algorytm przeszukiwania wszerz - tabela

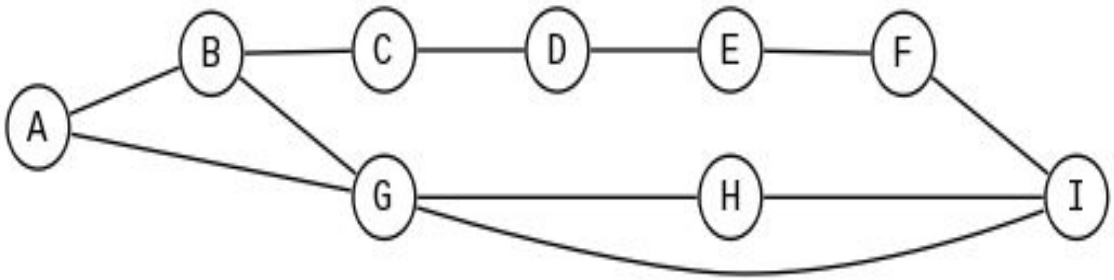


Poniższa tabela jest wynikiem działania algorytmu:

Nr	zbiór L	zbiór S	obliczone odległości	wskaźniki
1	A	B,G	$d(B) = d(G) = 1$	$p(B) = p(G) = A$
2	A,B,G	C,H,I	$d(C) = d(H) = d(I) = 2$	$p(C) = B$, $p(H) = p(I) = G$
3	A,B,C,G,H,I	D, F	$d(D) = d(F) = 3$	$p(D) = C$, $p(F) = I$
4	A,B,C,D,F,G,H,I	E	$d(E) = 4$	$p(E) = D$

Jeśli chcemy znaleźć najkrótszą drogę np. z A do F, musimy zacząć od F, sprawdzić $p(F) = I$,następnie $p(I) = G$

Algorytm przeszukiwania wszerz - tabela

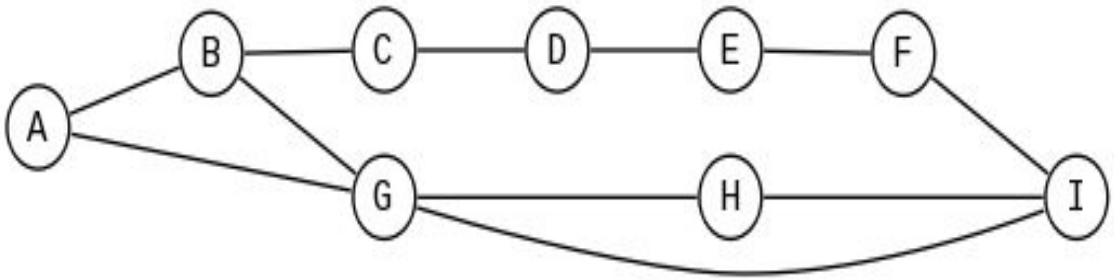


Poniższa tabela jest wynikiem działania algorytmu:

Nr	zbiór L	zbiór S	obliczone odległości	wskaźniki
1	A	B,G	$d(B) = d(G) = 1$	$p(B) = p(G) = A$
2	A,B,G	C,H,I	$d(C) = d(H) = d(I) = 2$	$p(C) = B$, $p(H) = p(I) = G$
3	A,B,C,G,H,I	D, F	$d(D) = d(F) = 3$	$p(D) = C$, $p(F) = I$
4	A,B,C,D,F,G,H,I	E	$d(E) = 4$	$p(E) = D$

Jeśli chcemy znaleźć najkrótszą drogę np. z A do F, musimy zacząć od F, sprawdzić $p(F) = I$,następnie $p(I) = G$ i $p(G) = A$.

Algorytm przeszukiwania wszerz - tabela



Poniższa tabela jest wynikiem działania algorytmu:

Nr	zbiór L	zbiór S	obliczone odległości	wskaźniki
1	A	B,G	$d(B) = d(G) = 1$	$p(B) = p(G) = A$
2	A,B,G	C,H,I	$d(C) = d(H) = d(I) = 2$	$p(C) = B$, $p(H) = p(I) = G$
3	A,B,C,G,H,I	D, F	$d(D) = d(F) = 3$	$p(D) = C$, $p(F) = I$
4	A,B,C,D,F,G,H,I	E	$d(E) = 4$	$p(E) = D$

Jeśli chcemy znaleźć najkrótszą drogę np. z A do F, musimy zacząć od F, sprawdzić $p(F) = I$,następnie $p(I) = G$ i $p(G) = A$.Skoro dotarliśmy do A, droga się kończy i wystarczy wypisać jej wierzchołki w odwrotnej kolejności: $AGIF$.

Dodatkowe informacje o algorytmie przeszukiwania wszerz

Dodatkowe informacje o algorytmie przeszukiwania wszerz

- Dla grafu $G = (V, E)$ czas działania algorytmu to $O(|V| + |E|)$.

Dodatkowe informacje o algorytmie przeszukiwania wszerz

- Dla grafu $G = (V, E)$ czas działania algorytmu to $O(|V| + |E|)$.
- Obecność pętli i krawędzi wielokrotnych nic by nie zmieniała w problemie.

Algorytm Dijkstry - wstęp

Algorytm przeszukiwania wszerz ma ograniczony zakres stosowalności. Nieczęsto się zdarza, by w rzeczywistym problemie wszystkie krawędzie grafu (drogi między skrzyżowaniami, procesy pomiędzy kolejnymi etapami produkcji) miały tę samą wagę (tj. odległość, koszt, czas konieczny do przebycia itp.).

Algorytm Dijkstry - wstęp

Algorytm przeszukiwania wszerz ma ograniczony zakres stosowalności. Nieczęsto się zdarza, by w rzeczywistym problemie wszystkie krawędzie grafu (drogi między skrzyżowaniami, procesy pomiędzy kolejnymi etapami produkcji) miały tę samą wagę (tj. odległość, koszt, czas konieczny do przebycia itp.).
Teraz przejdziemy do trudniejszego przypadku. Dany jest graf prosty (znów obecność pętli i krawędzi wielokrotnych nic nie zmienia, ale zapis algorytmu jest prostszy bez nich), skierowany $G = (V, E)$. Każdej jego krawędzi e przypisujemy wagę $W(e)$.

Algorytm Dijkstry - wstęp

Algorytm przeszukiwania wszerz ma ograniczony zakres stosowalności. Nieczęsto się zdarza, by w rzeczywistym problemie wszystkie krawędzie grafu (drogi między skrzyżowaniami, procesy pomiędzy kolejnymi etapami produkcji) miały tę samą wagę (tj. odległość, koszt, czas konieczny do przebycia itp.).
Teraz przejdziemy do trudniejszego przypadku. Dany jest graf prosty (znów obecność pętli i krawędzi wielokrotnych nic nie zmienia, ale zapis algorytmu jest prostszy bez nich), skierowany $G = (V, E)$. Każdej jego krawędzi e przypisujemy wagę $W(e)$.
Jak poprzednio, zaczynamy od ponumerowania w dowolny sposób wierzchołków takiego grafu $V = \{1, \ldots, n\}$. Poszukujemy drogi z wierzchołka 1 do każdego innego wierzchołka j o najmniejszej wadze (czyli sumie wag krawędzi). Rozwiązaniem tego problemu jest algorytm Dijkstry (ze wskaźnikami).

Algorytm Dijkstry ze wskaźnikami - część I

Algorytm Dijkstry

Dane: Graf $G = (V(G), E(G))$ prosty skierowany ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$ i funkcją W wag („długości") krawędzi o wartościach nieujemnych (jeśli krawędź (v, w) nie istnieje to podstawiamy $W(v, w) = \infty$).
Zmienne: L i V - zbiory wierzchołków , d, p - funkcje.

Algorytm Dijkstry ze wskaźnikami - część I

Algorytm Dijkstry

Dane: Graf $G = (V(G), E(G))$ prosty skierowany ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$ i funkcją W wag („długości") krawędzi o wartościach nieujemnych (jeśli krawędź (v, w) nie istnieje to podstawiamy $W(v, w) = \infty$).
Zmienne: L i V - zbiory wierzchołków , d, p - funkcje.

- I. $L := \{1\}$, $V := \{2, \ldots n\}$.

Algorytm Dijkstry ze wskaźnikami - część I

Algorytm Dijkstry

Dane: Graf $G = (V(G), E(G))$ prosty skierowany ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$ i funkcją W wag („długości") krawędzi o wartościach nieujemnych (jeśli krawędź (v, w) nie istnieje to podstawiamy $W(v, w) = \infty$).
Zmienne: L i V - zbiory wierzchołków , d, p - funkcje.

- I. $L := \{1\}$, $V := \{2, \ldots n\}$.
- II. Dla $i \in V$ wykonuj:

Algorytm Dijkstry ze wskaźnikami - część I

Algorytm Dijkstry

Dane: Graf $G = (V(G), E(G))$ prosty skierowany ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$ i funkcją W wag („długości") krawędzi o wartościach nieujemnych (jeśli krawędź (v, w) nie istnieje to podstawiamy $W(v, w) = \infty$).
Zmienne: L i V - zbiory wierzchołków , d, p - funkcje.

- I. $L := \{1\}$, $V := \{2, \ldots n\}$.
- II. Dla $i \in V$ wykonuj:
- II.1. $d(i) := W(1, i)$.

Algorytm Dijkstry ze wskaźnikami - część I

Algorytm Dijkstry

Dane: Graf $G = (V(G), E(G))$ prosty skierowany ze zbiorem wierzchołków $V(G) = \{1, \ldots, n\}$ i funkcją W wag („długości") krawędzi o wartościach nieujemnych (jeśli krawędź (v, w) nie istnieje to podstawiamy $W(v, w) = \infty$).
Zmienne: L i V - zbiory wierzchołków , d, p - funkcje.

- I. $L := \{1\}$, $V := \{2, \ldots n\}$.
- II. Dla $i \in V$ wykonuj:
- II.1. $d(i) := W(1, i)$.
- II.2. Jeśli $W(1, i) = \infty$ to $p(i) := 0$, w innym wypadku $p(i) := 1$.

Algorytm Dijkstry ze wskaźnikami - część II

Algorytm Dijkstry

Algorytm Dijkstry ze wskaźnikami - część II

Algorytm Dijkstry

- III. Dopóki $V \setminus L \neq \emptyset$, wykonuj:

Algorytm Dijkstry ze wskaźnikami - część II

Algorytm Dijkstry

- III. Dopóki $V \setminus L \neq \emptyset$, wykonuj:
- III.1. wybierz $k \in V \setminus L$ takie, że $d(k)$ przyjmuje najmniejszą możliwą wartość.

Algorytm Dijkstry ze wskaźnikami - część II

Algorytm Dijkstry

- III. Dopóki $V \setminus L \neq \emptyset$, wykonuj:
- III.1. wybierz $k \in V \setminus L$ takie, że $d(k)$ przyjmuje najmniejszą możliwą wartość.
- III.2. dołącz k do zbioru L.

Algorytm Dijkstry ze wskaźnikami - część II

Algorytm Dijkstry

- III. Dopóki $V \setminus L \neq \emptyset$, wykonuj:
- III.1. wybierz $k \in V \setminus L$ takie, że $d(k)$ przyjmuje najmniejszą możliwą wartość.
- III.2. dołącz k do zbioru L.
- III.3. dla każdego $j \in V \setminus L$ wykonuj: jeśli $d(j) > d(k) + d(k, j)$ to zastąp $d(j)$ sumą $d(k) + d(k, j)$ i zastąp $p(j)$ liczbą k.

Algorytm Dijkstry ze wskaźnikami - część II

Algorytm Dijkstry

- III. Dopóki $V \setminus L \neq \emptyset$, wykonuj:
- III.1. wybierz $k \in V \setminus L$ takie, że $d(k)$ przyjmuje najmniejszą możliwą wartość.
- III.2. dołącz k do zbioru L.
- III.3. dla każdego $j \in V \setminus L$ wykonuj: jeśli $d(j) > d(k) + d(k, j)$ to zastąp $d(j)$ sumą $d(k) + d(k, j)$ i zastąp $p(j)$ liczbą k.
- **Rezultat**: $d(j)$ to minimalna waga drogi z 1 do j, $p(j)$ to wskaźnik, który pokazuje poprzedni etap drogi z 1 do j.

Algorytm Dijkstry - wstępne uwagi

Przebieg drogi o najmniejszej wadze odczytujemy ze wskaźników, jak w wypadku przeszukiwania wszerz.

Algorytm Dijkstry - wstępne uwagi

Przebieg drogi o najmniejszej wadze odczytujemy ze wskaźników, jak w wypadku przeszukiwania wszerz.
Tłumaczenie algorytmu na ludzki: W każdym kroku zawsze patrzymy na ostatni wierzchołek (nazwijmy go k), który dołączyliśmy do zbioru L, czyli do zbioru wierzchołków, dla których już wszystko obliczyliśmy.

Algorytm Dijkstry - wstępne uwagi

Przebieg drogi o najmniejszej wadze odczytujemy ze wskaźników, jak w wypadku przeszukiwania wszerz.
Tłumaczenie algorytmu na ludzki: W każdym kroku zawsze patrzymy na ostatni wierzchołek (nazwijmy go k), który dołączyliśmy do zbioru L, czyli do zbioru wierzchołków, dla których już wszystko obliczyliśmy. Następnie sprawdzamy, do jakich wierzchołków, które nie są w zbiorze L prowadzą krawędzie z naszego wierzchołka

Algorytm Dijkstry - wstępne uwagi

Przebieg drogi o najmniejszej wadze odczytujemy ze wskaźników, jak w wypadku przeszukiwania wszerz.
Tłumaczenie algorytmu na ludzki: W każdym kroku zawsze patrzymy na ostatni wierzchołek (nazwijmy go k), który dołączyliśmy do zbioru L, czyli do zbioru wierzchołków, dla których już wszystko obliczyliśmy. Następnie sprawdzamy, do jakich wierzchołków, które nie są w zbiorze L prowadzą krawędzie z naszego wierzchołka i dla każdego z tych wierzchołków sprawdzamy, czy droga, która prowadzi do niego przez k nie jest krótsza od dotychczas znalezionych dróg.

Algorytm Dijkstry - wstępne uwagi

Przebieg drogi o najmniejszej wadze odczytujemy ze wskaźników, jak w wypadku przeszukiwania wszerz.
Tłumaczenie algorytmu na ludzki: W każdym kroku zawsze patrzymy na ostatni wierzchołek (nazwijmy go k), który dołączyliśmy do zbioru L, czyli do zbioru wierzchołków, dla których już wszystko obliczyliśmy. Następnie sprawdzamy, do jakich wierzchołków, które nie są w zbiorze L prowadzą krawędzie z naszego wierzchołka i dla każdego z tych wierzchołków sprawdzamy, czy droga, która prowadzi do niego przez k nie jest krótsza od dotychczas znalezionych dróg. Jeśli tak, zapisujemy jej nową wagę.

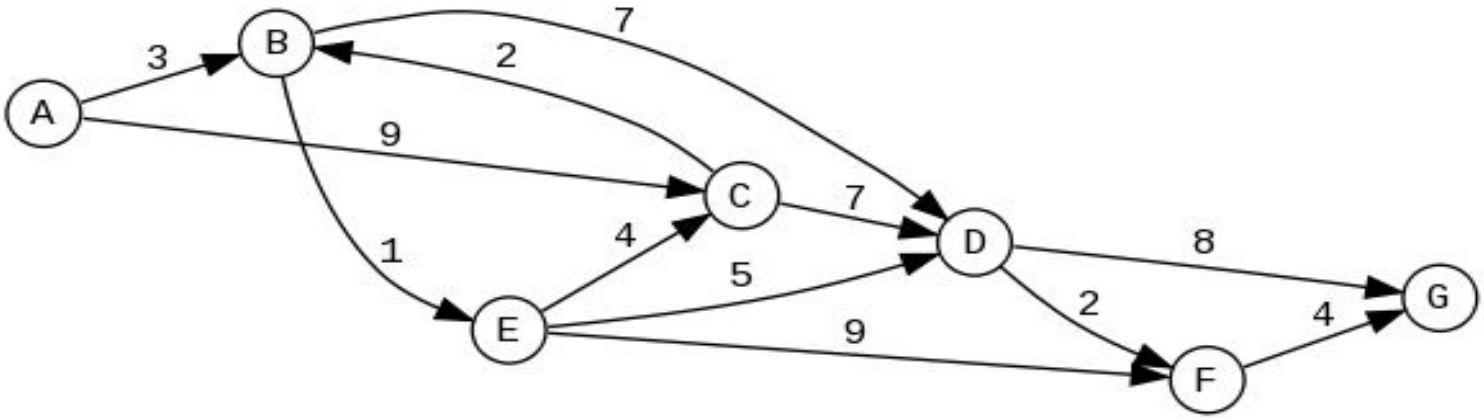
Algorytm Dijkstry - wstępne uwagi

Przebieg drogi o najmniejszej wadze odczytujemy ze wskaźników, jak w wypadku przeszukiwania wszerz.
Tłumaczenie algorytmu na ludzki: W każdym kroku zawsze patrzymy na ostatni wierzchołek (nazwijmy go k), który dołączyliśmy do zbioru L, czyli do zbioru wierzchołków, dla których już wszystko obliczyliśmy. Następnie sprawdzamy, do jakich wierzchołków, które nie są w zbiorze L prowadzą krawędzie z naszego wierzchołka i dla każdego z tych wierzchołków sprawdzamy, czy droga, która prowadzi do niego przez k nie jest krótsza od dotychczas znalezionych dróg. Jeśli tak, zapisujemy jej nową wagę.Gdy się skończą wierzchołki do zbadania, wybieramy ten, który ma w tym momencie najmniejszą odległość od startu i jest poza L i on się staje nowym wierzchołkiem k.

Algorytm Dijkstry - wstępne uwagi

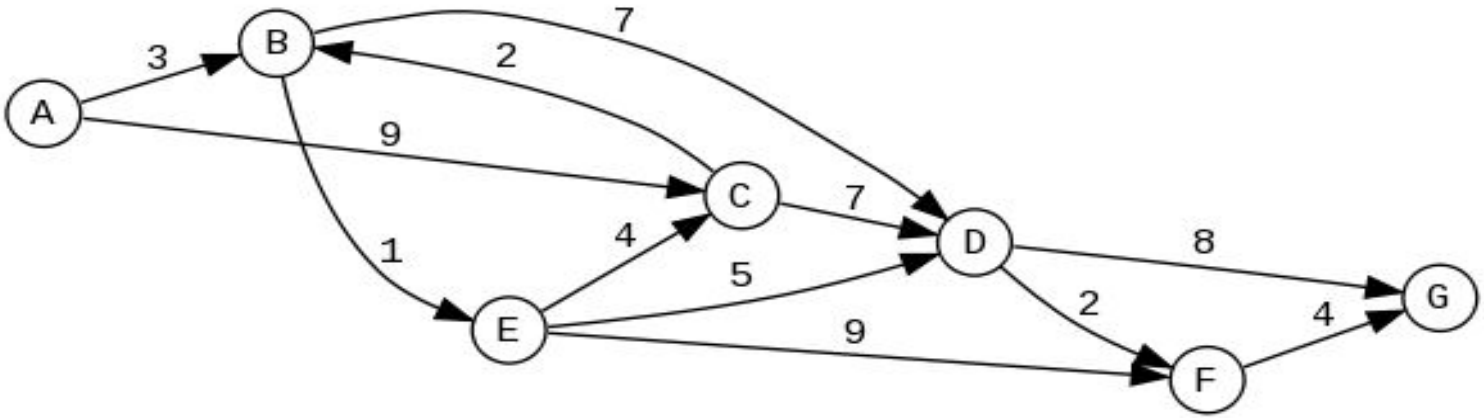
Przebieg drogi o najmniejszej wadze odczytujemy ze wskaźników, jak w wypadku przeszukiwania wszerz.
Tłumaczenie algorytmu na ludzki: W każdym kroku zawsze patrzymy na ostatni wierzchołek (nazwijmy go k), który dołączyliśmy do zbioru L, czyli do zbioru wierzchołków, dla których już wszystko obliczyliśmy. Następnie sprawdzamy, do jakich wierzchołków, które nie są w zbiorze L prowadzą krawędzie z naszego wierzchołka i dla każdego z tych wierzchołków sprawdzamy, czy droga, która prowadzi do niego przez k nie jest krótsza od dotychczas znalezionych dróg. Jeśli tak, zapisujemy jej nową wagę.Gdy się skończą wierzchołki do zbadania, wybieramy ten, który ma w tym momencie najmniejszą odległość od startu i jest poza L i on się staje nowym wierzchołkiem k. Powtarzamy, aż się skończą wierzchołki.

Algorytm Dijkstry - przykład



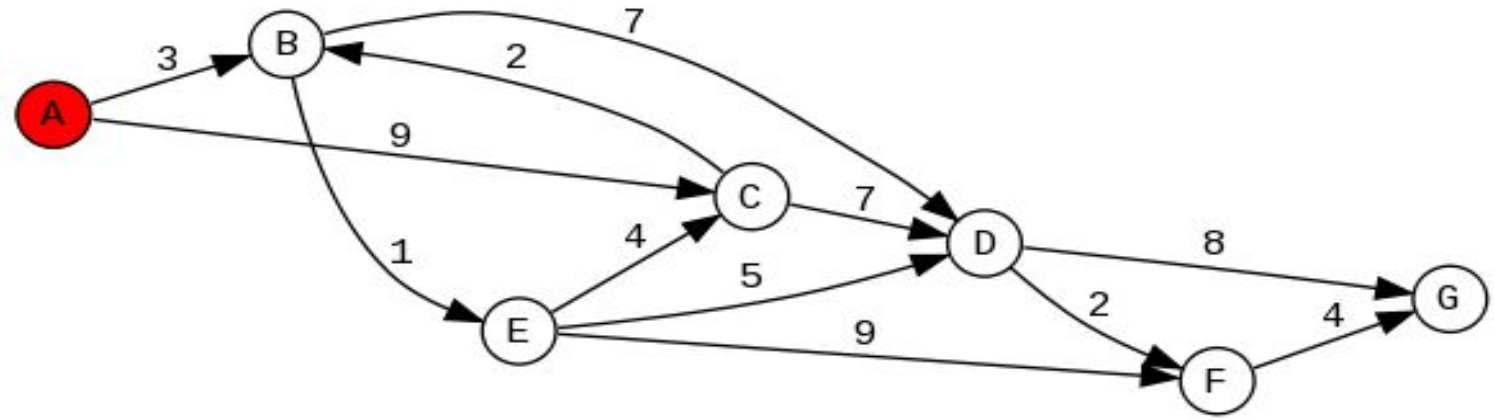
Spróbujemy zastosować algorytm Dijkstry dla powyższego grafu skierowanego. Przedstawię tutaj sposób postępowania przyjęty w ramach tego kursu - obowiązujący na sprawdzianie/egzaminie.

Algorytm Dijkstry - przykład



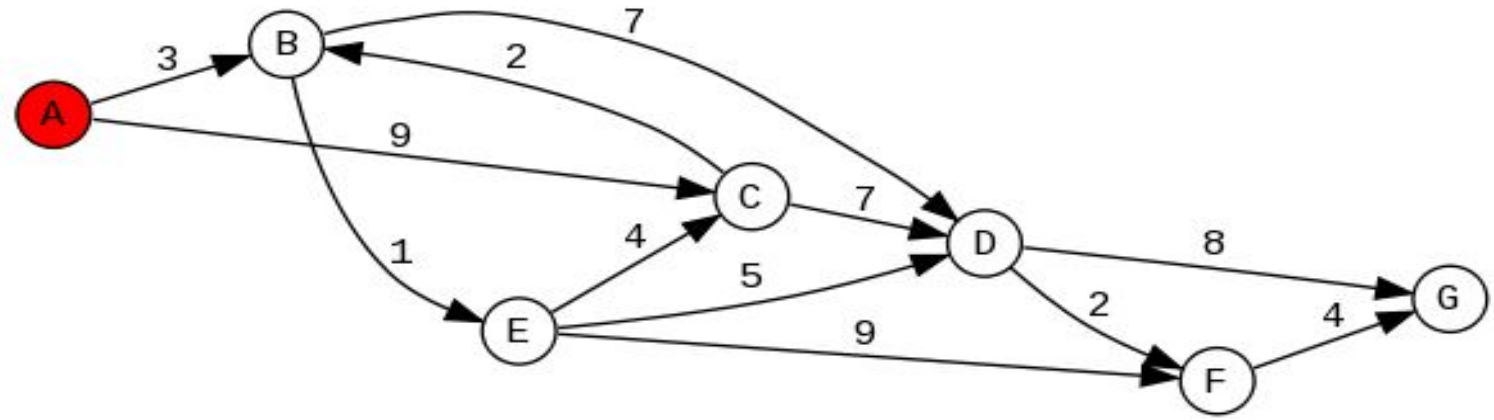
Spróbujemy zastosować algorytm Dijkstry dla powyższego grafu skierowanego. Przedstawię tutaj sposób postępowania przyjęty w ramach tego kursu - obowiązujący na sprawdzianie/egzaminie. Algorytm zaczynamy od ponumerowania wierzchołków w kolejności alfabetycznej: A-1, B-2 itd... Będziemy obliczać minimalną długość drogi pomiędzy A i każdym z pozostałych wierzchołków.

Algorytm Dijkstry - przykład



Zaczynamy od wierzchołka A. Na zielono będę zaznaczać wierzchołki będące w L od poprzedniego obiegu pętli, na czerwono wierzchołek ostatnio dołączony do L (w algorytmie - k), a na niebiesko wierzchołki, których dane właśnie korygujemy.

Algorytm Dijkstry - przykład

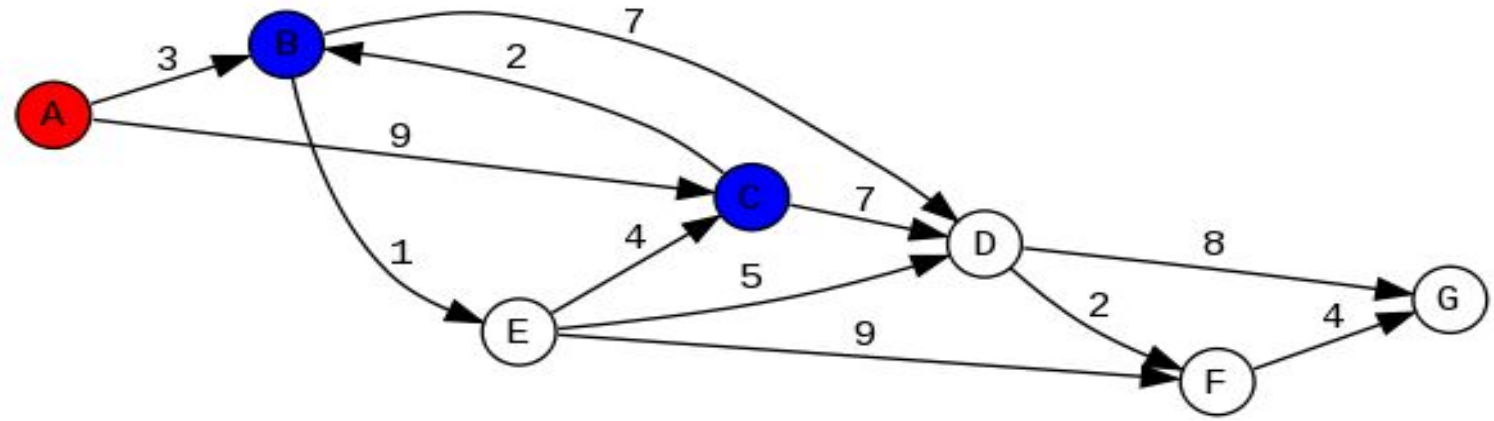


Zaczynamy od wierzchołka A. Na zielono będę zaznaczać wierzchołki będące w L od poprzedniego obiegu pętli, na czerwono wierzchołek ostatnio dołączony do L (w algorytmie - k), a na niebiesko wierzchołki, których dane właśnie korygujemy.

Działanie algorytmu zapisujemy w takiej tabeli:

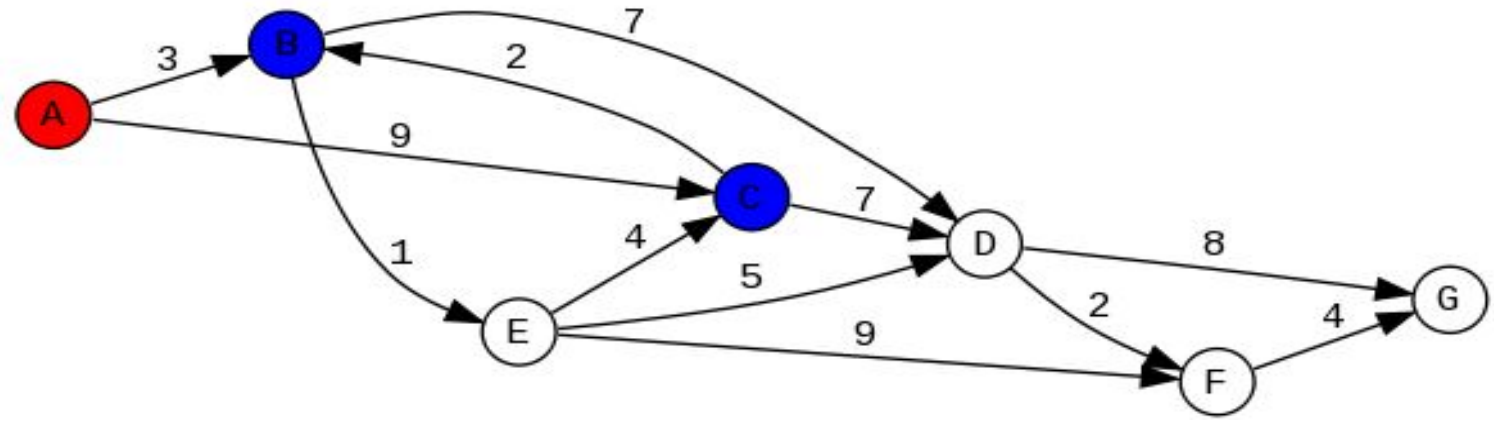
Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G)
1		
2		

Algorytm Dijkstry - przykład - krok 1



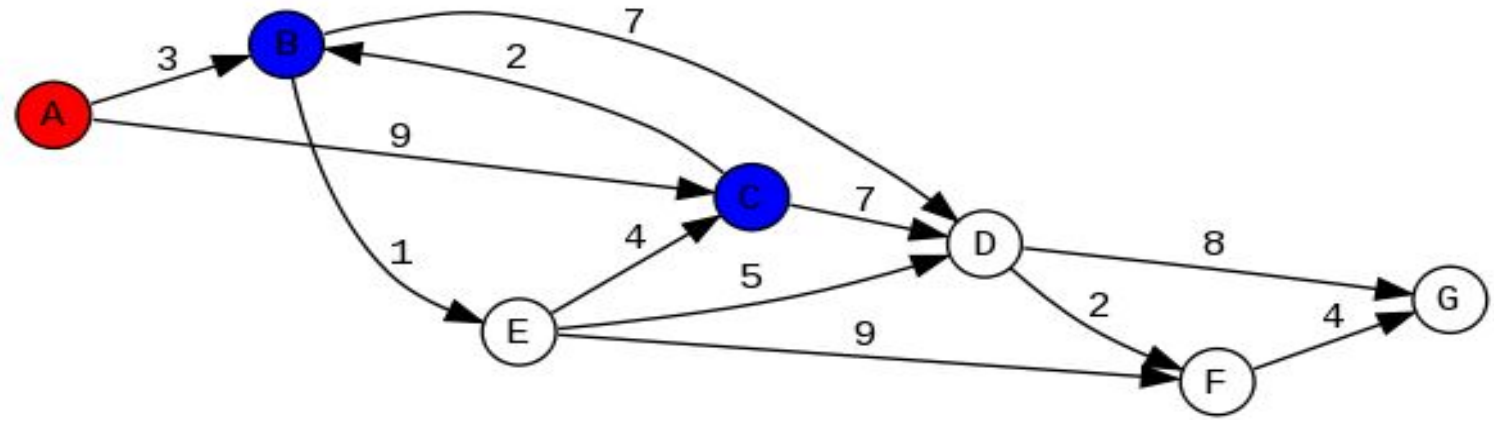
Na początku wszystkie wagi (odległości) są ustalone na $+\infty$, a wskaźniki mają wartość 0, póki te odległości się nie zmniejszą.

Algorytm Dijkstry - przykład - krok 1



Na początku wszystkie wagi (odległości) są ustalone na $+\infty$, a wskaźniki mają wartość 0, póki te odległości się nie zmniejszą. Badamy dwa wierzchołki, do których prowadzą krawędzie z A.

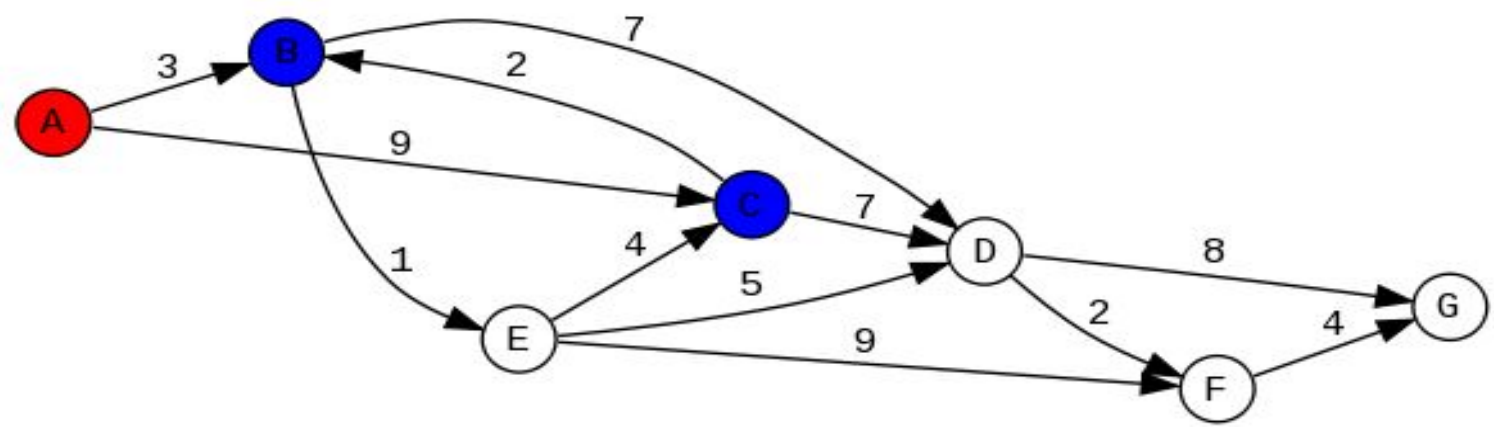
Algorytm Dijkstry - przykład - krok 1



Na początku wszystkie wagi (odległości) są ustalone na $+\infty$, a wskaźniki mają wartość 0, póki te odległości się nie zmniejszą. Badamy dwa wierzchołki, do których prowadzą krawędzie z A. Jako, że nowoznalezione drogi (AB i AC) są krótsze od dotychczasowych ($3 < \infty$ i $9 < \infty$), możemy zapisać ich długości w tabeli:

Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G)
1	A	$3A, 9A, +\infty, +\infty, +\infty, +\infty$

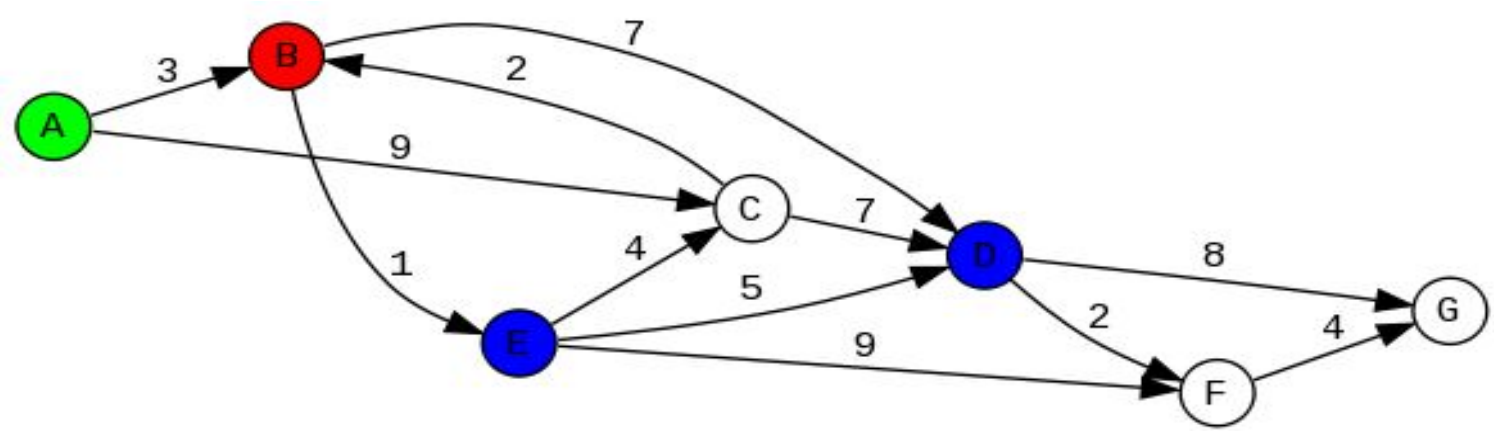
Algorytm Dijkstry - przykład - krok 1



Warto zauważyć, że poza nowymi odległościami, w tabeli notujemy jako wskaźnik - wierzchołek, z którego nowa, krótsza droga dotarła do danego. Teraz, do zbioru L dołączamy wierzchołek, który ma najmniejszą wartość funkcji d z jeszcze niedołączonych (czyli w tym wypadku B).

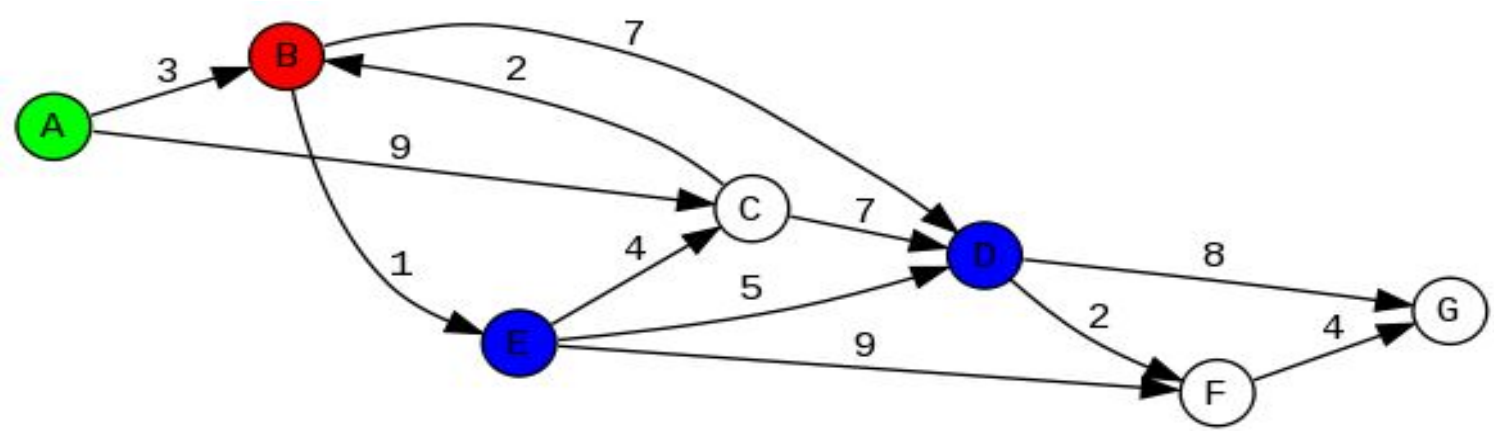
Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G)
1	A	$3A, 9A, +\infty, +\infty, +\infty, +\infty$
2	AB	

Algorytm Dijkstry - przykład - krok 2



Badamy teraz wierzchołki do których prowadzą krawędzie z B.

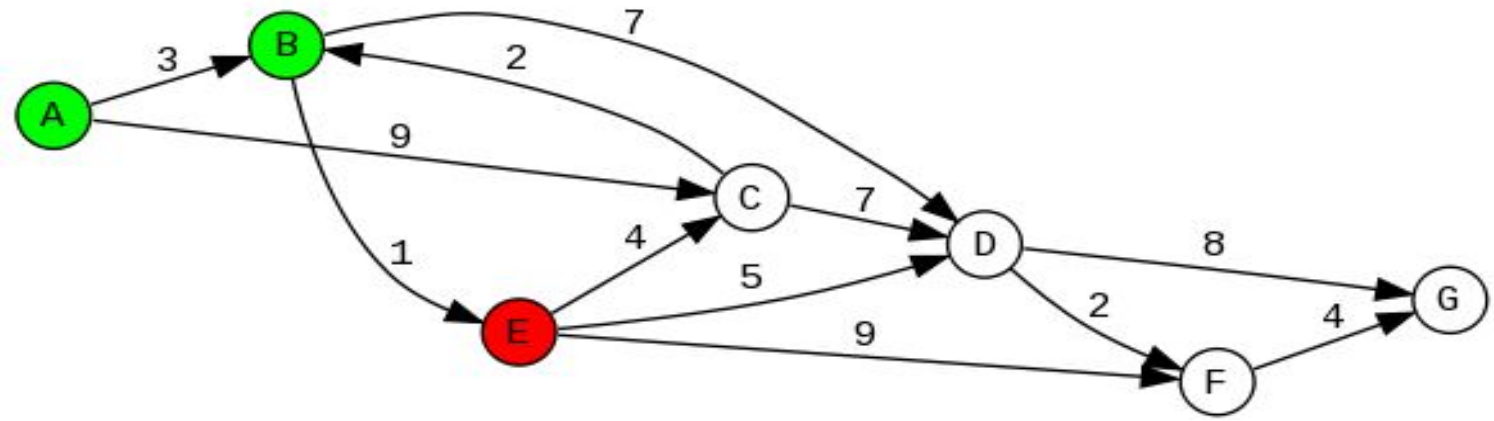
Algorytm Dijkstry - przykład - krok 2



Badamy teraz wierzchołki do których prowadzą krawędzie z B. Znajdujemy nowe wartości $d(D)$ i $d(E)$ dodając wagi krawędzi BD i BE do $d(B)$ i otrzymujemy propozycje $d(D) = 7 + 3 = 10$, $d(E) = 1 + 3 = 4$. Te wartości są mniejsze niż dotychczasowe $(+\infty)$, więc je zmieniamy, dopisując wskaźnik B.

Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G)
1	A	$3A, 9A, +\infty, +\infty, +\infty, +\infty$
2	A,B	$3A, 9A, 10B, 4B, +\infty, +\infty$

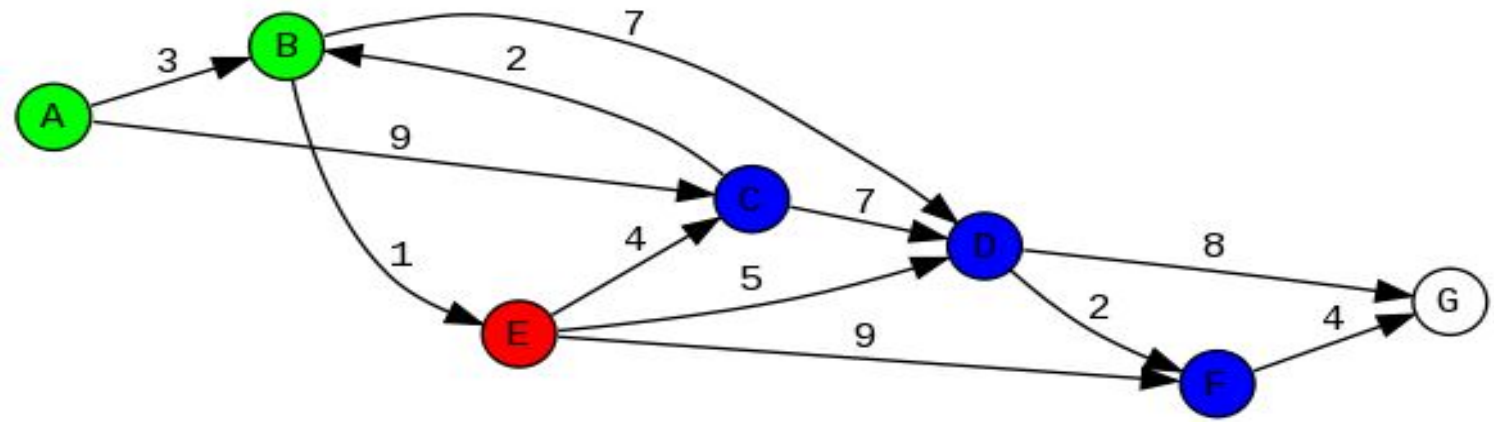
Algorytm Dijkstry - przykład - krok 2



Teraz, do zbioru L dołączamy wierzchołek, który ma najmniejszą wartość funkcji d z jeszcze niedołączonych (czyli w tym wypadku E). Zauważmy, że wśród potencjalnych kandydatów był wierzchołek C, choć nie zmienialiśmy jego odległości w tym kroku.

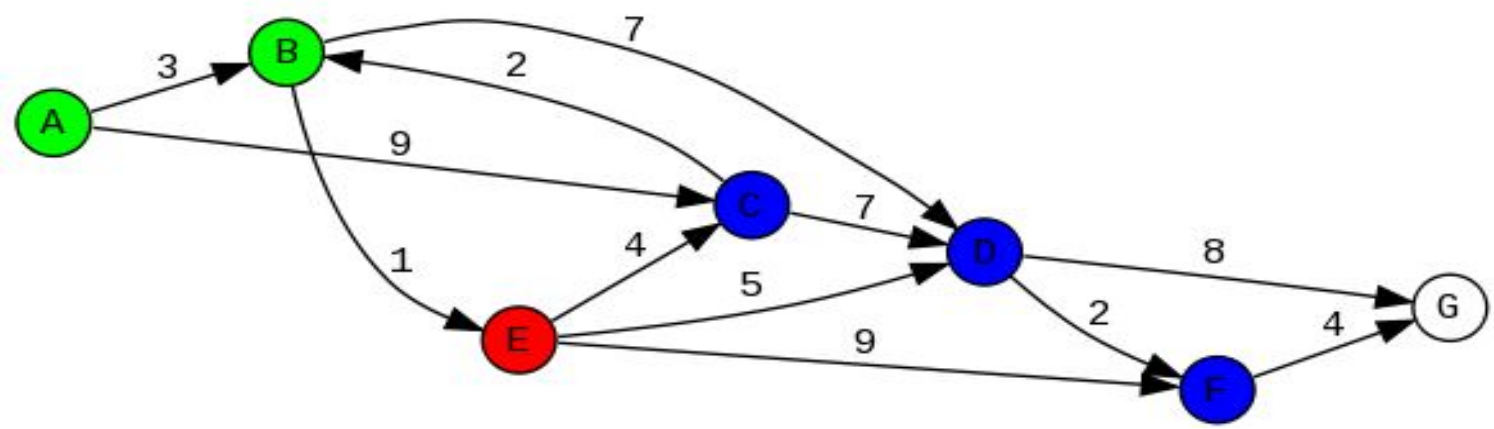
Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G)
1	A	$3A, 9A, +\infty, +\infty, +\infty, +\infty$
2	A,B	$3A, 9A, 10B, 4B, +\infty, +\infty$

Algorytm Dijkstry - przykład - krok 3



Badamy teraz wierzchołki do których prowadzą krawędzie z E.

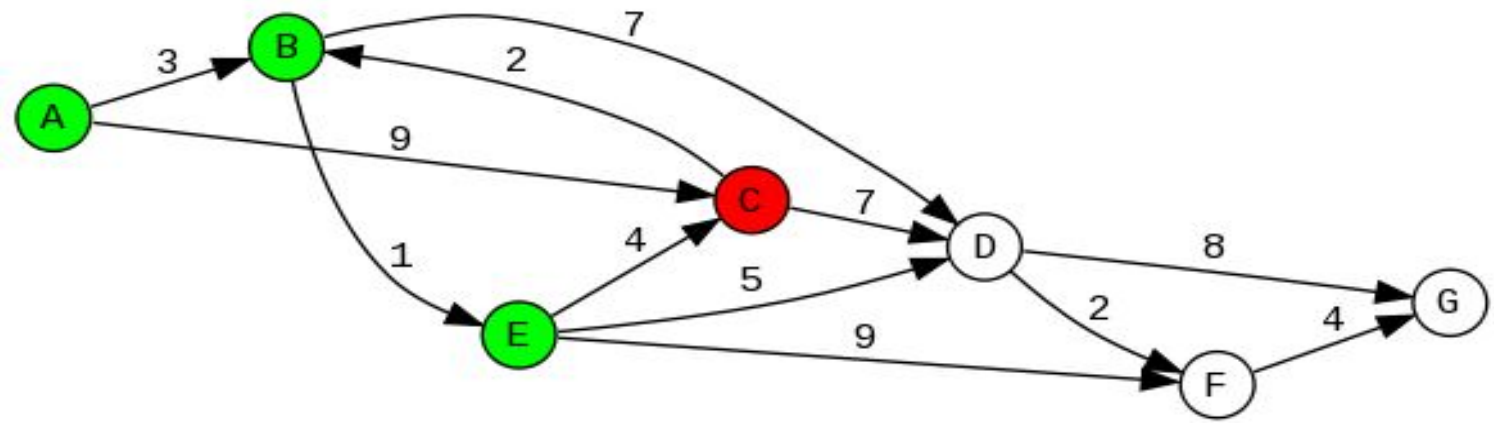
Algorytm Dijkstry - przykład - krok 3



Badamy teraz wierzchołki do których prowadzą krawędzie z E. Znajdujemy nowe wartości $d(C)$ i $d(D)$ i $d(F)$ dodając wagi krawędzi EC, ED i EF do $d(E)$ i otrzymujemy propozycje $d(C) = 4 + 4 = 8$, $d(D) = 5 + 4 = 9$, $d(D) = 9 + 4 = 13$. Te wartości są mniejsze niż dotychczasowe (9,10 i $+\infty$), więc je zmieniamy, dopisując wskaźnik E.

Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G)
2	A,B	$3A, 9A, 10B, 4B, +\infty, +\infty$
3	A,B,E	$3A, 8E, 9E, 4B, 13E, +\infty$

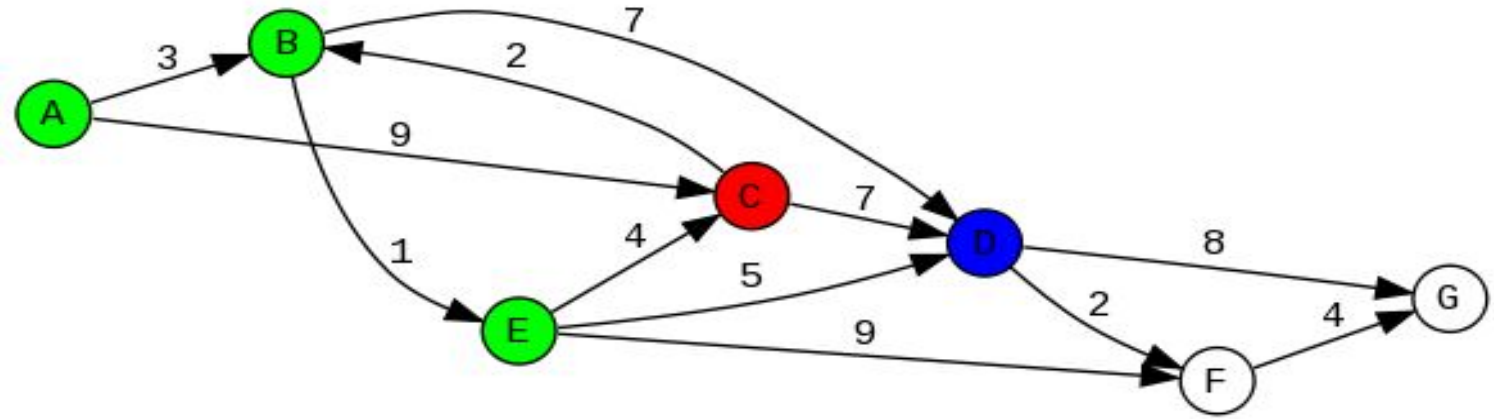
Algorytm Dijkstry - przykład - krok 3



Teraz, do zbioru L dołączamy wierzchołek, który ma najmniejszą wartość funkcji d z jeszcze niedołączonych (czyli w tym wypadku C).

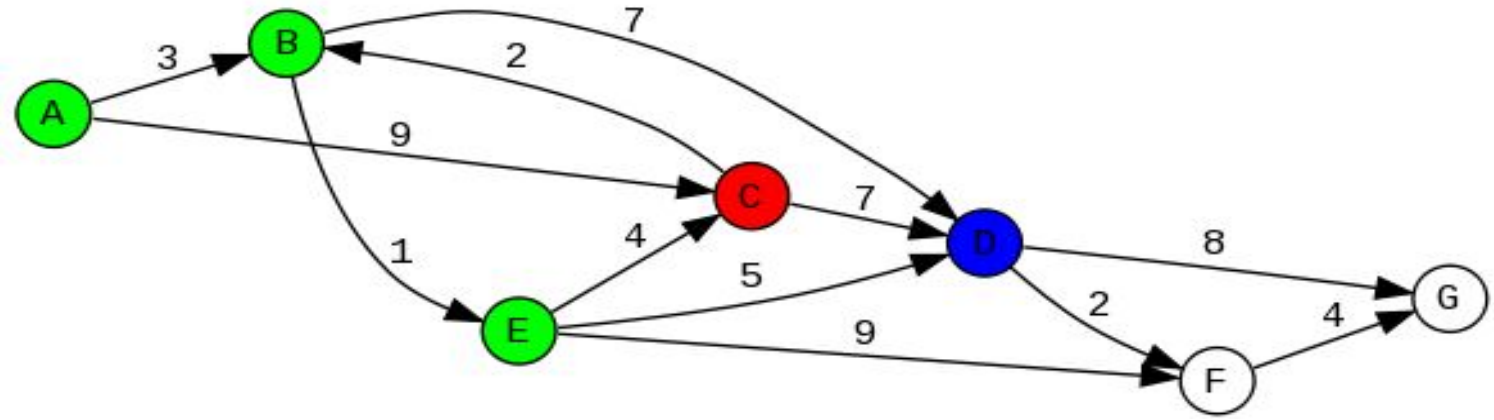
Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G)
2	A,B	$3A, 9A, 10B, 4B, +\infty, +\infty$
3	A,B,E	$3A, 8E, 9E, 4B, 13E, +\infty$

Algorytm Dijkstry - przykład - krok 4



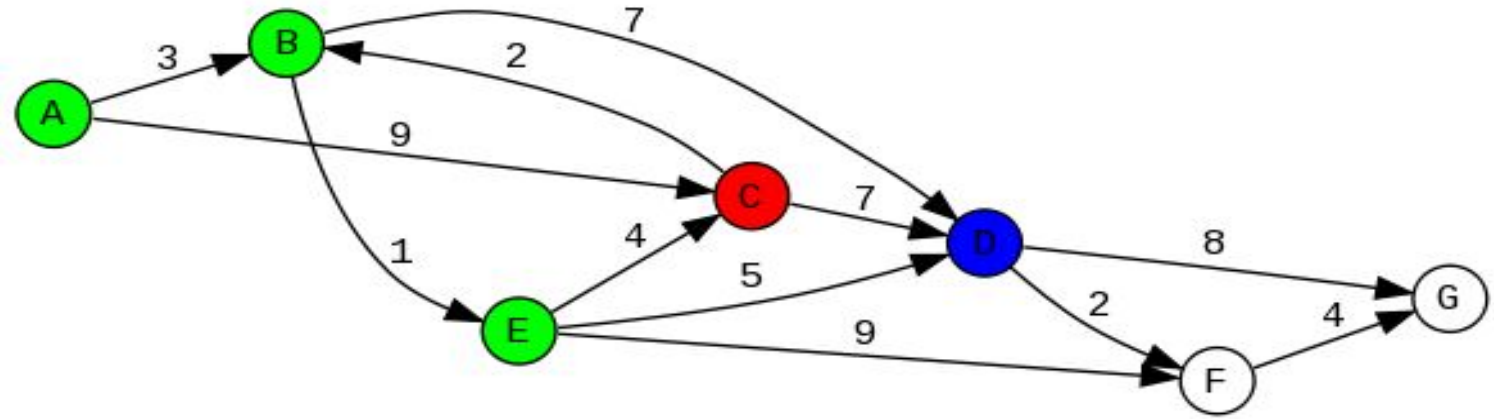
Badamy teraz wierzchołki do których prowadzą krawędzie z C.

Algorytm Dijkstry - przykład - krok 4



Badamy teraz wierzchołki do których prowadzą krawędzie z C. Znajdujemy nową wartość $d(D)$ dodając wagę krawędzi CD do $d(C)$ i otrzymujemy propozycję $d(D) = 7 + 8 = 15$.

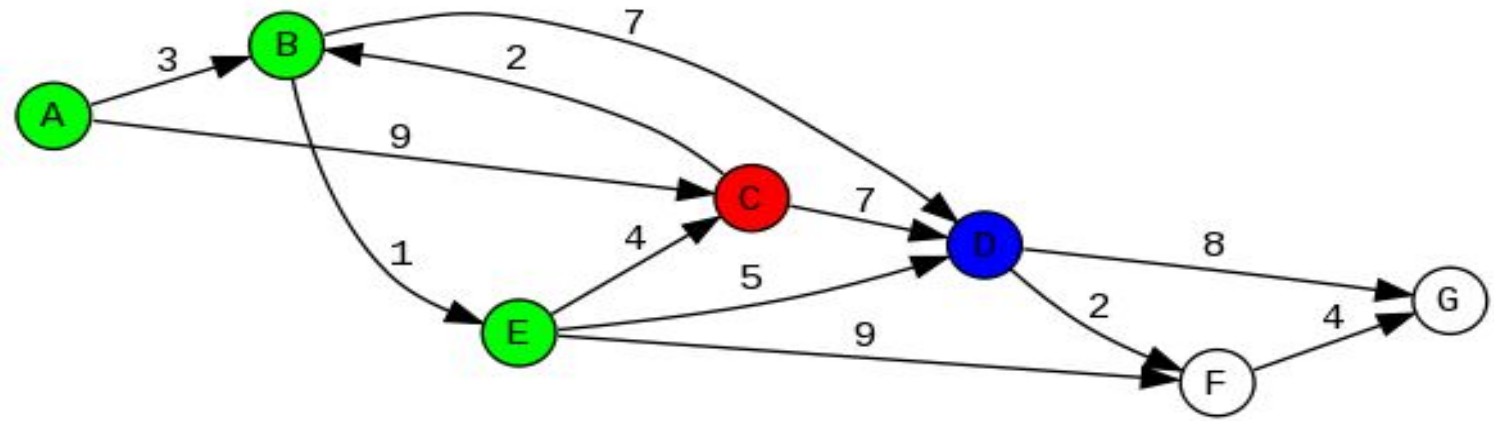
Algorytm Dijkstry - przykład - krok 4



Badamy teraz wierzchołki do których prowadzą krawędzie z C. Znajdujemy nową wartość $d(D)$ dodając wagę krawędzi CD do $d(C)$ i otrzymujemy propozycję $d(D) = 7 + 8 = 15$. Ta wartość jest WIĘKSZA niż dotychczasowa (9), więc jej nie zmieniamy, jak i wskaźnika.

Nr etapu	zbiór L	d(B)p(B),d(C)p(C),..., d(G)p(G)
3	A,B,E	$3A, 8E, 9E, 4B, 13E, +\infty$
4	A,B,E,C	$3A, 8E, 9E, 4B, 13E, +\infty$

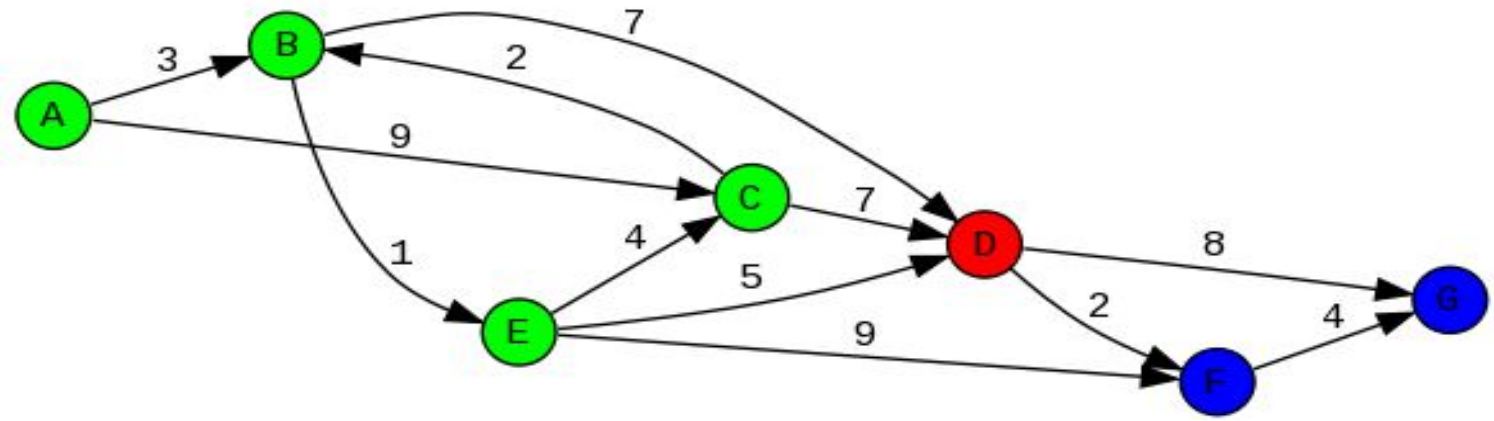
Algorytm Dijkstry - przykład - krok 4



Teraz, do zbioru L dołączamy wierzchołek, który ma najmniejszą wartość funkcji d z jeszcze niedołączonych (czyli w tym wypadku D).

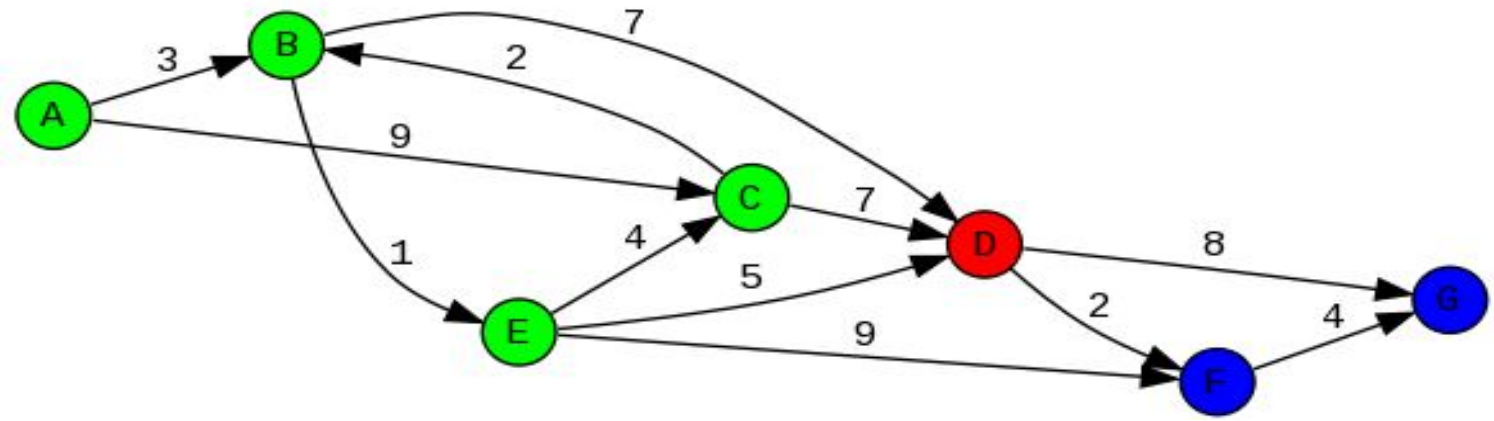
Nr etapu	zbiór L	d(B)p(B),d(C)p(C),..., d(G)p(G)
4	A,B,E,C	$3A, 8E, 9E, 4B, 13E, +\infty$
5	A,B,E,C,D	

Algorytm Dijkstry - przykład - krok 5



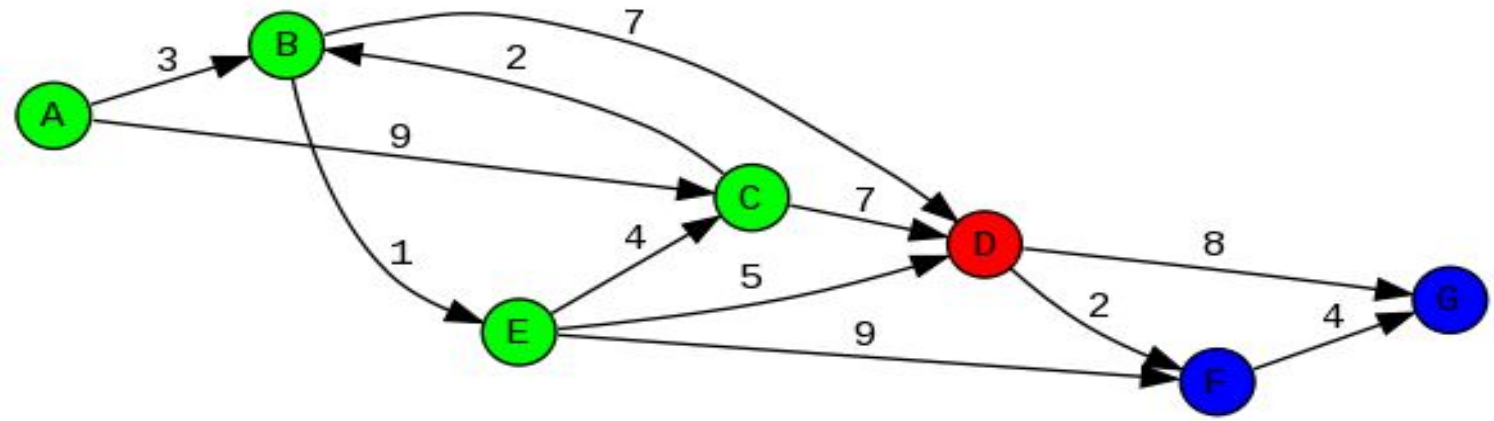
Badamy teraz wierzchołki do których prowadzą krawędzie z D.

Algorytm Dijkstry - przykład - krok 5



Badamy teraz wierzchołki do których prowadzą krawędzie z D. Znajdujemy nowe wartości $d(F)$ i $d(G)$ dodając wagi krawędzi DF i DG do $d(D)$ i otrzymujemy propozycje $d(F) = 2 + 9 = 11$, $d(G) = 8 + 9 = 17$.

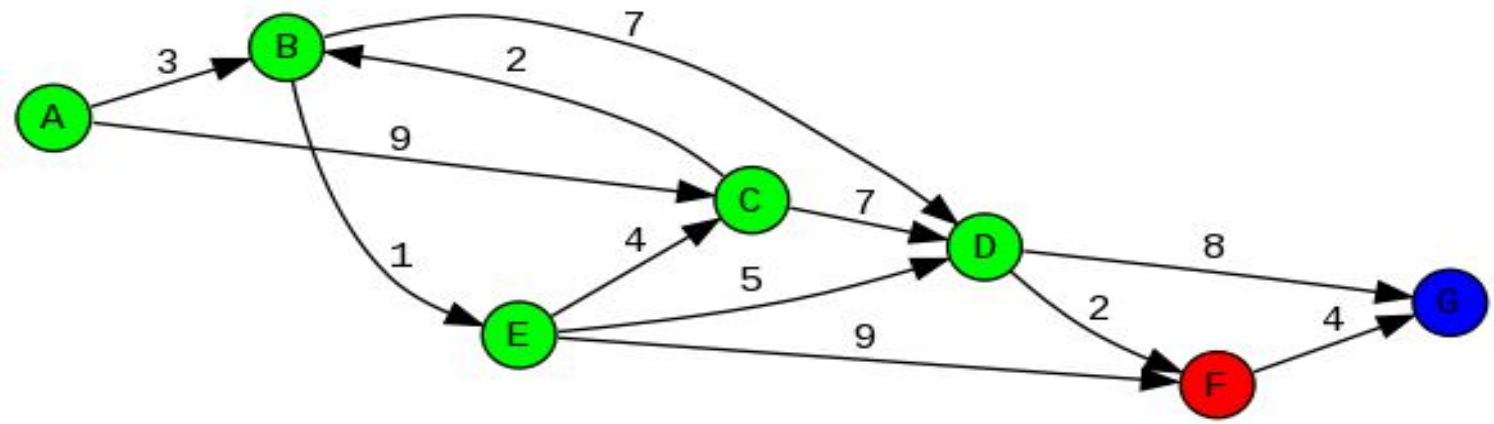
Algorytm Dijkstry - przykład - krok 5



Badamy teraz wierzchołki do których prowadzą krawędzie z D. Znajdujemy nowe wartości $d(F)$ i $d(G)$ dodając wagi krawędzi DF i DG do $d(D)$ i otrzymujemy propozycje $d(F) = 2 + 9 = 11$, $d(G) = 8 + 9 = 17$. Te wartości są mniejsze niż dotychczasowe (13 i $+\infty$), więc je zmieniamy, dopisując wskaźnik D.

Nr etapu	zbiór L	d(B)p(B),d(C)p(C),..., d(G)p(G)
4	A,B,E,C	$3A, 8E, 9E, 4B, 13E, +\infty$
5	A,B,E,C,D	$3A, 8E, 9E, 4B, 11D, 17D$

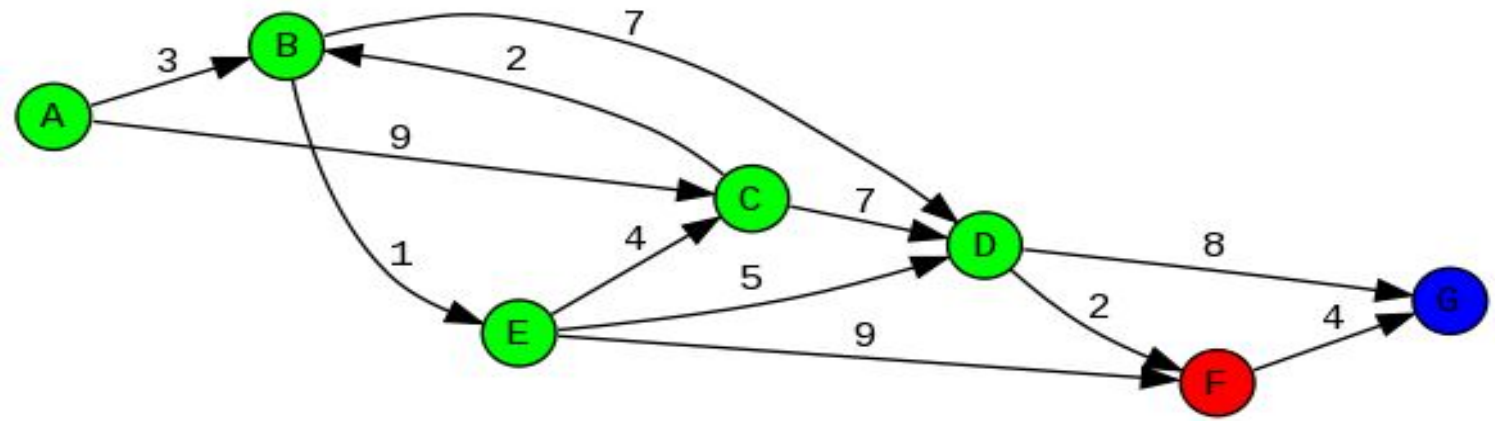
Algorytm Dijkstry - przykład - krok 5



Teraz, do zbioru L dołączamy wierzchołek, który ma najmniejszą wartość funkcji d z jeszcze niedołączonych (czyli w tym wypadku F).

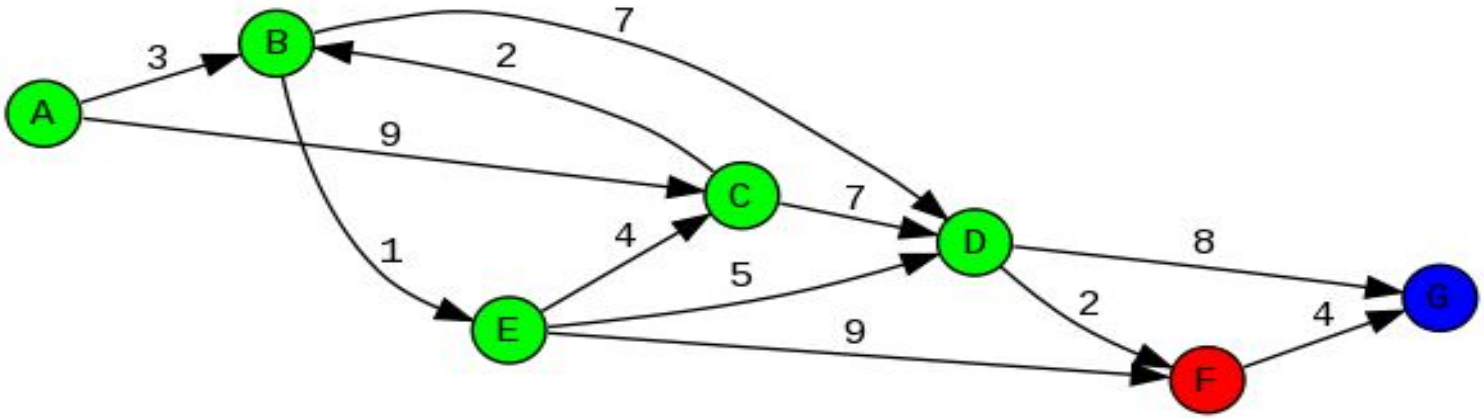
Nr etapu	zbiór L	d(B)p(B),d(C)p(C),..., d(G)p(G)
5	A,B,E,C,D	$3A, 8E, 9E, 4B, 11D, 17D$
6	A,B,E,C,D,F	

Algorytm Dijkstry - przykład - krok 6



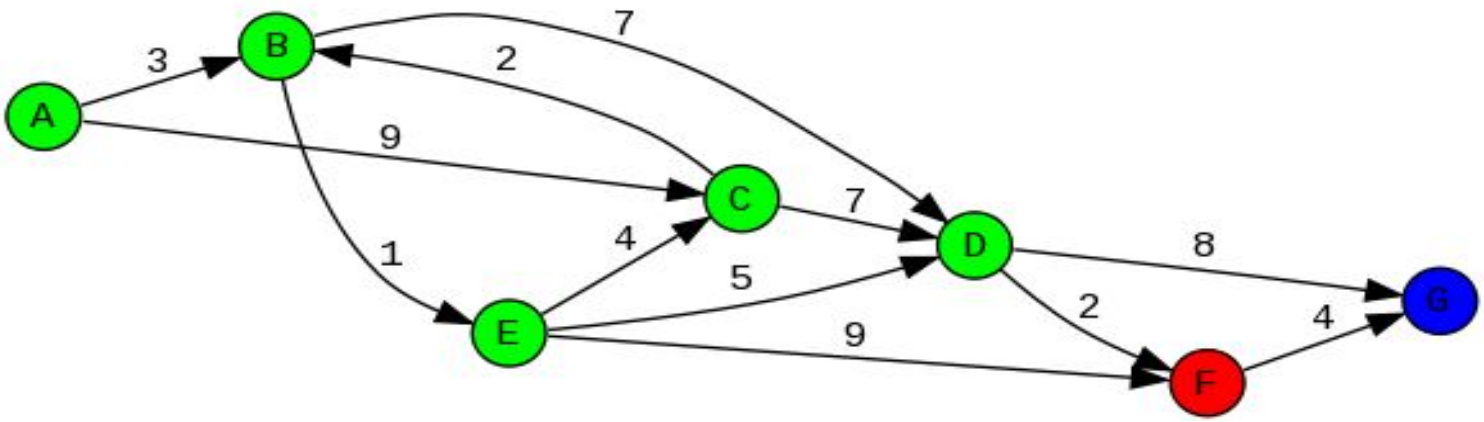
W ostatnim kroku zostało tylko sprawdzenie, czy droga przez wierzchołek F zmniejsza odległość wierzchołka G od A.

Algorytm Dijkstry - przykład - krok 6



W ostatnim kroku zostało tylko sprawdzenie, czy droga przez wierzchołek F zmniejsza odległość wierzchołka G od A. Znajdujemy nową wartość $d(G)$ dodając wagę krawędzi FG do $d(F)$ i otrzymujemy propozycję $d(G) = 11 + 4 = 15$.

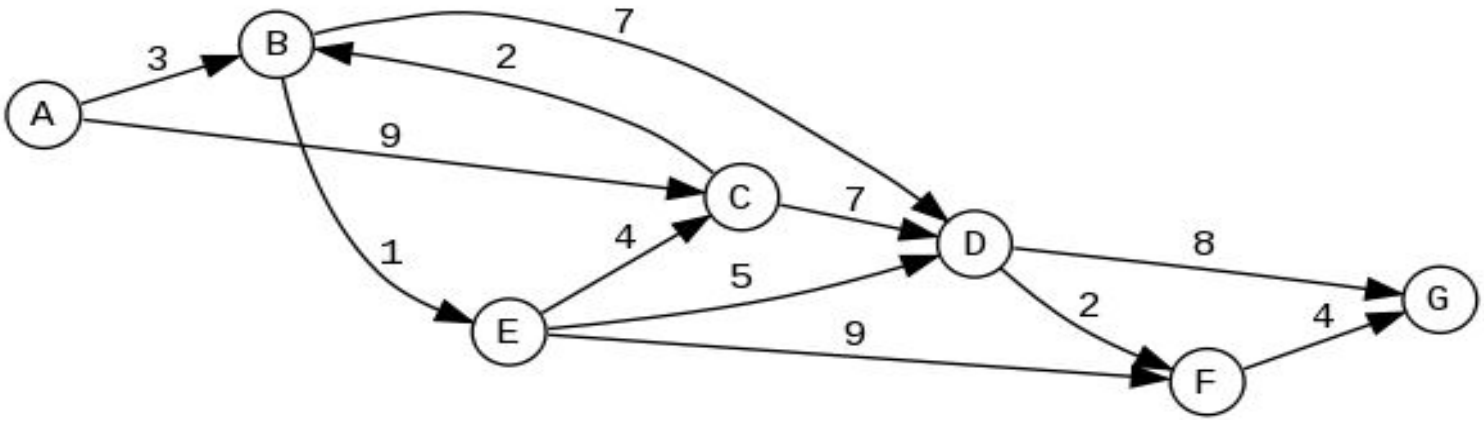
Algorytm Dijkstry - przykład - krok 6



W ostatnim kroku zostało tylko sprawdzenie, czy droga przez wierzchołek F zmniejsza odległość wierzchołka G od A. Znajdujemy nową wartość $d(G)$ dodając wagę krawędzi FG do $d(F)$ i otrzymujemy propozycję $d(G) = 11 + 4 = 15$. Te wartość jest mniejsza niż dotychczasowa (17), więc ją zmieniamy, dopisując wskaźnik F, co kończy działanie algorytmu.

Nr etapu	zbiór L	d(B)p(B),d(C)p(C),..., d(G)p(G)
5	A,B,E,C,D	$3A, 8E, 9E, 4B, 11D, 17D$
6	A,B,E,C,D,F	$3A, 8E, 9E, 4B, 11D, 15F$

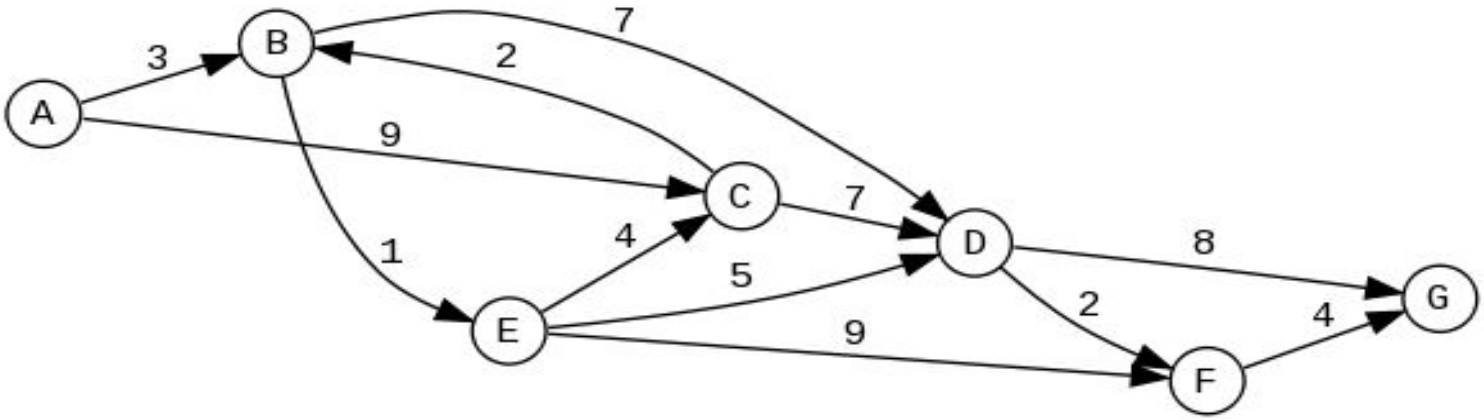
Algorytm Dijkstry - wynik



Wynikiem algorytmu jest tabela:

Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G).
1	A	$3A, 9A, +\infty, +\infty, +\infty, +\infty$
2	A,B	$3A, 9A, 10B, 4B, +\infty, +\infty$
3	A,B,E	$3A, 8E, 9E, 4B, 13E, +\infty$
4	A,B,E,C	$3A, 8E, 9E, 4B, 13E, +\infty$
5	A,B,E,C,D	$3A, 8E, 9E, 4B, 11D, 17D$
6	A,B,E,C,D,F	$3A, 8E, 9E, 4B, 11D, 15F$

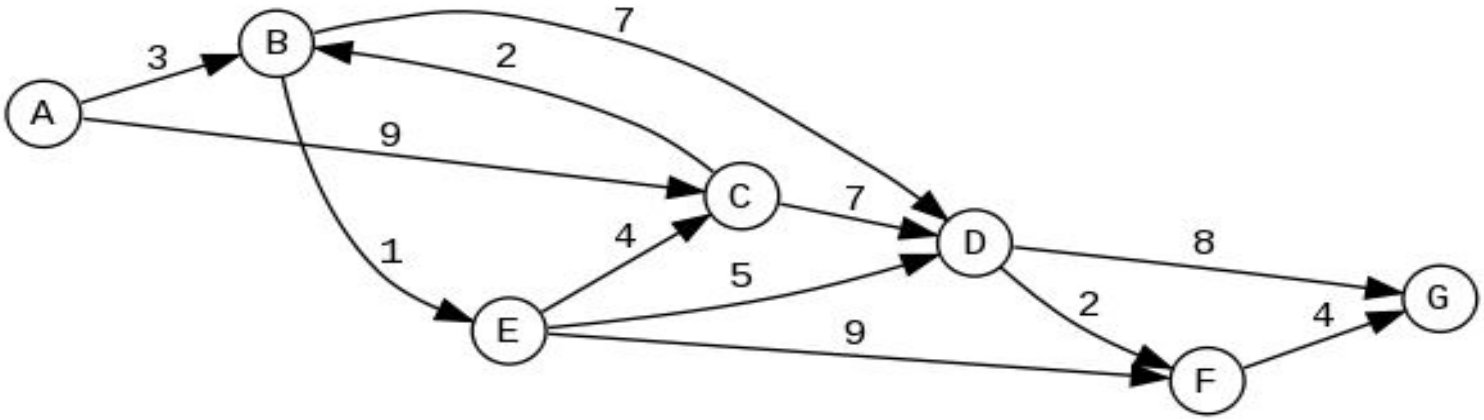
Algorytm Dijkstry - wynik



Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G).
6	A,B,E,C,D,F	$3A, 8E, 9E, 4B, 11D, 15F$

Z tabeli, dzięki wskaźnikom, możemy odczytać nie tylko długość najkrótszej drogi do danego punktu, ale i jej przebieg. Na przykład, by odczytać drogę od A do G odczytujemy: $p(G) = F$,

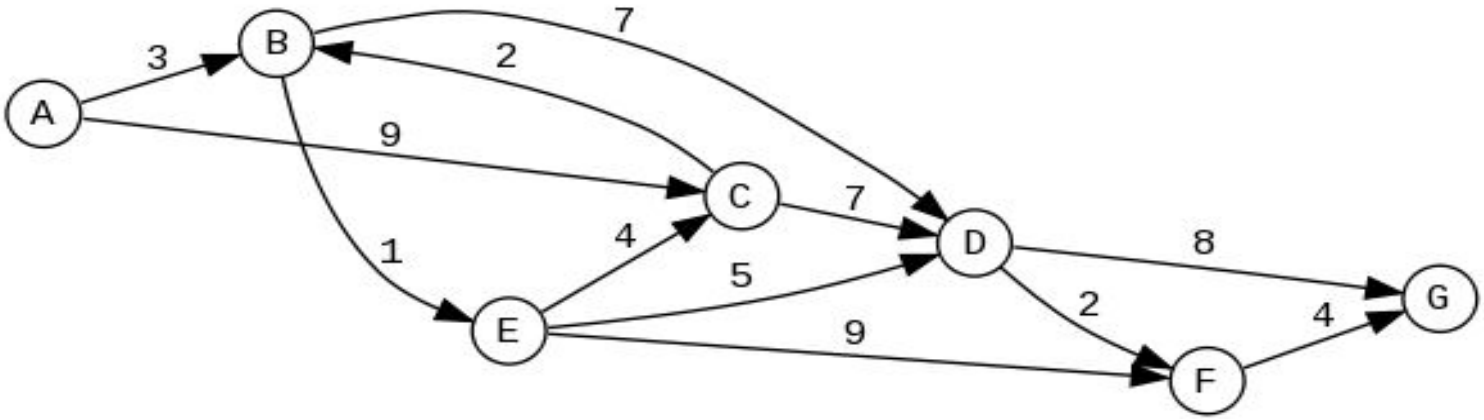
Algorytm Dijkstry - wynik



Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G).
6	A,B,E,C,D,F	$3A, 8E, 9E, 4B, 11D, 15F$

Z tabeli, dzięki wskaźnikom, możemy odczytać nie tylko długość najkrótszej drogi do danego punktu, ale i jej przebieg. Na przykład, by odczytać drogę od A do G odczytujemy: $p(G) = F, p(F) = D$,

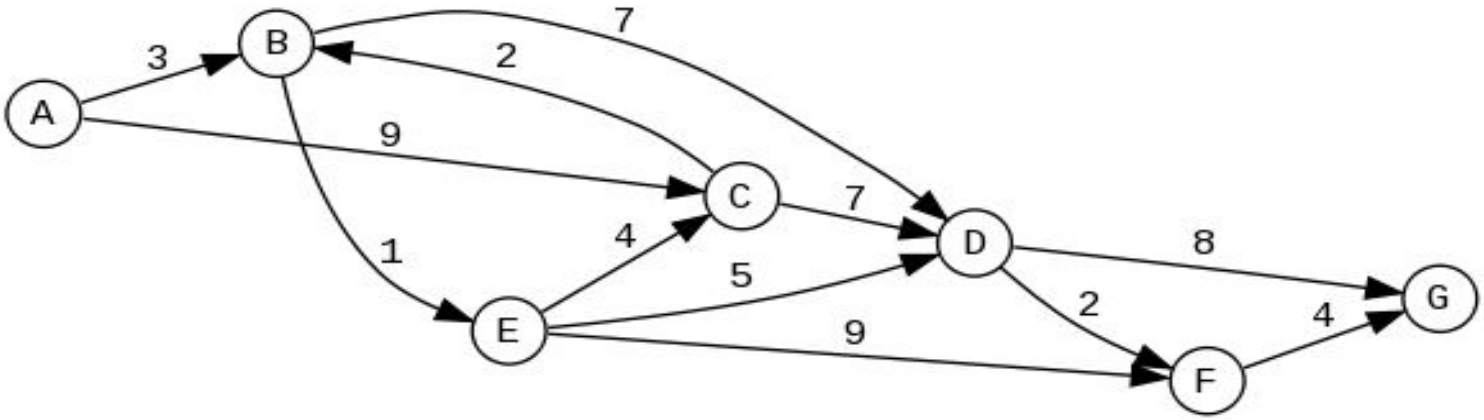
Algorytm Dijkstry - wynik



Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G).
6	A,B,E,C,D,F	$3A, 8E, 9E, 4B, 11D, 15F$

Z tabeli, dzięki wskaźnikom, możemy odczytać nie tylko długość najkrótszej drogi do danego punktu, ale i jej przebieg. Na przykład, by odczytać drogę od A do G odczytujemy: $p(G) = F$,$p(F) = D$, $p(D) = E$,

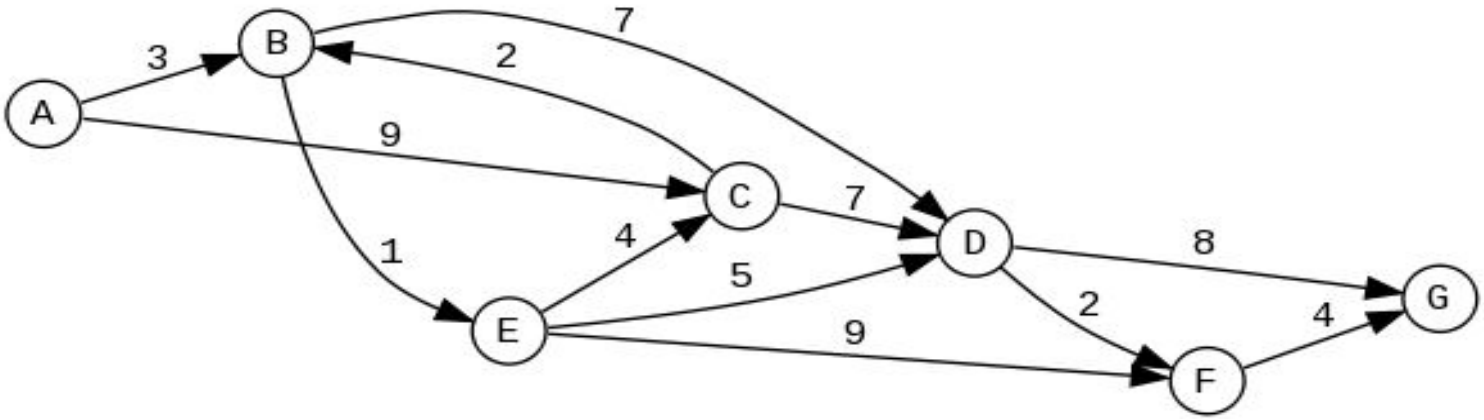
Algorytm Dijkstry - wynik



Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G).
6	A,B,E,C,D,F	$3A, 8E, 9E, 4B, 11D, 15F$

Z tabeli, dzięki wskaźnikom, możemy odczytać nie tylko długość najkrótszej drogi do danego punktu, ale i jej przebieg. Na przykład, by odczytać drogę od A do G odczytujemy: $p(G) = F$,$p(F) = D$, $p(D) = E$, $p(E) = B$,

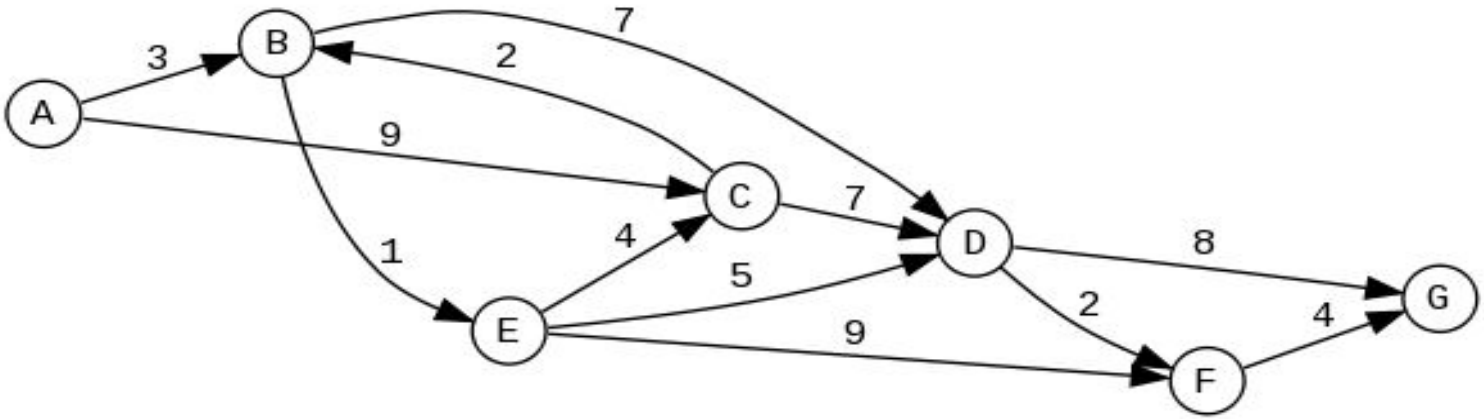
Algorytm Dijkstry - wynik



Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G).
6	A,B,E,C,D,F	$3A, 8E, 9E, 4B, 11D, 15F$

Z tabeli, dzięki wskaźnikom, możemy odczytać nie tylko długość najkrótszej drogi do danego punktu, ale i jej przebieg. Na przykład, by odczytać drogę od A do G odczytujemy: $p(G) = F$,$p(F) = D$, $p(D) = E$, $p(E) = B$, $p(B) = A$

Algorytm Dijkstry - wynik



Nr etapu	zbiór L	d(B)p(B),d(C)p(C),. . . , d(G)p(G).
6	A,B,E,C,D,F	$3A, 8E, 9E, 4B, 11D, 15F$

Z tabeli, dzięki wskaźnikom, możemy odczytać nie tylko długość najkrótszej drogi do danego punktu, ale i jej przebieg. Na przykład, by odczytać drogę od A do G odczytujemy: $p(G) = F$,$p(F) = D$, $p(D) = E$, $p(E) = B$, $p(B) = A$ i piszemy w odrotnej kolejności: $ABEDFG$.

Uwagi dotyczące algorytmu Dijkstry i jego zapisu

Uwagi dotyczące algorytmu Dijkstry i jego zapisu

- Od momentu, gdy jakiś wierzchołek zostanie włączony do zbioru L, jego wartości d i p się nie zmieniają.

Uwagi dotyczące algorytmu Dijkstry i jego zapisu

- Od momentu, gdy jakiś wierzchołek zostanie włączony do zbioru L, jego wartości d i p się nie zmieniają.
- Tym samym algorytmem można sobie radzić z grafami nieskierowanymi, traktując każdą nieskierowaną krawędź jako dwie przeciwnie skierowane krawędzie i sprowadzając zagadnienie do przypadku grafów skierowanych.

Uwagi dotyczące algorytmu Dijkstry i jego zapisu

- Od momentu, gdy jakiś wierzchołek zostanie włączony do zbioru L, jego wartości d i p się nie zmieniają.
- Tym samym algorytmem można sobie radzić z grafami nieskierowanymi, traktując każdą nieskierowaną krawędź jako dwie przeciwnie skierowane krawędzie i sprowadzając zagadnienie do przypadku grafów skierowanych.
- W każdym kroku ustalamy odległość jednego wierzchołka, więc algorytm (i jego tabelka) powinien mieć tyle kroków, ile jest wierzchołków (poza startowym) w tej samej składowej spójnej grafu. Jeśli jest więcej niż jeden wierzchołek poza L w tej samej, najmniejszej odległości od punktu startowego, wybieramy dowolny.