

Preliminaria,

Witaj

...w mojej rozległej notatce w której tłumaczę algorytm i struktury danych najprościej jak to tylko możliwe. Zakładam Twoją wiedzę o podstawach algorytmów, ponieważ staram się opisywać algorytmy najprościej jak się da - wraz z przykładami. Staram się ma zrozumienie idei zamiast ma rozwodzenie się nad rzeczami, które nie mają znaczenia.

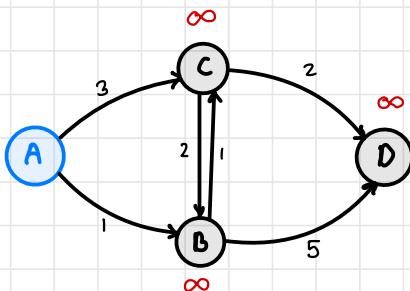
Zapnij pasy, przyjedźmyj podróż!



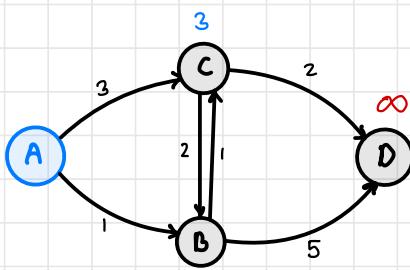
W tym rozdziale przypomnimy sobie podstawowe algorytmy. Nie będziemy się zbytnio rozwodzić nad detalami, ponieważ jest wiele świetnych materiałów w internecie.

Dijkstra

Znajduje najkrótszą ścieżkę z punktu A do każdego innego

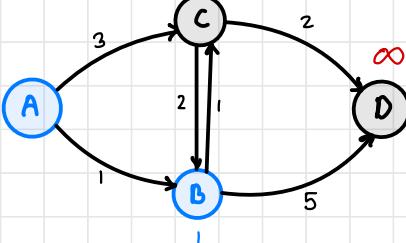


1. Sprawdź nieodkryte krawędzie dla odwiedzonych wierzchołków



2. Dokonaj relaksacji, czyli zaktualizuj odległość najkrótszych dróg dla każdego wierzchołka.

Ta relaksacja ujawnia się w następnej iteracji
32



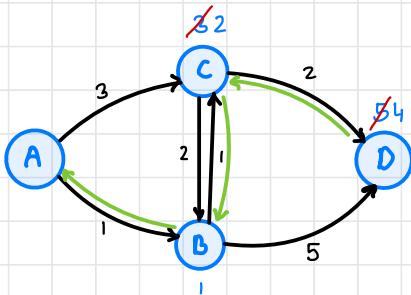
3. Odwiedź następny wierzchołek.

Odrzut ścieżki

- Zacznię od wierzchołka do którego chcesz dojść
- Iterujemy od końca do wierzchołka początkowego wypierając kolejne wierzchołki

Przykład

Odrzut dla $A \rightarrow D$



✖ - Obliczony algorytm

➡ - Odrzut ścieżki

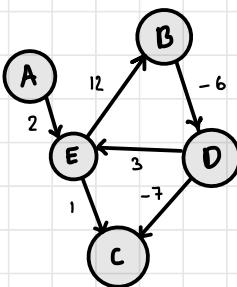
Złożoność czasowa: $O(|E| + |V| \log |V|)$ ← Pod warunkiem użycia
Algorytmu Fibonacciego...
Jeszcze do nich dojdziemy :-)

Dijkstra nie działa dla grafów z ujemnymi wagami

Jak ten problem rozwiązać? → Algorytm Bellmana-Forda

Bellman Ford

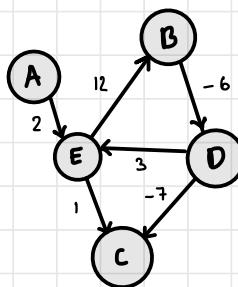
- Wykonyujemy $|V|-1$ iteracji
- Podczas każdej iteracji przedostajemy przez listę wierzchołków i dokonujemy relaksacji dla każdego z nich. Tak samo jak w algorytmie Dijkstry.



0 ∞ ∞ ∞ ∞

A B C D E

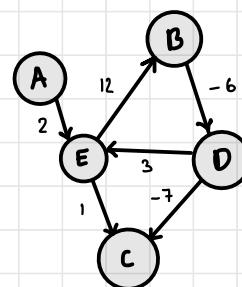
Stan początkowy



0 14 3 ∞ 2

A B C D E

Iteracja I



0 14 1 8 2

A B C D E

Iteracja II

Z każdym razem sprawdzamy doąd dojścia i dokonujemy relaksacji. Np. z A dojście do E, potem to samo dla B, C, D: E.

Zostanie wykonała jeszcze iteracja III i IV, ale nie się już nie zmieni dla przypadku tego grafu.

Złożoność czasowa: $O(|V| \cdot |E|)$

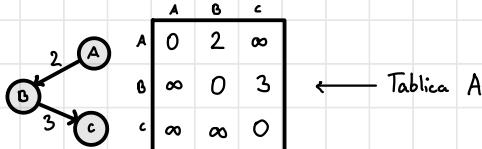
Czy jeśli chcemy znaleźć wszystkie najkrótsze ścieżki dla każdego v

Jak ten problem rozwiązać? \rightarrow Algorytm Floyd-Wallrella

Floyd Warshall

Ten algorytm, to zastosowanie Bellmana-Forda ma większą skalę.

1. Przygotuj macierz incydencji dla grafu gdzie ma przekątną się zerą



2. Iteracyjnie dokonuj relaksacji

for w from 1 to $|V|$

Kolejność jest ważna!

for x from 1 to $|V|$

for y from 1 to $|V|$

if $A[x, y] > A[x, w] + A[w, y]$

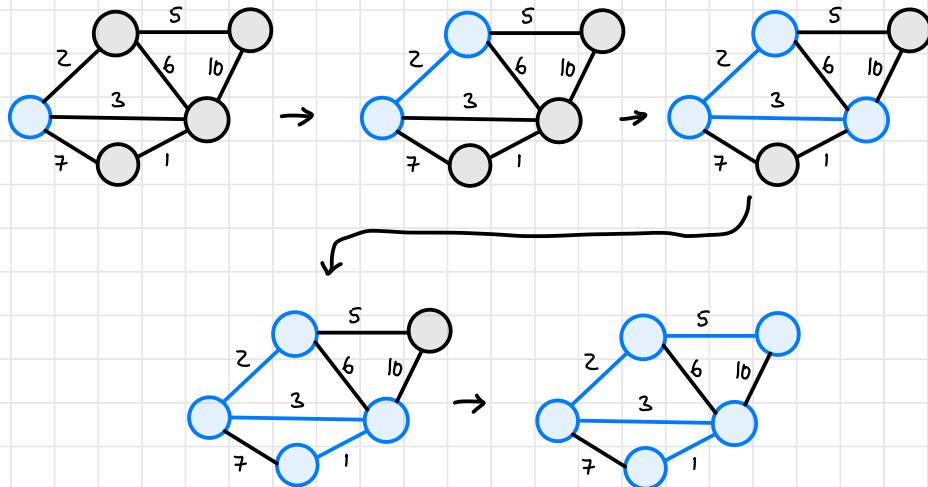
$$A[x, y] = A[x, w] + A[w, y]$$

Złożoność obliczeniowa Flouda Warrella to
 $O(|V|^3)$

Prim

Znajduje mając najmniejsze drzewo rozpinające w grafie.

- Zaczyna od pierwszego wierzchołka i wybiera taki, do którego jest najtańsza krawędź.
- W każdej iteracji rozwija istniejący zbiór krawędzi Tychących wierzchołków drzewa z wierzchołkami spoza drzewa.

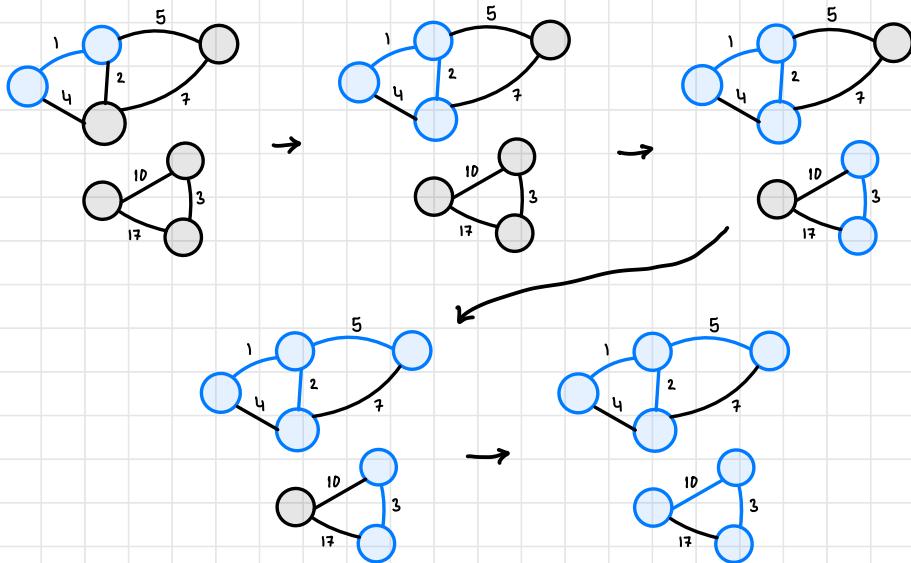


Gdy graf jest niespojny, Prim znajdzie tylko jedno MST

Jak ten problem rozwiązać? \rightarrow Kruskal

Kruskal

- Wybieramy za każdym razem krawędzie spoza drzewa, które są najmniejsze; nie tworzą cyklu.



Jak widać udało nam się zbudować los drzew MST.

Kruskal wykorzystuje strukturę UnionFind, która usprawnia złożoność algorytmu.

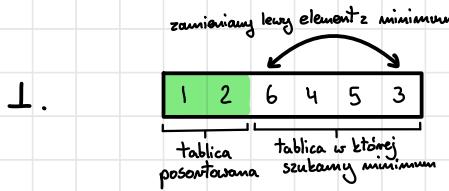
↑ o tym wróć

Wtedy złożoność wynosi $O(|E| \log |E|)$, ponieważ musimy posortować wierzchołki i iterować od najmniejszego do największego.

Sprawdzanie czy wierzchołek jest w drzewie oraz dodawanie go do drzewa zapewnia nam strukturę UnionFind w $O(\log^* n) \approx O(1)$ w amortyzowanej złożoności czasowej.

Selection Sort

1. Znajdź minimum w nieposortowanej podtablicy i wykonaj swap z następującym elementem obok tablicy posortowanej.
2. Powiększ rozmiar tablicy posortowanej o 1.



Selection sort jest niestabilny

↑
Tzn. że elementy, które są sobie równe, nie zachowują tej samej kolejności w tablicy po posortowaniu.

Złożoność czasowa Selection Sorta wynosi $O(n^2)$

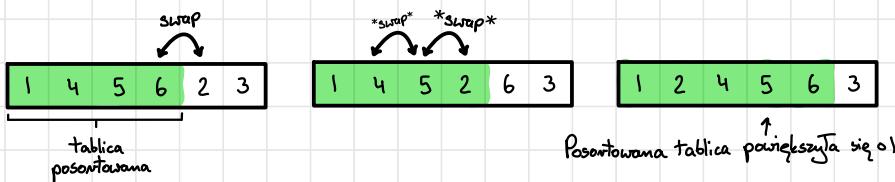
Insertion Sort

- Traktujemy pierwszy element w tablicy jako już posortowany.
- Zaczynamy od elementu o indeksie 1.

Algorytm

1. Iterujemy się przez tablicę. W każdej iteracji:

- a) Wstaw aktualny element do posortowanej tablicy
sekwencją funkcji **swap**, aż gdy będzie mniejszy niż jego następnik.



Insertion Sort jest stabilny, a jego złożoność to $O(n^2)$

Zadania Egzaminacyjne

1. Które z poniższych algorytmów mogą działać niepoprawnie dla grafów z ujemnymi wagami.

- Algorytm Kruskala - Działa Dobrze ✓
- Algorytm Prima - Działa Dobrze ✓
- Algorytm Dijkstry - Działa Źle ✗

2. Rozwiąż równanie rekurencyjne

$$\begin{cases} T(1) = 1 \\ T(2) = 3 \\ T(n) = T(n-2) + 2n - 1 \end{cases}$$

$$\begin{aligned} T(n) &= 2n - 1 + 2(n-2) - 1 + 2(n-4) - 1 + \dots + 3 + 1 = \\ &= n + n - 1 + n - 2 + n - 3 + T(n-4) = \\ &= \frac{n(n+1)}{2} = O(n^2) \end{aligned}$$

3. W jaki sposób DFS może być wykorzystany do znalezienia cyklu Eulera?

Wykonujemy DFS tylko na nieodwiedzonych wierzchołkach i dodajemy do listy wierzchołków. Sprawdzamy, czy ostatni i pierwszy wierzchołek są ze sobą połączone.

4. W jaki sposób problem mnożenia macierzy może być wykorzystany do rozwiązywania problemu majątkowych ścieżek?

- ZasymulujemyAlgorytm Floyda Warshalla
- Weźmy graf A jako macierz sąsiedztwa

$$c_{xy} = \sum_i a_{iy} \cdot b_{xi}$$

Gdzie podniemiamy "+" na min, a ":" na sumę

Wtedy mamy:

$$c_{xy} = \min_{\forall i \in \{1, \dots, n\}} (a_{iy} + b_{xi})$$

5. Oblicz równanie rekurencyjne

$$T(n) = T(\sqrt{n}) + O(1)$$

Podstawmy $m = 2^k$

$$T(2^k) = T(2^{\frac{k}{2}}) + O(1)$$

$$S(k) = S\left(\frac{k}{2}\right) + O(1)$$

Z master theorem

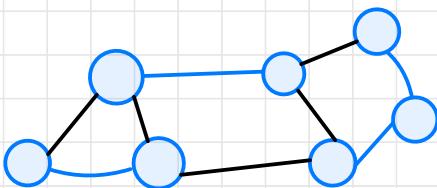
$$S(k) \rightarrow \Theta(\log k)$$

Wiem, że $k = \log n$

Zatem

$$\Theta(\log k) = \Theta(\log \log n)$$

6. Z ile drzew może się składać las spajający
utworzony przez algorytm Kruskala po
4 krawędziach grafu o 7 wierzchołkach



Dokładnie z trzech.

Więcej być nie może,
bo musieli by być > 7 wierzchołków

Mniej być nie może,
bo musielibyśmy rozpatrywać < 4 krawędzie