Znajdowania maksymalnego podgrafu spójnego - opis algorytmu dokładnego

Pamela Krzypkowska, Michał Kortała

6 listopada 2018

1 Założenia

Mamy dwa grafy spojne, które charakteryzują się zbiorem wierzchołków i krawędzi.
Można te grafy opisać w poniższy sposób: $G1 = (V1, E1), G2 = (V2, E2)$.
Reprezentacja grafu to macierz sąsiedztwa M[,] o wymiarach nxn, gdzie n - liczba wierzchołków grafu.
Jeżeli istnieje krawędz miedzy wierzchołkiem v i u, to w macierzy sasiedztwa M[v,u] = 1 oraz M[u,v] = 1, gdyż grafy nie sa skierowane. Dla ustalenia uwagi przyjmiemy, że $|V1| = n, |V2| = m, |E1| = k, |E2| = porazn <= m$. Wynika z tego, że każdy wierzchołek G_1 musi być uwzględniony mapowaniu.

2 Opis algorytmu

Opisany algorytm jest modyfikacją algorytmu do znajdywania maksymalnego wspólnego podgrafu autorstwa J.J. McGregora do celów postawionych w zadaniu.Działanie algorytmu opiera się na idei algorytmu z nawrotami. Do znalezienia maksymalnego wspólnego podgrafu wykorzystujemy dwie tablice VMatch i EMatch.

G_1 G_2	u_1	u_2	...	u_m
v_1				
v_2				
...				
v_n				

1.Tablica VMatch.

W tablicy VMatch przechowujemy informację na temat dopasowania wierzchołków z grafu G_2 do wierzchołków G_1. Gdy wierzchołek z grafu G_2 jest dopasowany w tablicy do więcej niż jednego z G_1 oznacza to, że może być dopasowany do któregokolwiek z nich.

E_1 E_2	f_1	f_2	...	f_p
e_1	1	1	1	1
e_2	1	1	1	1
...	1	1	1	1
e_k	1	1	1	1

2.Tablica EMatch.

W tablicy EMatch przechowujemy informacje na temat odpowiadania sobie

krawędzi. Wartość 1 w komórce EMatch(r,s) informuję o tym, że krawędzie r i s mogą sobie odpowiadać. W momencie utworzenia tablica wypełniona jest samymi 1.
Przebieg algorytmu wygląda w następujący sposób. Dopóki i != 0, dla każdego wierzchołka v_i grafu G_1 sprawdzamy niesprawdzony jeszcze wierzchołek u_j grafu G_2. Dla każdego takiego sprawdzenia aktualizujemy tablicę EMatch tak, aby wartość 1 znajdowała się w komórce tabeli o indeksach (r,s) jesli krawędź e_r jest połączona z wierzchołkiem v_i, a krawędź f_s z wierzchołkiem u_j. Jeśli i = n to sprawdzamy czy dla wybranego mapowania wierzchołków z tablicy VMatch oraz mapowania krawędzi z tablicy EMatch uzyskaliśmy graf spójny. Jeśli tak i lepiej spełnia warunki bycia maksymalnym to nadpisujemy nim znaleziony maksymalny podgraf. Jeśli i != n zapisujemy kopię tablicy EMatch dla wierzchołka o numerze i oraz zwiększamy i o 1 zaznaczając przy tym wszystkie wierzchołki G_2 dla nowej wartości i jako niesprawdzone. Jeśli nie ma już więcej niesprawdzonych wierzchołków w grafie G_2 dla aktualnej wartości i to zmniejszamy wartość i o jeden i przywracamy kopię tablicy EMatch powiązanej z nową wartością i.

Pseudokod:

```
i = 1, oznacz wszystkie wartości VMatch[i,:] jako niesprawdzone.
EMatch[:,:] = 1
while i!=0 do
    if są niesprawdzone wartości w VMatch[i,:] then
        u_j = niesprawdzona wartość, VMatch[i,j] = sprawdzona
         aktualizuj tablicę EMatch
        if i == n then
            zbuduj wspólny podgraf na podstawie VMatch i EMatch
            if spójny AND maksymalny then
                zapisz podgraf
            else

            end
        else
            zapamiętaj wartość EMatch dla v_i
            i = i + 1
            oznacz VMatch[i, :] jako niesprawdzone
        end
    else
        i = i - 1
        przywróć wartość EMatch dla v_i
    end
end
```

3 Dowód poprawności

Poprawność zaprezentowanego algorytmu wynika z faktu, że szukając mapowań pomiędzy wierzchołkami grafów G_1 i G_2 sprawdzimy wszystkie możliwe kombinacje ze względu na ilość wynikowych wierzchołków (od 1 do n) co da pewność znalezienia maksymalnego pod względem liczby wierzchołków wspólnego podgrafu spójnego. W tablicy EMatch przechowywana będzie informacja o dostępnych krawędziach co pozwoli użyć ich maksymalną liczbę a zatem wypełnić kryterium maksymalności ze względu na liczbę krawędzi i wierzchołków.

4 Złożoność

W tablicy VMatch o wymiarach n x n do wierzchołka reprezentowanego przez wiersz mapowany może być tylko jeden wierzchołek z drugiego grafu. Zatem pierwszy wiersz wypełnimy raz, drugi n razy, trzeci n^2 razy... n-ty $n^(n-1)$ razy. Wynika z tego, że liczba operacji w tablicy VMatch wynosi:

$$\sum_{i=0}^{n} \quad n^i$$

Dla każdej takiej operacji musimy zaktualizować tablicę EMatch. Liczba zmian w tej tablicy jest równa iloczynowi stopni mapowanych obecnie wierzchołków. A więc w najgorszym przypadku

$$(n-1) * [(n-1) * n/2 - (n-1)]$$

Wynika z tego zatem, że złożoność algorytmu możemy ograniczyć z dołu przez:

$$C = (\sum_{i=0}^{n} \quad n^i) * (n-1) * [(n-1) * n/2 - (n-1)]$$

$$C = O(n^n)$$