Funkcja low, mosty, punkty artykulacji

Bartosz Chomiński

1 Wstęp

Rozwiążemy w tej notatce pewne dwa klasyczne siostrzane zadania w teorii grafów:

Wyznacz wszystkie krawędzie, których usunięcie rozspójni graf.

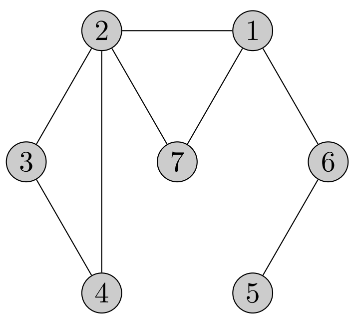
oraz

Wyznacz wszystkie wierzchołki, których usunięcie rozspójni graf.

Krawędzie rozspajające graf nazywamy *mostami*, natomiast wierzchołki rozspajające graf nazywamy *punktami artykulacji*.

Jeśli początkowy graf nie był spójny, to będą nas interesować wierzchołki lub krawędzie, które rozspójnią spójny fragment grafu, w którym były.

Przykładowo, w grafie



mosty to krawędzie $\{1, 6\}$ oraz $\{5, 6\}$, a punkty artykulacji to 1, 2 oraz 6.

W szczególności wierzchołek 5 *nie* jest punktem artykulacji, ponieważ po jego usunięciu graf nadal jest spójny.

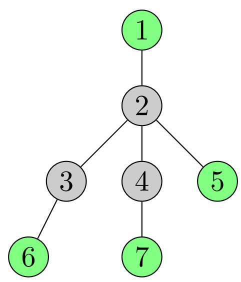
2 Pierwsze obserwacje

Rozwiążmy nasze dwa siostrzane zadania najpierw w jakikolwiek sposób. Oczywiście moglibyśmy sprawdzać wprost co się dzieje z grafem po usunięciu którejś krawędzi lub któregoś wierzchołka. Ten bezpośredni algorytm wymaga uruchomienia dowolnego przeszukiwania grafu dla każdego przypadku usuwanej krawędzi lub usuwanego wierzchołka i ma czas działania $\mathcal{O}(M(N + M))$ (dla krawędzi) lub $\mathcal{O}(N(N + M))$ (dla wierzchołków).

Spróbujmy rozwiązać nasze dwa zadania na mniejszej klasie grafów – często jest tak, że zadanie trudne dla grafów w ogólności staje się łatwiejsze dla drzew, cykli czy DAGów (np. problem wyznaczenia średnicy grafu albo znalezienia idealnego skojarzenia).

Rozważmy więc nasze zadania na drzewach (czyli spójnych nieskierowanych grafach bez cykli). Łatwo możemy zauważyć, że każdy wierzchołek, który nie jest liściem jest punktem artykulacji, a każda krawędź jest mostem.

W poniższym drzewie mosty to wszystkie krawędzie, a punkty artykulacji to wierzchołki niezaznaczone na zielono (czyli dokładnie wszystkie nieliście).

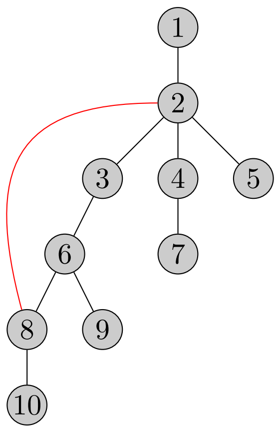


Spróbujmy nieco uogólnić nasze drzewo.

3 Drzewa z dodatkowymi krawędziami

3.1 1-drzewa

Skoro już umiemy poprawnie i efektywnie rozwiązać oba nasze zadania dla drzew, to rozważmy drzewa z dodatkową jedną krawędzią (w literaturze znane jako *1-drzewa*). Przykładowe 1-drzewo wygląda jak niżej.



Tym razem rozwiązanie ma ciekawszą strukturę: wszystkie krawędzie na ścieżce od 2 do 8 i czerwona krawędź nie będą mostami, ponieważ leżą na cyklu, a usunięcie dowolnej krawędzi cyklu nie rozspójnia go.

W zadaniu „wierzchołkowym” sytuacja jest jeszcze ciekawsza: punktami artykulacji przestały być te wierzchołki na ścieżce od 2 do 8, które mają tylko jedno dziecko (poza ewentualnie końcami tej ścieżki).

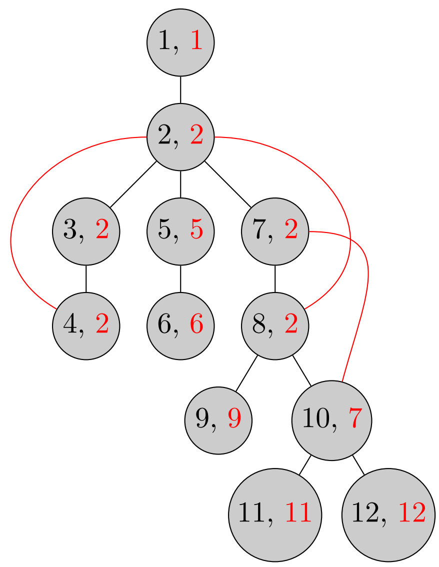
3.2 Uogólnienie

Teraz rozważmy drzewa z wieloma dodatkowymi krawędziami, ale takimi, które łączą tylko wierzchołki z ich przodkami. Bardzo ważna cecha tego typu drzew jest taka, że każdy graf nieskierowany można przedstawić jako właśnie takie drzewo. Można to zrobić wykonując przeszukiwanie grafu w głąb (DFS) i uznając krawędzie, po których przejdzie DFS jako krawędzie drzewowe, natomiast pozostałe uznając za dodatkowe.

Gdy rozważaliśmy 1-drzewa, mosty i punkty artykulacji wyznaczaliśmy w oparciu o to, czy dana krawędź lub dany wierzchołek były „pod[1]” dodatkową (czerwoną) krawędzią. Tym razem tych dodatkowych krawędzi będzie więcej, ale nadal będzie nas interesować to, które wierzchołki i krawędzie można „obejść” korzystając z czerwonych krawędzi.

Zdefiniujmy tablicę `low`, która precyzyjnie opisze jak wysoko w drzewie możemy dojść z danego wierzchołka, jeśli możemy przejść tylko raz dowolną czerwoną krawędzią, a krawędziami czarnymi możemy chodzić tylko w dół.

Oto przykład drzewa z dodatkowymi krawędziami i wartości tablicy `low` dla tego drzewa (wartości `low` dla poszczególnych wierzchołków to czerwone liczby).



Przykładowo, `low`$[10] = 7$, ponieważ z wierzchołka 10 możemy przejść czerwoną krawędzią do wierzchołka 7, potem w dół do wierzchołka 8, ale już nie możemy przejść czerwoną krawędzią w górę (bo skorzystaliśmy już z jakiejś czerwonej krawędzi) i nie możemy przejść czarną krawędzią w górę. W ten sposób nie możemy dojść wyżej, niż do wierzchołka 7, zatem wynik to 7.

Jeśli będziemy mieli taką tablicę, to łatwo będziemy mogli sprawdzić czy dana krawędź drzewowa (u, v) (u jest rodzicem v) jest mostem – jeśli `low`$[v] = v$, to nie możemy przejść w górę drzewa inaczej niż krawędzią (u, v), czyli jest ona mostem; jeśli `low`$[v] \neq v$, to mamy jakieś inne połączenie w górę drzewa, a zatem usunięcie krawędzi (u, v) nie rozspójni drzewa, czyli ta krawędź nie będzie mostem.

3.3 Funkcja `low`

Od teraz zakładamy, że wierzchołki drzewa z dodatkowymi krawędziami, które rozpatrujemy mają numerację *pre-order*, czyli zgodną z kolejnością ich rozpatrywania w algorytmie DFS.

[1] W przykładzie 1-drzewa powyżej krawędzie $(2, 3)$, $(3, 6)$ oraz $(6, 8)$ są „pod” czerwoną krawędzią dodatkową.

Znamy już definicję tablicy `low`, więc nauczmy się jak ją efektywnie wyznaczyć. Na początek wyznaczmy wartości `low` dla liści drzewa. Z liści nie możemy pójść w dół czarnymi krawędziami, bo ich nie ma. Zostaje jedynie użycie (raz!) czerwonej krawędzi, stąd wzór:

$$\texttt{low}[v] = \min(v, \min_{(u,v)\in\text{czerwone}} u) \quad \text{dla } v \text{ będącego liściem.}$$

Teraz możemy wyznaczać dalej wartości `low` – od liści do korzenia. Ogólny wzór jest nieco bardziej skomplikowany niż ten dla liści:

$$\texttt{low}[v] = \min(v, \min_{(u,v)\in\text{czerwone}} u, \min_{c\in\text{dzieci } v} \texttt{low}[c]).$$

Pierwszy argument minimum w powyższym zbiorze to v, bo z wierzchołka zawsze można dojść do niego samego. Drugi argument odpowiada za ten jeden skok czerwoną krawędzią, na jaki możemy sobie pozwolić (nie opłaca się chodzić dalej z wierzchołka u, bo wyżej już z niego nie dojdziemy zgodnie z zasadami). Ostatni argument odpowiada za to, że możemy przejść w dół, do swojego dziecka, czarną krawędzią i z niego chodzić dalej.

3.4 Podsumowanie zastosowania

Czarna krawędź (u, v) (u jest rodzicem v) jest mostem wtedy i tylko wtedy, gdy $\texttt{low}[v] = v$.

Żadna czerwona krawędź nie jest mostem.

Korzeń drzewa jest punktem artykulacji wtedy i tylko wtedy, gdy ma więcej niż jedno dziecko.

Wierzchołek v niebędący korzeniem jest punktem artykulacji wtedy i tylko wtedy, gdy posiada dziecko c, dla którego $\texttt{low}[c] \geqslant v$.

3.5 Dowody

Udowodnimy twierdzenie łączące tablicę `low` z byciem punktem artykulacji.

($\implies$). Załóżmy, że v jest punktem artykulacji. To oznacza, z definicji, że jego usunięcie rozspójni graf. Jedną z jego składowych musi być wtedy poddrzewo jakiegoś dziecka v – oznaczmy to dziecko jako c. Poddrzewo c nie może być połączone żadnymi czerwonymi krawędziami z przodkami v (ale z v już może!), a zatem najwyższym wierzchołkiem, do którego można dojść z c czarnymi krawędziami w dół i jedną czerwoną krawędzią jest v. Stąd $\texttt{low}[c] \geqslant v$.

($\impliedby$). Załóżmy, że c jest dzieckiem v oraz $\texttt{low}[c] \geqslant v$. To oznacza, że z wierzchołka c nie można dojść wyżej niż do v chodząc czarnymi w dół i maksymalnie jedną czerwoną. To oznacza, że po usunięciu wierzchołka v, z poddrzewa c nie wyjdą już żadne czerwone krawędzie wyżej, czyli to poddrzewo będzie całkowicie odłączone od reszty grafu z przodkami v – czyli graf będzie niespójny, a więc v będzie punktem artykulacji.

4 Zobacz też

- `https://cp-algorithms.com/graph/bridge-searching.html`
- `https://cp-algorithms.com/graph/cutpoints.html`
- `http://informatyka.wroc.pl/node/752?page=0,2`