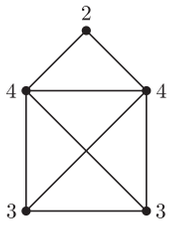
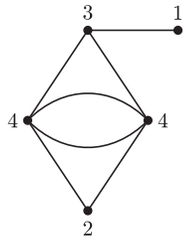
Odtwarzanie grafu

Jakub RADOSZEWSKI



Rys. 1



Rys. 2

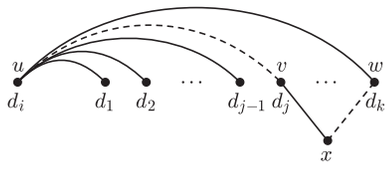
W grafie nieskierowanym możemy obliczyć stopień każdego wierzchołka, czyli liczbę krawędzi incydentnych z tym wierzchołkiem. Przykładowo, dla grafu-koperty (rys. 1) otrzymujemy w ten sposób ciąg stopni 4, 4, 3, 3, 2. Wykonanie takiego przekształcenia dla danego grafu jest naprawdę proste. Możemy jednak postawić pytanie odwrotne: czy mając dany ciąg liczb, możemy stwierdzić, czy odpowiada on stopniom wierzchołków jakiegoś grafu nieskierowanego, a jeśli tak, zrekonstruować ten graf?

Weźmy, na przykład, ciąg 4, 4, 3, 3, 1 – czy jest on ciągiem stopni jakiegoś grafu? Tutaj łatwo udzielić odpowiedzi negatywnej. Faktycznie, każda krawędź w grafie wpływa na zwiększenie stopni dwóch wierzchołków o jeden, czyli suma stopni musi zawsze być parzysta. A teraz coś bardziej skomplikowanego: ciąg 4, 4, 3, 2, 1. Jeśli dopuszczamy krawędzie wielokrotne, graf możemy odtworzyć, na przykład, tak jak na rysunku 2. To jest jednak multigraf – w „zwykłym" grafie nie możemy mieć krawędzi wielokrotnych ani pętli. Po chwili kombinowania można sprawdzić, że podany ciąg nie odpowiada żadnemu zwykłemu grafowi. Przydałby się jakiś uniwersalny przepis na takie sprawdzanie.

Na szczęście taki przepis – algorytm – istnieje. Jest on przykładem podejścia **zachłannego**: wybieramy dowolny wierzchołek grafu, czyli element ciągu stopni, po czym łączymy go krawędziami z wierzchołkami o możliwie najwyższych stopniach. Następnie zmniejszamy stopnie tych wierzchołków, usuwamy wybrany wierzchołek z grafu i powtarzamy to samo od początku. Jeśli w pewnym momencie nie uda się znaleźć odpowiedniej liczby wierzchołków o dodatnich stopniach, kończymy algorytm z wynikiem negatywnym. Rozważmy, dla przykładu, początkowy ciąg 4, 4, 3, 3, 2. Wybieramy wierzchołek o stopniu 3 i łączymy go z wierzchołkami o stopniach 4, 4, 3. Nowy ciąg to 3, 3, 2, 2. Wybierzmy ponownie wierzchołek o stopniu 3; łączymy go ze wszystkimi pozostałymi i otrzymujemy ciąg 2, 1, 1. Jeśli teraz ponownie wybierzemy wierzchołek o najwyższym stopniu, to uda nam się skonstruować cały graf, dokładnie taki jak na rysunku 1. Gdybyśmy natomiast zaczęli od ciągu 4, 4, 3, 2, 1 i wybrali kolejno pierwsze dwa wierzchołki (te o początkowym stopniu 4), już w drugim kroku – ciąg 3, 2, 1, 0 – otrzymalibyśmy odpowiedź negatywną.

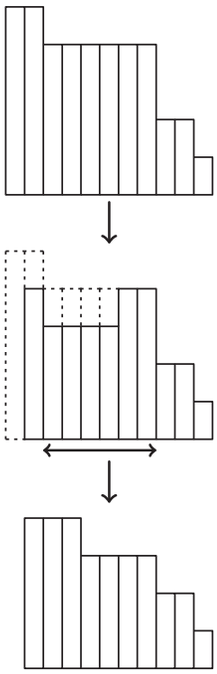
Wykażemy, że ta metoda, znana też pod nazwą algorytmu Havla–Hakimiego, zawsze daje poprawne wyniki. W tym celu wystarczy udowodnić, że jeśli dla danego ciągu liczb istnieje graf, dla którego rozważany ciąg jest ciągiem stopni wierzchołków, to nasz algorytm taki graf znajdzie. Załóżmy, przez zaprzeczenie, że nierosnący ciąg $d_1, \ldots, d_n$ odpowiada stopniom wierzchołków pewnego grafu G, ale nasz algorytm uruchomiony dla ciągu (d_i) zwrócił odpowiedź negatywną. Niech d_i będzie stopniem pierwszego wierzchołka rozważanego w algorytmie; nazwijmy ten wierzchołek u. W naszym algorytmie próbujemy połączyć wierzchołek u z wierzchołkami o stopniach $d_1, d_2, \ldots$ (z pominięciem samego d_i, rzecz jasna). Skoro graf G istnieje, to na pewno dla wierzchołka u musi nam się to udać.

Spytajmy zatem, z jakimi wierzchołkami w G jest połączony wierzchołek u. Jeśli z tymi samymi co w naszym algorytmie, to możemy usunąć wierzchołek u wraz z tymi połączeniami i rozumować indukcyjnie dla ciągu o jeden wierzchołek krótszego. Załóżmy więc, że tak nie jest: niech v będzie wierzchołkiem o największym możliwym stopniu d_j, który nie jest połączony z u w grafie G. Wierzchołek u musi być zatem połączony w G z jakimś innym wierzchołkiem, powiedzmy w, o stopniu d_k, dla $k > j$. Pokażemy, że możemy w G tak pozamieniać krawędzie, żeby wierzchołek u był połączony z v zamiast z w (patrz rysunek 3). Faktycznie, ponieważ $d_j > d_k - 1$, więc w G musi istnieć jakiś wierzchołek (nazwijmy go x) połączony z v i niepołączony z w. Podmieniając w grafie G krawędzie uw i vx na krawędzie uv i wx, otrzymujemy dokładnie to, czego chcieliśmy. Jeśli po wykonaniu tej operacji zbiór sąsiadów u w G wciąż nie jest tej samej postaci co w naszym algorytmie, kontynuujemy tego typu podmiany aż do chwili, kiedy te zbiory sąsiadów będą takie same. Po tym usuwamy z grafu G wierzchołek u wraz z incydentnymi krawędziami i powtarzamy wcześniejsze rozumowanie na tak zmniejszonym grafie. Widać, że na końcu otrzymamy dokładnie taki graf, jaki skonstruowałby nasz algorytm.



Rys. 3. Podmiana krawędzi uw i vx na uv i wx.

Wtręt implementacyjny: Podany algorytm można zaimplementować w czasie kwadratowym ze względu na liczbę wierzchołków grafu, czyli $O(n^2)$. Wystarczy w metodzie zachłannej za każdym razem wybierać wierzchołek o największym stopniu, czyli odpowiadający elementowi d_1. Wówczas obsłużenie tego wierzchołka polega na zmniejszeniu elementów $d_2, \ldots, d_{d_1+1}$ o jeden, usunięciu elementu d_1 z ciągu i poprawieniu uporządkowania ciągu. Wszystko to można zrobić w czasie $O(n)$,



Rys. 4. Pierwszy krok algorytmu zachłannego wykonywany dla ciągu 5, 5, 4, 4, 4, 4, 4, 4, 2, 2, 1.

przy czym najmniej oczywistą fazę – przywracanie porządku nierosnącego – wykonujemy za pomocą co najwyżej jednego odwrócenia fragmentu ciągu, patrz też rysunek 4.

Przy nieco ostrożniejszej implementacji można zapisać ten algorytm w złożoności czasowej proporcjonalnej do sumy elementów ciągu (d_i), czyli w czasie proporcjonalnym do liczby krawędzi konstruowanego grafu. Jak to zrobić?

Jeśli jesteśmy zainteresowani tylko binarną informacją – czy podany ciąg jest ciągiem stopni wierzchołków jakiegoś grafu (czy ciąg jest **graficzny**) – możemy posłużyć się jeszcze prostszym kryterium. Pochodzi ono od Erdősa i Gallaiego (w skrócie EG) i orzeka, że nierosnący ciąg $d_1, \ldots, d_n$ o parzystej sumie jest graficzny wtedy i tylko wtedy, gdy dla każdego $k = 1, 2, \ldots, n-1$ zachodzi nierówność

$$\sum_{i=1}^{k} d_i \leqslant k(k-1) + \sum_{i=k+1}^{n} \min(d_i, k).$$

Aby zrozumieć sens nierówności EG, pokażemy, że stanowią one warunek konieczny na graficzność ciągu (d_i). Ustalmy parametr k. Wówczas po lewej stronie nierówności mamy „zapotrzebowanie na krawędzie" pierwszych k wierzchołków. Owo zapotrzebowanie może zostać zaspokojone przez krawędzie łączące pewne pary tych wierzchołków – takich krawędzi może być co najwyżej $\frac{k(k-1)}{2}$, a każda z nich ma wpływ na stopnie dwóch spośród rozważanych wierzchołków – oraz przez krawędzie łączące pewne z tych wierzchołków z pozostałymi. Wierzchołek o stopniu d_i, $i = k+1, \ldots, n$, dostarcza co najwyżej k takich krawędzi – po jednej do każdego z wierzchołków $d_1, \ldots, d_k$ – a zarazem, oczywiście, co najwyżej d_i krawędzi, skąd otrzymujemy konieczność nierówności EG.

Z kolei dostateczność warunków EG można próbować pokazać, wykorzystując własności algorytmu zachłannego. Przykładowo, polecamy Czytelnikowi pouczające sprawdzenie, że pierwsze dwie nierówności EG są równoważne temu, iż algorytm zachłanny poprawnie przetworzy kolejno wierzchołki o stopniach d_1 i d_2. Pełny dowód dostateczności kryterium EG jest, niestety, bardziej skomplikowany.

Wtręt implementacyjny: Kryterium EG ma także tę przewagę nad algorytmem Havla–Hakimiego, że jego prawdziwość można sprawdzić w czasie liniowym względem liczby wierzchołków grafu. Łatwo zauważyć, że wszystkie sumy prefiksowe i sufiksowe ciągu (d_i):

$$S_i = \sum_{j=1}^{i} d_j, \qquad S'_i = \sum_{j=i}^{n} d_j,$$

można obliczyć w czasie $O(n)$. Wtedy lewa strona k-tej nierówności EG to dokładnie S_k. Z kolei trochę nietypowe sumy występujące po prawej stronie nierówności można łatwo obliczyć, jeśli znamy liczbę elementów ciągu (d_i) większych niż k (oznaczenie: p_k). Wówczas suma po prawej stronie to S'_{k+1}, jeśli $p_k \leqslant k$, natomiast w przeciwnym przypadku jest ona równa $S'_{p_k+1} + k \cdot (p_k - k)$. Wreszcie ciąg (p_i) możemy wyznaczyć w czasie $O(n)$, i to na kilka różnych sposobów.

$$M = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Rys. 5. Macierz M skonstruowana dla ciągu graficznego 4, 3, 3, 2, 2. Ciąg sum w kolumnach tej macierzy to 4, 4, 2, 3, 1.

Warto na koniec wspomnieć o jeszcze jednym kryterium graficzności ciągu. Nie jest ono efektywniejsze ani bardziej praktyczne od kryterium EG, ale za to wygląda bardziej egzotycznie. Załóżmy, że $d_1 \leqslant n-1$, w przeciwnym przypadku mamy natychmiast odpowiedź negatywną. Rozważmy macierz zero-jedynkową M wymiaru $n \times n$, o zerowej przekątnej, w której i-ty wiersz zawiera d_i jedynek dopchniętych do lewej strony – z uwzględnieniem jednak zerowej przekątnej (rys. 5). Niech $c_1, \ldots, c_n$ będzie ciągiem sum w kolumnach macierzy M. Wówczas kryterium Gale'a–Rysera orzeka, że ciąg (d_i) jest graficzny wtedy i tylko wtedy, gdy sumy prefiksowe ciągu (d_i) są zdominowane przez sumy prefiksowe ciągu (c_i), tzn.

$$\sum_{i=1}^{k} d_i \leqslant \sum_{i=1}^{k} c_i \quad \text{dla każdego } k = 1, \ldots, n-1.$$

Wnikliwemu Czytelnikowi pozostawiamy podanie związku między tym kryterium a warunkami EG.

Na koniec kilka pytań do Czytelnika. Jakie warunki musi spełniać ciąg stopni (d_i), żeby dało się z niego odtworzyć graf, który jest spójny? A jakie, żeby odpowiadał stopniom wierzchołków jakiegoś grafu dwudzielnego? No dobrze, to drugie zadanie jest dosyć trudne. Ale jakie warunki musi spełniać para ciągów $(a_i)_{i=1}^{k}$ i $(b_i)_{i=1}^{l}$, tak aby istniał graf dwudzielny, w którym (a_i) odpowiada stopniom wierzchołków z jednej grupy, a (b_i) – wierzchołkom z drugiej grupy?