Ciągi de Bruijna

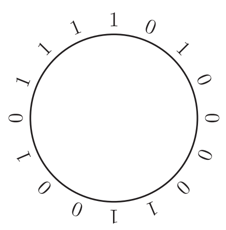
Jakub RADOSZEWSKI

Rozważa się także ciągi de Bruijna złożone z liter $0, 1, \ldots, M$, dla $M > 1$. My nie będziemy się nimi zajmować, jednakże większość tego artykułu można łatwo uogólnić do ciągów tej postaci.

Ciągiem de Bruijna rzędu n (oznaczenie: B_n) nazywamy słowo cykliczne długości 2^n złożone z „liter” 0 i 1, w którym każde n-literowe słowo zerojedynkowe występuje jako podsłowo (tzn. spójny fragment) dokładnie raz. Słowo cykliczne to ciąg liter bez wyróżnionego początku i końca – aby przekształcić zwykłe słowo w cykliczne, wystarczy je sobie wyobrazić jako zapisane wokół okręgu. Na przykład ciągiem de Bruijna rzędu 4 jest

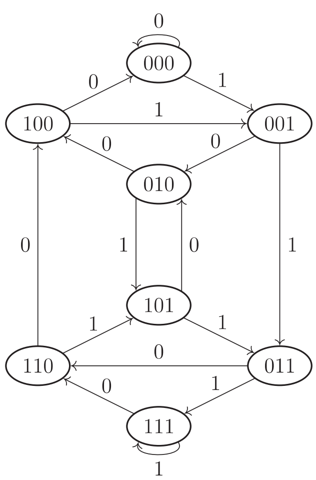
$$\texttt{1010000110010111},$$

który to ciąg można zapisać w postaci cyklicznej jak na rysunku 1.



Rys. 1

Ciągi de Bruijna można wykorzystywać do efektywnego generowania wszystkich słów binarnych długości n, które z kolei można utożsamiać np. z podzbiorami zbioru n-elementowego lub z zapisami dwójkowymi liczb naturalnych od 0 do $2^n - 1$. Są one ciekawe także same w sobie – na pierwszy rzut oka w ogóle nie widać, czy istnieją ciągi de Bruijna wszystkich możliwych rzędów oraz w jaki sposób można by się zabrać za ich konstruowanie. W tym artykule spróbujemy więc rzucić trochę światła na ich strukturę.



Rys. 2. Graf de Bruijna G_4.

Pokażemy najpierw, że ciągi te są mocno związane z tzw. grafami de Bruijna, G_n. Graf G_n ma 2^{n-1} wierzchołków, odpowiadających wszystkim zerojedynkowym słowom długości $n-1$. Z wierzchołka v odpowiadającego słowu $a_1a_2\ldots a_{n-1}$ wychodzą dokładnie dwie krawędzie skierowane:

$$a_1a_2\ldots a_{n-1} \xrightarrow{0} a_2\ldots a_{n-1}0 \quad \text{oraz} \quad a_1a_2\ldots a_{n-1} \xrightarrow{1} a_2\ldots a_{n-1}1,$$

etykietowane odpowiednio literami 0 i 1.

G_n zawiera łącznie 2^n krawędzi, czyli dokładnie tyle, ile liter ma ciąg de Bruijna rzędu n. Pokażemy, że nie jest to tylko przypadkowy zbieg okoliczności – otóż ciąg etykiet kolejnych krawędzi na dowolnym cyklu Eulera w G_n odpowiada pewnemu ciągowi de Bruijna rzędu n.

Cykl Eulera to cykl przechodzący przez każdą krawędź grafu dokładnie raz. Więcej o cyklach Eulera można poczytać np. w książce R. J. Wilsona „Wprowadzenie do teorii grafów”.

Na początku zastanówmy się, dlaczego w ogóle G_n zawiera jakikolwiek cykl Eulera. Dla grafów skierowanych jest na to proste kryterium: z każdego wierzchołka musi wychodzić dokładnie tyle krawędzi, ile do niego wchodzi, oraz graf musi być silnie spójny (tzn. z dowolnego wierzchołka musi się dać dojść – za pomocą krawędzi – do dowolnego innego wierzchołka). Sprawdzenie, że każdy graf G_n spełnia to kryterium, pozostawiamy Czytelnikowi jako ćwiczenie. Mała podpowiedź: aby pokazać, że druga część kryterium jest spełniona, wystarczy umieć explicite wskazać ścieżkę z dowolnego wierzchołka do dowolnego innego.

Niech więc $e = e_1, \ldots, e_{2^n}$ będzie ciągiem etykiet kolejno odwiedzanych krawędzi na dowolnym cyklu Eulera w G_n. Aby uzasadnić, że e jest ciągiem de Bruijna, pokażemy, że dowolne dwa podsłowa długości n (cyklicznej wersji) słowa e są różne (dlaczego wystarczy to uzasadnić?). Niech $e_ie_{i+1}\ldots e_{i+n-1}$ będzie dowolnym takim podsłowem. Zauważmy, że przejście z dowolnego wierzchołka G_n kolejno krawędziami o etykietach $e_i, e_{i+1}, \ldots, e_{i+n-2}$ prowadzi zawsze do wierzchołka v odpowiadającego właśnie słowu $e_ie_{i+1}\ldots e_{i+n-2}$. Z tego względu wystąpienie $e_ie_{i+1}\ldots e_{i+n-2}e_{i+n-1}$ na cyklu jednoznacznie wyznacza krawędź odpowiadającą e_{i+n-1} jako jedyną wychodzącą z v o takiej etykiecie. Gdyby więc pewne dwa podsłowa $e_ie_{i+1}\ldots e_{i+n-1}$ oraz $e_je_{j+1}\ldots e_{j+n-1}$ słowa e były takie same (dla $i \neq j$), to dokładnie te same krawędzie odpowiadałyby e_{i+n-1} oraz e_{j+n-1}, co nie jest możliwe, gdyż e jest cyklem Eulera.

Okazuje się, że każdy cykl Hamiltona w G_n (czyli cykl przechodzący przez każdy wierzchołek dokładnie raz) odpowiada pewnemu ciągowi de Bruijna rzędu $n-1$ (czy potrafisz to uzasadnić?). Ponieważ nie są znane efektywne metody wyznaczania cyklu Hamiltona (ten problem jest NP-zupełny), więc to spostrzeżenie jest interesujące jedynie z teoretycznego punktu widzenia.

Ze związku między cyklami Eulera a ciągami de Bruijna wynika, między innymi, to, że istnieją ciągi de Bruijna wszystkich możliwych rzędów. Ponieważ znane są efektywne algorytmy wyznaczania cyklu Eulera, to otrzymaliśmy w ten sposób także metodę konstrukcji ciągów de Bruijna. Można pokazać, że za jej pomocą da się wygenerować dowolny możliwy ciąg de Bruijna, czyli że każdy ciąg de Bruijna odpowiada pewnemu cyklowi Eulera w G_n. (Czy potrafisz wskazać cykl Eulera w G_4 z rysunku 2 odpowiadający ciągowi de Bruijna z rysunku 1?)

Rozwiązanie zadania M 1235.
Wystarczy wykazać, że suma pól czworokątów $AKSN$ i $SLCM$ jest równa połowie pola czworokąta $ABCD$.

Oznaczmy przez $[\mathcal{F}]$ pole figury $\mathcal{F}$. Czworokąt $KLMN$ jest równoległobokiem, gdyż na mocy twierdzenia odwrotnego do twierdzenia Talesa mamy $KL \parallel AC$, $MN \parallel AC$ oraz $KL = \frac{1}{2}AC$, $MN = \frac{1}{2}AC$. Stąd wynika, że $[AKN] = \frac{1}{4}[ABD]$ oraz $[LCM] = \frac{1}{4}[BCD]$. Dodając stronami dwie ostatnie zależności, otrzymujemy

(1) $\quad [AKN] + [LCM] = \frac{1}{4}[ABCD].$

Analogicznie dowodzimy, że $[BLK] + [MDN] = \frac{1}{4}[ABCD]$, co w połączeniu z zależnością (1) daje $[KLMN] = \frac{1}{2}[ABCD]$.

Ponieważ czworokąt $KLMN$ jest równoległobokiem, więc

(2) $\quad [KSN] + [LMS] = \frac{1}{2}[KLMN] = \frac{1}{4}[ABCD].$

Dodając stronami równości (1) i (2), otrzymujemy tezę.

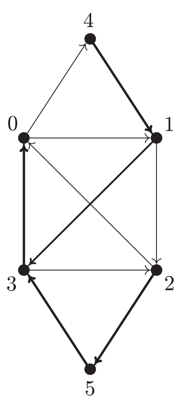
Mogłoby się wydawać, że problem mamy już całkowicie rozwiązany. Niestety, podany algorytm generowania ciągów de Bruijna jest stosunkowo skomplikowany jak na to, że ciągi te z założenia mają pozwalać na *szybkie* generowanie n-literowych słów zerojedynkowych. Z tego względu opiszemy teraz pewien pozornie zupełnie inny algorytm konstrukcji tych ciągów, który jest mniej ogólny (pozwala na wyznaczenie zaledwie jednego ciągu de Bruijna o ustalonym rzędzie n), ale za to dużo prostszy w zapisie.

Algorytm Forda
$u := 11\ldots1$; `/*` n `jedynek */`
while (true) **do begin**
  $v := u_2u_3\ldots u_n$;
  if (było$[v0]$) **then** $u := v1$;
  else $u := v0$;
  if (było$[u]$) **then break**;
  było$[u]$:=**true**;
  pisz(u_n);
end;

W algorytmie Forda konstruujemy ciąg literka po literce, wybierając zawsze najmniejszą możliwą literę, tak aby żadne podsłowo długości n się nie powtórzyło. Okazuje się, że ciąg kolejno dokładanych literek tworzy pewien ciąg de Bruijna rzędu n. Na przykład wykonanie algorytmu Forda dla $n = 4$ prowadzi do otrzymania ciągu de Bruijna

`0000100110101111.`

Fakt, że opisany algorytm jest rzeczywiście poprawny, jest dosyć zaskakujący. Zastanawiające wydaje się już chociażby to, dlaczego akurat zaczynamy całe postępowanie od słowa złożonego z samych jedynek, a nie, na przykład, z samych zer – co ciekawe, rozpoczęcie algorytmu od słowa złożonego z samych zer *nie prowadzi* do wygenerowania ciągu de Bruijna, co Czytelnik może łatwo sprawdzić chociażby dla $n = 4$. Dalej, Czytelnicy zaznajomieni ze standardowym algorytmem wyznaczania cyklu Eulera w grafie mogą zauważyć, że algorytm Forda jest w gruncie rzeczy do niego bardzo podobny, lecz różni się od niego w jednym istotnym względzie: w oryginalnym algorytmie na początku wybierany jest jeden dowolny cykl, a następnie do niego doklejane są kolejne, rekurencyjnie wyznaczane cykle, tymczasem w opisanej właśnie metodzie zakładamy, że na pewno „będziemy mieli szczęście”, czyli że już w pierwszym przebiegu skonstruowany cykl będzie na pewno cyklem Eulera. Pokażemy jednak, że opisany algorytm nie jest aż tak „magiczny”, jak by się mogło wydawać.



Rys. 3. Przykładowy sześciowierzchołkowy graf skierowany z zaznaczonym skierowanym drzewem rozpinającym ukorzenionym w $v_0 = 0$.

Uzasadnienie rozpoczniemy od trochę mniej znanego algorytmu wyznaczania cyklu Eulera w dowolnym grafie $G = (V, E)$. Powiemy, że podgraf T grafu G jest jego *skierowanym drzewem rozpinającym* ukorzenionym w v_0, jeżeli:

a) T zawiera $|V| - 1$ krawędzi;
b) dla każdego wierzchołka $v \in V \setminus \{v_0\}$ w T istnieje dokładnie jedna krawędź, która wychodzi z v;
c) z v_0 nie wychodzi żadna krawędź;
d) z każdego wierzchołka $v \in V \setminus \{v_0\}$ można dojść do v_0, wykorzystując tylko krawędzie z T.

Czy w każdym grafie eulerowskim istnieje skierowane drzewo rozpinające? Jak takie drzewo wyznaczyć?

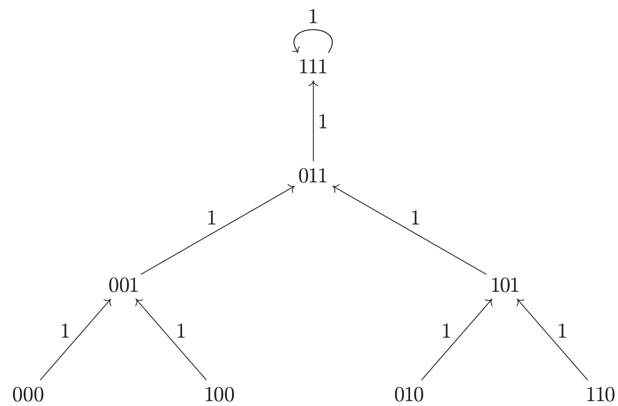
Okazuje się, że na podstawie dowolnego skierowanego drzewa rozpinającego T grafu eulerowskiego można skonstruować cykl Eulera w G. Budowę cyklu rozpoczynamy w v_0. Następnie wędrujemy po kolejnych (parami incydentnych oczywiście) krawędziach grafu, traktując krawędzie z T jako „zakazane”, tzn. wybierając je najpóźniej, jak tylko się da. Dla grafu z rysunku 3 można w ten sposób otrzymać np. cykl

$$0 \to 4 \to 1 \to 2 \to 0 \to 1 \to 3 \to 2 \to 5 \to 3 \to 0.$$

Dlaczego to działa? G jest eulerowski, więc z każdego wierzchołka wychodzi dokładnie tyle krawędzi, ile do niego wchodzi. Stąd, po wejściu do dowolnego wierzchołka $v \neq v_0$ zawsze pozostaje jakaś niewykorzystana krawędź wychodząca z v, co pokazuje, że ścieżka S konstruowana zgodnie z opisanym kryterium może się skończyć jedynie w v_0, czyli jest cyklem. Aby udowodnić, że jest to także cykl Eulera, wystarczy pokazać, że wszystkie krawędzie z T leżą na S (dlaczego?). Aby S mogła skończyć się w v_0, wszystkie krawędzie wychodzące z v_0 muszą leżeć na S, co na mocy własności grafu G oznacza, że również wszystkie krawędzie wchodzące do v_0 muszą leżeć na S. W szczególności dotyczy to wszystkich krawędzi $(w, v_0) \in T$ (np. krawędzi $3 \to 0$ dla grafu z rysunku 3). Dalej, każda krawędź $(w, v_0) \in T$ zostaje wykorzystana w algorytmie jako ostatnia wychodząca z w. To oznacza, że przed tym, jak taka krawędź znajdzie się na S, wszystkie krawędzie wchodzące do w musiały już się na tej ścieżce znaleźć, w tym wszystkie krawędzie postaci $(u, w) \in T$ (na rysunku 3: $1 \to 3$ i $5 \to 3$) itd. Ze względu na warunek d) z definicji skierowanego drzewa rozpinającego kontynuując to rozumowanie, w końcu dojdziemy do liści grafu T, czyli do wierzchołków, do których nie wchodzi już żadna krawędź z T. To kończy uzasadnienie tego, że rzeczywiście S zawiera wszystkie krawędzie z T.

Można pokazać, że cykle Eulera w grafie skierowanym odpowiadają wzajemnie jednoznacznie skierowanym drzewom rozpinającym. Na podstawie tego faktu można wyprowadzić wzór na liczbę cykli Eulera w danym grafie, z którego wynika m.in. wzór na liczbę ciągów de Bruijna rzędu n: $2^{2^{n-1}}$.

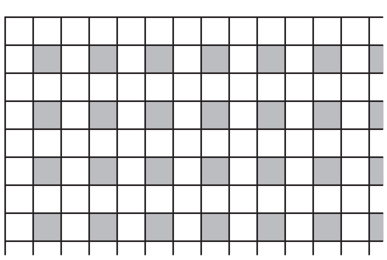
Opisany algorytm jest może ciekawy sam w sobie, ale jaki ma on związek z generowaniem ciągów de Bruijna, a w szczególności z algorytmem Forda? Rozważmy podgraf H grafu de Bruijna G_n, złożony z krawędzi o etykiecie 1. Jak łatwo zauważyć, jest to drzewo skierowane z dołączoną pętlą w korzeniu: na najniższym poziomie drzewa mamy wierzchołki odpowiadające słowom zakończonym zerem, dalej – wierzchołki zakończone jedną jedynką, potem dwiema, itd. aż do korzenia odpowiadającego słowu $11\ldots1$.



Rys. 4. Struktura podgrafu H grafu G_4.

Wyrzućmy pętlę z H, a resztę potraktujmy jako skierowane drzewo rozpinające G_n, ukorzenione w $11\ldots1$. Zauważmy, że poszukiwanie cyklu Eulera w oparciu o to drzewo odpowiada każdorazowemu wyborowi niewykorzystanej krawędzi o najmniejszej możliwej etykiecie. Ponieważ, jak już wcześniej zauważyliśmy, n-literowe słowa binarne jednoznacznie odpowiadają krawędziom grafu G_n (a dokładniej parom postaci: (wierzchołek, etykieta krawędzi wychodzącej z wierzchołka)), więc otrzymany algorytm jest tak naprawdę... dokładnie algorytmem Forda!

Na koniec dodajmy, że algorytm Forda ma jeszcze jedną zaletę: można zauważyć, że generowany w nim ciąg jest najmniejszym alfabetycznie (tj. słownikowo) spośród wszystkich ciągów de Bruijna danego rzędu.



Rozwiązanie zadania M 1234.
Podzielmy dany prostokąt na kwadraty 1×1 (które będziemy nazywać polami) oraz pokolorujmy niektóre z nich tak, jak pokazano na rysunku. Przyjmijmy, że liczba pomalowanych pól wynosi p.

Każdy kwadrat 2×2 zawiera dokładnie jedno pokolorowane pole, a każdy prostokąt 1×4 pokrywa parzystą liczbę pokolorowanych pól (zero lub dwa). Stąd wynika, że liczby k oraz p są tej samej parzystości.

Gdyby żądane pokrycie było możliwe, to rozumując analogicznie, doszlibyśmy do wniosku, że liczby $k-1$ oraz p są tej samej parzystości. Uzyskaliśmy sprzeczność.