Ćwiczenia 12

6–kolorowanie listy

Dane: Lista pokolorowana n kolorami
Wynik: Lista pokolorowana 6 kolorami

Tak długo, jak lista zawiera kolory o numerach ≥ 6, powtarzaj następującą procedurę:

- każdy element analizuje swój kolor i kolor swojego sąsiada, powiedzmy $c = b_l b_{l-1} \dots b_2 b_1 b_0$ i $c' = b'_l b'_{l-1} \dots b'_2 b'_1 b'_0$.
- niech k indeks najmniej znaczącego bitu, t. że $b_k \neq b'_k$,
- nowym kolorem elementu staje się $2k + b_k$,
- ostatni element listy (o kolorze c), zostaje pokolorowany kolorem $c \; mod \; 2$.

W pojedynczym kroku liczba używanych kolorów redukowana jest z n do $2\lceil \log n \rceil$.

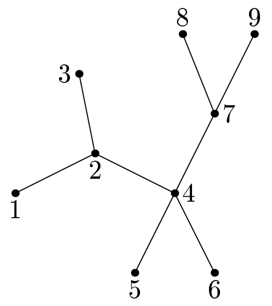
Przykład:

```
(10, 8, 1, 6,14, 5, 3,11)
( 3, 0, 1, 6, 0, 2, 6, 1)
( 1, 0, 1, 3, 2, 4, 0, 1)
```

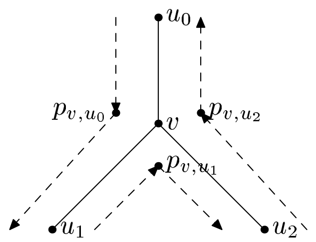
Cykl Eulera dla drzewa dowolnego stopnia

Dane: drzewo dowolnego stopnia dane w postaci grafu (lista sąsiedztwa)
Wynik: cykl Eulera, ustalenie wskaźnika parent dla każdego wierzchołka drzewa

v	$\{u_i\} = adj(v)$
1	2
2	1, 3, 4
3	2
4	2, 5, 6, 7
5	4
6	4
7	8, 4, 9
8	7
9	7



Każdemu wierzchołkowi v przypisano d procesorów (gdzie d — stopień wierzchołka), należy teraz tak ustalić wskaźniki $next$ procesorom (przerywane linie na rysunku), by wszystkie zostały połączone w cykl.



Można tu zastosować następującą metodę:

- niech wierzchołek v ma następujących sąsiadów: $adj(v) = u_0, u_1, u_2, \ldots, u_{d-1}$, i zostały mu przypisane procesory: $p_{v,u_0}, p_{v,u_1}, \ldots, p_{v,u_{d-1}}$.
- dla $0 \leq i < d$, $NEXT(p_{u_i,v}) = p_{v,(i+1)\ mod\ d}$.

Dla drzewa z przykładu cykl ma postać:

$$p_{4,5} \to p_{5,4} \to p_{4,6} \to p_{6,4} \to p_{4,7} \to p_{7,9} \to p_{9,7} \to p_{7,8} \to p_{8,7} \to p_{7,4} \to p_{4,2} \to$$
$$p_{2,1} \to p_{1,2} \to p_{2,3} \to p_{3,2} \to p_{2,4} \to cykl$$

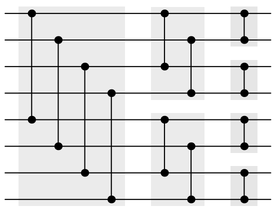
Ukorzenienie drzewa:

- dla wybranego jako korzeń wierzchołka i, należy przerwać cykl przed dowolnym procesorem $p_{i,j}$,
- każdemy procesorowi przypisywana jest wartość $+1$,
- obliczenie sum prefiksowych na otrzymanych listach,
- dla każdego węzła v (o stopniu większym od 1), t.że: $adj(v) = u_0, u_1, u_2, \ldots, u_{d-1}$, z obliczonymi sumami prefiksowymi $s_{v,u_0}, s_{v,u_1}, \ldots, s_{v,u_{d-1}}$, jako ojca wierzchołka v należy wybrać u_i, takie, że $s_{v,i} < s_{v,(i+1)\ mod\ d}$.

Przykład:
jeśli dla przykładowego drzewa wybierze się węzeł 4, a cykl przerwie przed procesorem $p_{4,5}$, to dla wierzchołka 2 ($adj(2) = 1, 3, 4$), otrzymane zostaną sumy: $s_{2,1} = 5$, $s_{2,3} = 3$, $s_{2,4} = 1$, tak więc ojcem węzła 2 zostanie $u_3 = 4$.

Bitoniczna sieć sortująca

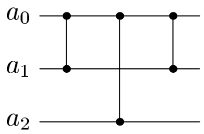
Dane: ciąg bitoniczny
Wynik: posortowany ciąg



Zasada zero–jedynkowa

Jeśli sieć porównująca o n wejściach poprawnie sortuje wszystkie 2^n ciągi zer i jedynek, to sortuje poprawnie dowolne ciągi liczb.

Dla przykładowej sieci:



Wyniki dla ciągów zero–jedynkowych (a_0, a_1, a_2):

```
000 - OK        001 - OK
010 - Błąd      011 - OK
100 - Błąd      101 - OK
110 - OK        111 - OK
```

System kryptograficzny RSA

- wybierz losowe liczby pierwsze p, q,
- $n = pq$,
- wybierz nieparzystą liczbę e względnie pierwszą z $(p-1)(q-1)$,
- znajdź d, takie, że $ed \equiv 1 \pmod{(p-1)(q-1)}$,
- klucz publiczny: $P = (e, n)$,
- klucz prywatny: $S = (d, n)$,
- szyfrowanie: $P(M) = M^e \pmod{n}, \quad (0 < M < n)$,
- odszyfrowanie: $S(M) = M^d \pmod{n}$,

Przykład:
Niech $p = 11$, $q = 17$, $n = 187$, $e = 3$, $d = 107$, $m = 100$.
$P(100) = 100^3\ mod\ 187 = 111$.
$S(100) = 111^{107}\ mod\ 187 = 100$.

Test Millera–Rabina

Dane: n — testowana liczba, a — liczba losowa z $(0 < a < n)$
Wynik: 0 — n nie jest pierwsza, 1 — raczej jest

```
niech (b_k, b_{k-1}, ..., b_2, b_1, b_0) binarna reprezentacja n - 1
d:=1;
for i:=k downto 0 do
    x:=d;
    d:=d * d mod n;
    if (d = 1) and (abs(x) ≠ 1) then
        return 0 // n nie jest pierwsza
    if (b_i = 1) then
        d:=d · a mod n;
if (d ≠ 1) then
    return 0 // test Fermata
return 1
```

Twierdzenie:
Jeśli n jest nieparzystą liczbą pierwszą, to równanie $x^2 \equiv 1 \pmod{n}$, ma dokładnie dwa rozwiązania: $x = 1$ i $x = -1$.

Algorytm Ziva–Lempela

```
A[0]=ϵ;
n:=1;
i:=1;
while (i<=length(T)) do
  niech k najdłuższe słowo z A[0..n − 1] takie, że A[k]
    jest prefiksem słowa T[i...]
  l:=length(A[k]);
  c:=T[i+l];
  A[n]:=A[k]+c;
  n:=n+1;
  zapisz (k,c)
  i:=i+l+1;
```

Przykład:

S=`0 00 1 11 10 101 1010 00`

zostanie zakodowane jako:

$(0,0),(1,0),(0,1),(3,1),(3,0),(5,1),(6,0),(1,0)$

Algorytm jest używany m.in w formatach: GIF (LZ78), ZIP, ARJ, LHA, gzip (LZ77).