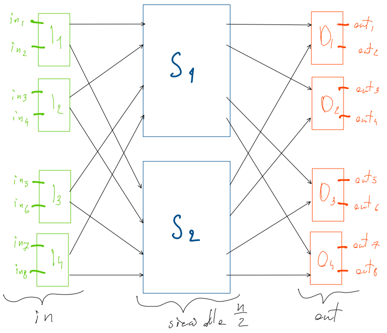
AiSD Lista 3-4

środa, 28 kwietnia 2021 00:23

9. (2pkt) Jakie jest prawdopodobieństwo wygenerowania permutacji identycznościowej przez sieć Beneša-Waksmana, w której przełączniki ustawiane są losowo i niezależnie od siebie w jeden z dwóch stanów (każdy stan przełącznika jest osiągany z prawdopodobieństwem 1/2).

Zobaczmy, jak wygląda taka sieć.



Chcemy uzyskać permutację identycznościową, zatem każda dana na wejściu in_k musi opuścić układ wyjściem out_k.
Załóżmy, że sygnał wejdzie do układu górnym wejściem i opuści go dolnym wyjściem. Jeśli trafi do S_2, to do out_k może dotrzeć tylko poprzez jego dolne wejście. Wtedy out_k musi mieć ten sam układ, co in_k, tzn. na wprost lub na krzyż.
Możemy wnioskować, że stany odpowiadających sobie przełączników in i out muszą być takie same.

Jeśli sygnał trafi do S_1 lub S_2 przez i-te wejście, to przybył z i-tego przełącznika wejściowego. W takim razie, żeby wyszedł na i-tym przełączniku wyjściowym, musi wyjść i-tym wyjściem. W takim razie podsieci muszą tworzyć sieci identycznościowe.

Wyznaczmy prawdopodobieństwo

Prawdopodobieństwo, by obie warstwy (in i out) miały ten sam stan wynosi $\frac{1}{2}$ — oba na krzyż lub oba prosto, gdyż mamy 4 możliwe stany. Mamy $\frac{n}{2}$ przełączników w tych warstwach, zatem łącznie mamy prawdopodobieństwo $\left(\frac{1}{2}\right)^{\frac{n}{2}}$. Dodatkowo pamiętamy, że warstwa wewnętrzna też musi być identycznościowa, prawdopodobieństwo tego oznaczmy $P(\frac{n}{2})$. Mamy dwa przełączniki w tej warstwie, więc łącznie $P(\frac{n}{2})^2$.
Wtedy dla $n = 2^k$ mamy:

$$P(n) = \begin{cases} \frac{1}{2}, & \text{dla } n=2 \\ P(\frac{n}{2})^2 \cdot \left(\frac{1}{2}\right)^{\frac{n}{2}}, & \text{w.p.p.} \end{cases}$$

$$P(n) = P(\tfrac{n}{2})^2 \left(\tfrac{1}{2}\right)^{\frac{n}{2}} = \left(P(\tfrac{n}{4})^2 \cdot \left(\tfrac{1}{2}\right)^{\frac{n}{4}}\right)^2 \left(\tfrac{1}{2}\right)^{\frac{n}{2}} = P(\tfrac{n}{4})^4 \left(\tfrac{1}{2}\right)^{\frac{n}{2}} \cdot \left(\tfrac{1}{2}\right)^{\frac{n}{2}} =$$

$$= \left(P(\tfrac{n}{8})^2 \left(\tfrac{1}{2}\right)^{\frac{n}{8}}\right)^4 \left(\tfrac{1}{2}\right)^{\frac{n}{2}} \cdot \left(\tfrac{1}{2}\right)^{\frac{n}{2}} = P(\tfrac{n}{8})^2 \left(\tfrac{1}{2}\right)^{\frac{n}{2}} \cdot \left(\tfrac{1}{2}\right)^{\frac{n}{2}} \cdot \left(\tfrac{1}{2}\right)^{\frac{n}{2}} = \ldots =$$

$$= P(2)^{\frac{n}{2}} \cdot \underbrace{\left(\tfrac{1}{2}\right)^{\frac{n}{2}} \cdot \ldots \cdot \left(\tfrac{1}{2}\right)^{\frac{n}{2}}}_{\text{łącznie } \log(n-1) \text{ wyrazów}} = \left(\tfrac{1}{2}\right)^{\frac{n}{2}} \cdot \left(\tfrac{1}{2}\right)^{\log(n-1)\frac{n}{2}} = \left(\tfrac{1}{2}\right)^{\log(n) \cdot \frac{n}{2}}$$

$$P(2) = \tfrac{1}{2}$$

Udowodnijmy ten wzór indukcyjnie.

Teza: $P(2^k) = \left(\frac{1}{2}\right)^{\log 2^k \cdot \frac{2^k}{2}} = \left(\frac{1}{2}\right)^{k \cdot 2^{k-1}}$

Podstawa: $k=1$

$$P(2) = \left(\tfrac{1}{2}\right)^{2^0} = \left(\tfrac{1}{2}\right)^1 = \tfrac{1}{2} \quad \checkmark$$

Krok: Załóżmy, że teza jest spełniona dla k i pokażmy, że wtedy jest spełniona dla $k+1$.

$$P(2^{k+1}) = P\left(\tfrac{2^{k+1}}{2}\right)^2 \cdot \left(\tfrac{1}{2}\right)^{\frac{2^{k+1}}{2}} = \underbrace{P(2^k)}_{\text{z zał. ind.}} \cdot \left(\tfrac{1}{2}\right)^{2^k} = \left(\tfrac{1}{2}\right)^{k \cdot 2^{k-1}} \cdot \left(\tfrac{1}{2}\right)^{2^k} = \underline{\left(\tfrac{1}{2}\right)^{(k+1)2^k}} \quad \blacksquare$$

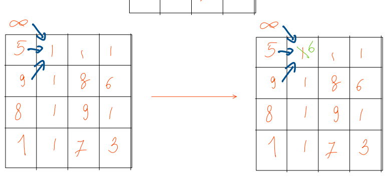
Lista 3

1. (1pkt) Ułóż algorytm znajdujący najtańszą drogę przejścia przez tablicę, w którym oprócz ruchów dopuszczalnych w wersji problemu prezentowanej na wykładzie, dozwolone są także ruchy w górę i w dół tablicy.

Przypomnienie metody z wykładu

∞

5	1	1	1
9	1	8	6
8	1	9	1
1	1	7	3



Każda komórka [illegible]

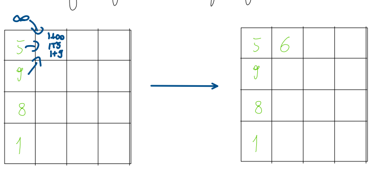
komórka wynikowej tabeli zawiera długość najkrótszej ścieżki, która do niej prowadzi.

Algorytm:

1. Pierwszą kolumnę wypełniamy liczbami z tabeli (są to punkty startowe)

5			
9			
8			
1			

2. Pozostałe komórki wypełniamy licząc sumę wartości komórki z każdym z jej trzech lewych sąsiadów i wybierając z nich najmniejszą.



3. Wypełniamy tak całą tabelę i wynik otrzymujemy w jej skrajnie prawej kolumnie.

2. (1.5pkt) Ułóż algorytm, który dla danego ciągu znajduje długość najdłuższego jego podciągu, który jest palindromem.

3. (1.5pkt) Chcemy obliczać wartości współczynników dwumianowych modulo liczba pierwsza. Ułóż algorytm, który dla danej liczby pierwszej p oraz ciągu par $(n_1, k_1), (n_2, k_2), ...(n_r, k_r)$ obliczy wartości $\binom{n_i}{k_i} \bmod p$. Twój algorytm powinien wypisywać wartości $\binom{n_i}{k_i} \bmod p$ po czasie nie większym niż $O(\max\{n_1, n_2, \ldots, n_i\})$.
Możesz założyć, że $k_i \leq n_i < p$ dla każdego $i = 1, \ldots, r$.

```
N_max <- max(n_1, n_2, ...)
factorials[0] <- 1

for i from 1 to N_max do:
    factorials[i] <- (factorials[i-1] * i) % p
```

```
inverse_p <- [1]
for i from 2 to N_max do:
    Inverse_p <- (-floor(p / i) * inverse_p[p % i]) % p

inverse_factorials[0] <- 1
for i from 1 to N_max do:
    Inverse_factorials[i] <- (inverse_factorials[i-1] * inverse_p[i]) % p
```

```
pairs = [ [n_1, k_2], [n_2, k_2], ... [n_r, k_r]]

for i from 1 to r do:
    n = pairs[i][0]
    k = pairs[i][1]
    print(factorials[n] * inverse_factorials[n - k] * inverse_factorials[k]) % p)
```