582359 Algoritmit ongelmanratkaisussa (kevät 2013)
Viikon 6 ratkaisuja

1. Määritellään matriisit A ja B seuraavasti:

$$A = \begin{pmatrix} 4 & 2 & 4 \\ 1 & 8 & 5 \end{pmatrix}$$

$$B = \begin{pmatrix} 6 & 9 \\ 4 & 2 \\ 1 & 3 \end{pmatrix}$$

Laske matriisitulot AB ja BA.

Ratkaisu:

$$AB = \begin{pmatrix} 4 \cdot 6 + 2 \cdot 4 + 4 \cdot 1 & 4 \cdot 9 + 2 \cdot 2 + 4 \cdot 3 \\ 1 \cdot 6 + 8 \cdot 4 + 5 \cdot 1 & 1 \cdot 9 + 8 \cdot 2 + 5 \cdot 3 \end{pmatrix} = \begin{pmatrix} 36 & 52 \\ 43 & 40 \end{pmatrix}$$

$$BA = \begin{pmatrix} 6 \cdot 4 + 9 \cdot 1 & 6 \cdot 2 + 9 \cdot 8 & 6 \cdot 4 + 9 \cdot 5 \\ 4 \cdot 4 + 2 \cdot 1 & 4 \cdot 2 + 2 \cdot 8 & 4 \cdot 4 + 2 \cdot 5 \\ 1 \cdot 4 + 3 \cdot 1 & 1 \cdot 2 + 3 \cdot 8 & 1 \cdot 4 + 3 \cdot 5 \end{pmatrix} = \begin{pmatrix} 33 & 84 & 69 \\ 18 & 24 & 26 \\ 7 & 26 & 19 \end{pmatrix}$$

2. Määritellään Fibonaccin luvut seuraavasti:

$$F(n) = \begin{cases} 0 & \text{jos } n = 0 \\ 1 & \text{jos } n = 1 \\ F(n-2) + F(n-1) & \text{muuten} \end{cases}$$

Perustele, miksi seuraava yhtälö pätee:

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} F(n) \\ F(n+1) \end{pmatrix}$$

Laske yhtälön ja tehokkaan matriisipotenssilaskun avulla $F(16)$.

Ratkaisu:

Yhtälössä on ideana, että vasemmalla puolella 2×1 -matriisi sisältää peräkkäiset Fibonaccin luvut $F(k)$ ja $F(k+1)$ ja 2×2 -matriisilla kertominen tuottaa uuden 2×1 -matriisin, joka sisältää Fibonaccin luvut $F(k+1)$ ja $F(k+2)$.

Luvun $F(16)$ voi laskea seuraavasti:

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^2 = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}^2 = \begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix}^2 = \begin{pmatrix} 13 & 21 \\ 21 & 34 \end{pmatrix}$$

$$\begin{pmatrix} 13 & 21 \\ 21 & 34 \end{pmatrix}^2 = \begin{pmatrix} 610 & 987 \\ 987 & 1597 \end{pmatrix}$$

Tämän perusteella $F(16) = 987$.

3. Toteuta seuraavat algoritmit Fibonaccin lukujen laskentaan:

   (a) ”perinteinen” for-silmukka

   (b) tehtävässä 2 esitetty menetelmä

   (c) seuraavan kaavan käyttäminen:

$$F_n = \frac{(\frac{1+\sqrt{5}}{2})^n - (\frac{1-\sqrt{5}}{2})^n}{\sqrt{5}}$$

   Vertaile menetelmien tehokkuutta suurilla n:n arvoilla. Mitkä ovat menetelmien aikavaativuudet?

   Ratkaisu:

   Tiedostossa `Fibotesti.java` on testiohjelma, joka sisältää kohtien (a)–(c) algoritmit. Menetelmän (a) aikavaativuus on $O(n)$ ja muiden aikavaativuudet ovat $O(\log n)$ – olettaen, että peruslaskutoimitukset ovat vakioaikaisia. Käytännössä menetelmä (b) vaikuttaa toimivan nopeiten.

   Menetelmän (c) toteutuksessa luvut on esitetty muodossa $a + b\sqrt{5}$, jossa a ja b ovat rationaalilukuja. Tämä mahdollistaa tarkan vastauksen laskemisen.

4. Määritellään funktio $N(n)$ seuraavasti:

$$N(n) = \begin{cases} 0 & \text{jos } n < 0 \\ 1 & \text{jos } n = 0 \\ \sum_{i=1}^{6} N(n-i) & \text{jos } n > 0 \end{cases}$$

   Funktio kertoo, monellako tavalla luvun n voi esittää nopan silmälukujen summana. Esitä funktio $N(n)$ tehtävän 2 tapaisessa matriisimuodossa. Laske sitten luvun $N(10^{100})$ viimeiset 5 numeroa.

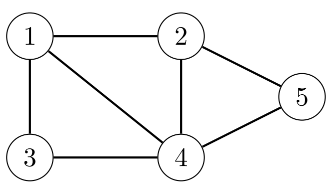
   Ratkaisu:

   Rekursiokaavasta saa seuraavan matriisiyhtälön:

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}^n \begin{pmatrix} 1 \\ 1 \\ 2 \\ 4 \\ 8 \\ 16 \end{pmatrix} = \begin{pmatrix} N(n) \\ N(n+1) \\ N(n+2) \\ N(n+3) \\ N(n+4) \\ N(n+5) \end{pmatrix}$$

   Tiedostossa `NopanVikat.java` on tehokasta matriisipotenssilaskua käyttävä ratkaisu, joka laskee tapauksen $N(10^{100})$ vastauksen 5 viimeistä numeroa.

5. Vierusmatriisi on verkkoa esittävä matriisi, jossa luku 1 tarkoittaa, että kahden solmun välillä on kaari, ja luku 0 tarkoittaa, että kaarta ei ole. Esimerkiksi verkkoa



   vastaa vierusmatriisi

$$V = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

Kun vierusmatriisin korottaa potenssiin n, tuloksena on uusi matriisi, joka kertoo, montako erilaista n kaaren polkua on solmujen välillä. Esimerkiksi seuraava matriisi esittää 3 kaaren polkuja yllä olevassa verkossa:

$$V^3 = \begin{pmatrix} 4 & 7 & 5 & 7 & 3 \\ 7 & 4 & 3 & 7 & 5 \\ 5 & 3 & 2 & 6 & 3 \\ 7 & 7 & 6 & 6 & 6 \\ 3 & 5 & 3 & 6 & 2 \end{pmatrix}$$

Matriisi kertoo esimerkiksi, että solmusta 1 solmuun 3 on 5 polkua, joissa on 3 kaarta. Nämä ovat:

- $1 \to 2 \to 1 \to 3$
- $1 \to 2 \to 4 \to 3$
- $1 \to 3 \to 1 \to 3$
- $1 \to 3 \to 4 \to 3$
- $1 \to 4 \to 1 \to 3$

Todista, että tämä toimii millä tahansa verkolla ja polun pituudella.

Ratkaisu:

Osoitetaan induktiolla, että missä tahansa verkossa V matriisi V^n kertoo n:n pituisten polkujen määrän solmujen välillä. Jos $n = 1$, väite pätee, koska V on verkon vierusmatriisi. Jos väite pätee, kun $n = k$, niin väite pätee myös, kun $n = k + 1$. Syynä tähän on, että mikä tahansa $k + 1$:n pituinen polku saadaan kulkemalla ensin k:n pituista polkua johonkin välisolmuun ja sitten 1:n pituista polkua perille. Matriisien V^k ja V kertolasku käy läpi kaikki mahdolliset polun välisolmut ja laskee lukumäärät yhteen.

6. Suoraviivainen matriisin kertolaskun toteutus kolmella for-silmukalla vie aikaa $O(n^3)$. Tunnettu tehokkaampi algoritmi on Strassenin algoritmi, joka vie aikaa vain $O(n^{2,807})$. Mutta onko Strassenin algoritmi oikeasti tehokkaampi? Toteuta suoraviivainen algoritmi ja Strassenin algoritmi ja tutki niiden tehokkuutta erikokoisilla matriiseilla.

   Ratkaisu:

   Tiedostossa `Matriisit.java` on perusalgoritmin ja Strassenin algoritmin toteutus. Käytännössä perusalgoritmi on yleensä nopeampi, koska sen vakiokertoimet ovat paljon pienemmät. Kuitenkin Strassenin algoritmia on mahdollista parantaa niin, että se siirtyy käyttämään perusalgoritmia pienillä matriiseilla. Tällöin kummankin algoritmin hyvät puolet tulevat esille.