Ulančano množenje matrica

Ime: Mirjana Jukić-Bračulj

Predmet: Oblikovanje i analiza algoritama

Zagreb, 18.1.2016.

Sadržaj:

- Ponavljanje

Sadržaj:

- Ponavljanje
- Opis problema

Sadržaj:

- Ponavljanje
- Opis problema
- Rekurzivno rješenje

Sadržaj:

- Ponavljanje
- Opis problema
- Rekurzivno rješenje
- Dinamičko programiranje

Sadržaj:

- Ponavljanje
- Opis problema
- Rekurzivno rješenje
- Dinamičko programiranje
- Složenost algoritma

Sadržaj:

- Ponavljanje
- Opis problema
- Rekurzivno rješenje
- Dinamičko programiranje
- Složenost algoritma
- Primjeri i zaključak

Ponavljanje

- Množenje matrica je asocijativno

$$A(BC)=(AB)C$$

Ponavljanje

- Množenje matrica je asocijativno

$$A(BC)=(AB)C$$

- Algoritam za množenje matrica dimenzija axb i bxc:

```
for(int i=0; i<a; ++i)
    for(int j=0; j<c; ++j)
        for(int k=0; k<b; ++k)
            c[i][j]+=a[i][k]*b[k][j]
```

Ponavljanje

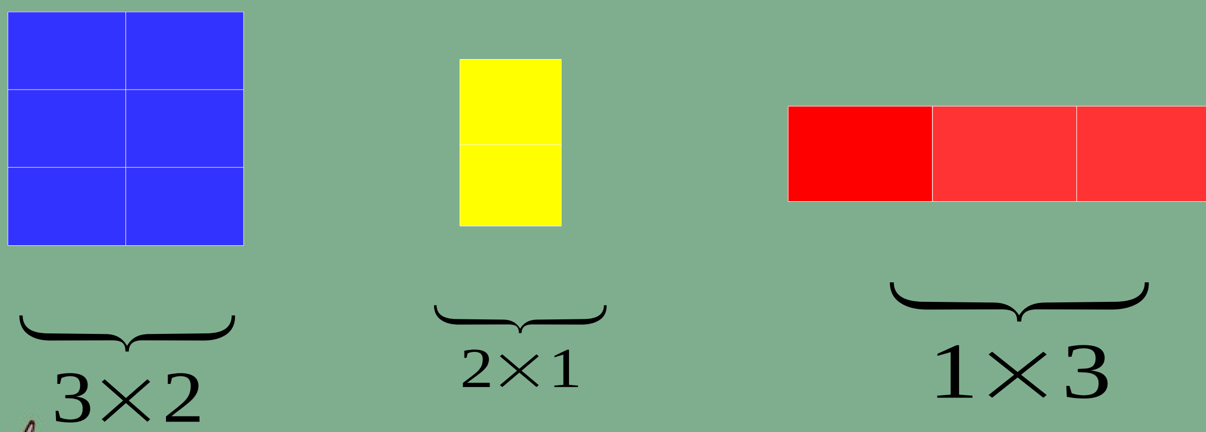
- Za izračunat jedan element umnoška treba nam b množenja.

Ponavljanje

- Za izračunat jedan element umnoška treba nam b množenja.
- Za izračunat njih axc treba nam axbxc množenja.

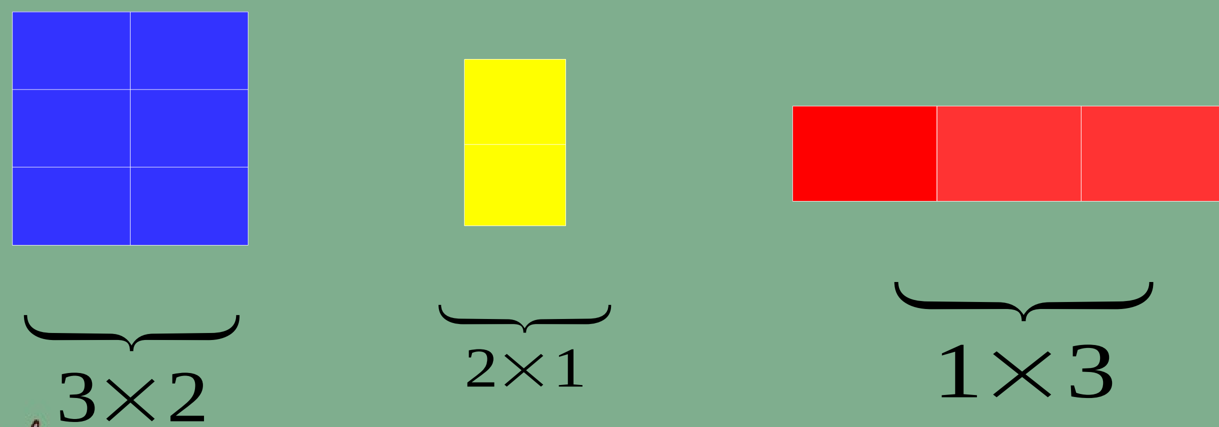
Primjer:

- Uzmimo 3 matrice A, B, C



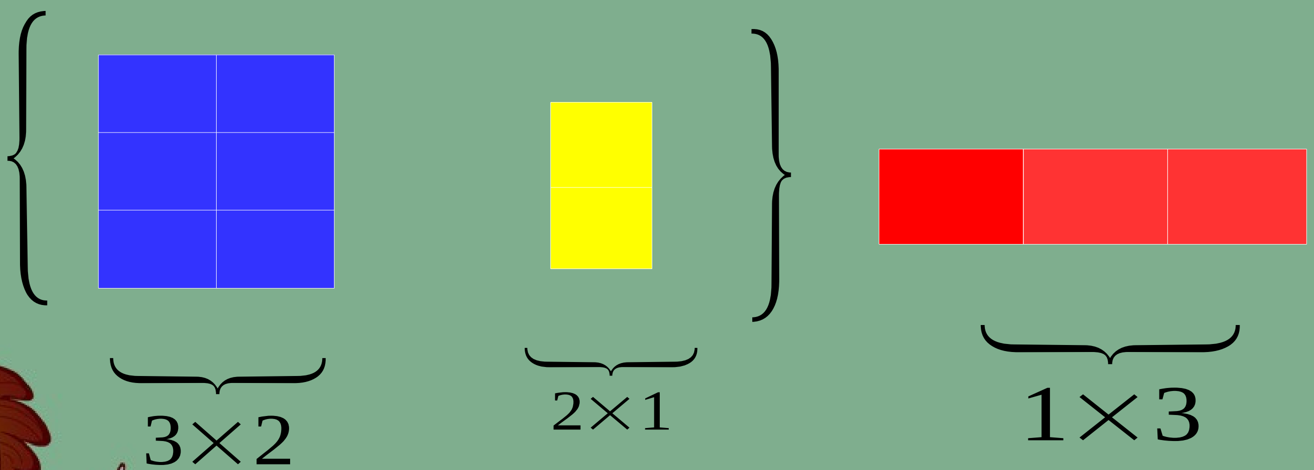
Primjer:

- Uzmimo 3 matrice A, B, C



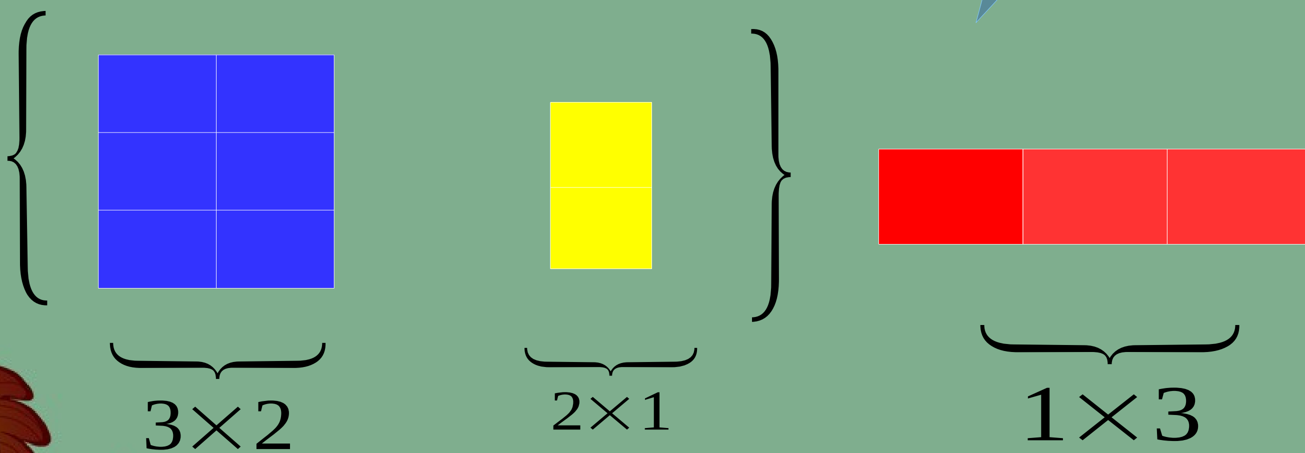
Kojim ćemo redoslijedom množiti matrice?

Prvi način:

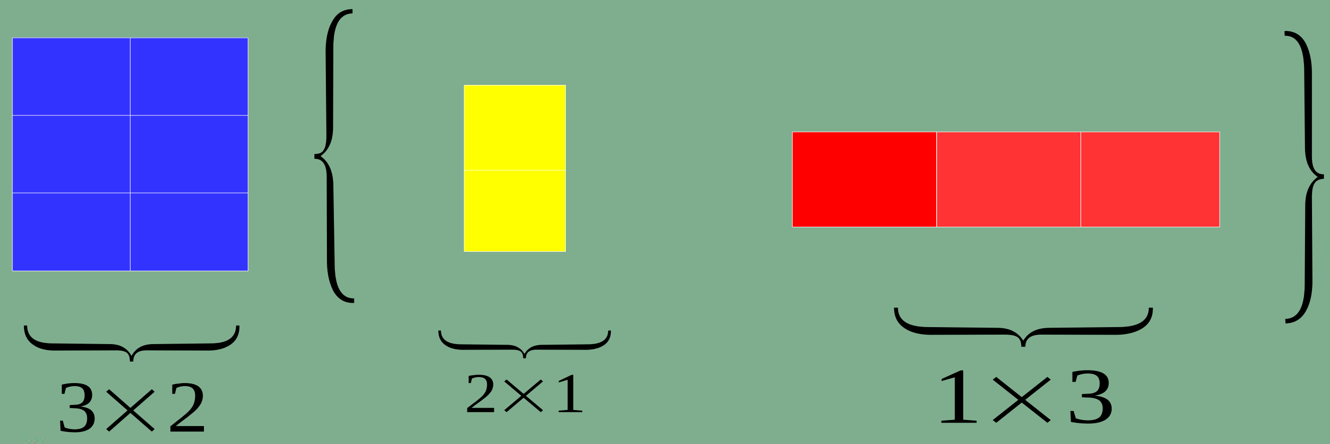


Prvi način:

3*2*1+3*1*3=
6+9=
15

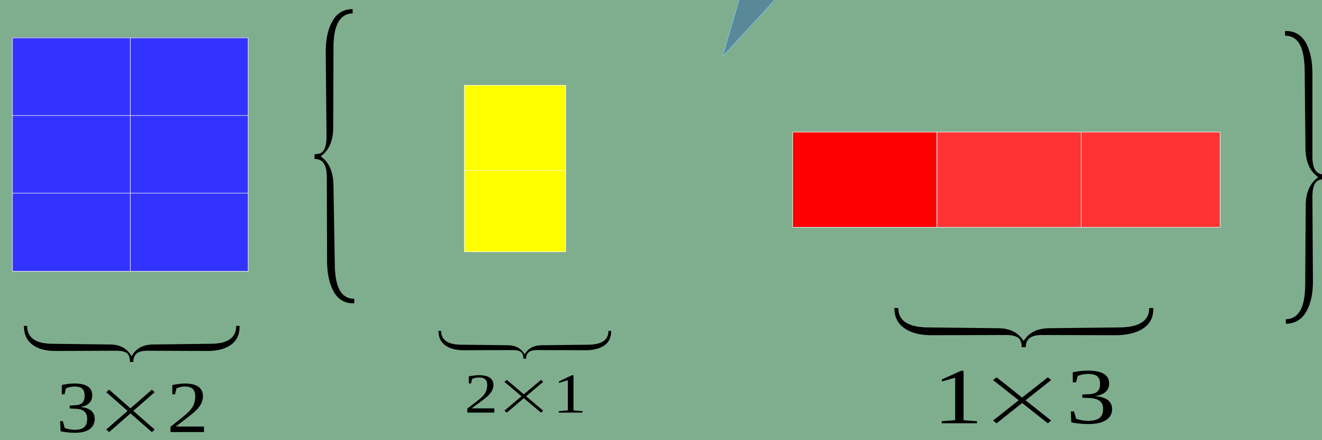


Drugi način:



Drugi način:

2*1*3+3*2*3=
6+18=
24



Opis problema

- Neka je dan niz matrica $A_1,..., A_n$

Opis problema

- Neka je dan niz matrica $A_1,..., A_n$
- Matrica A_i je dimenzija $p_{i-1} x p_i$

Opis problema

- Neka je dan niz matrica $A_1,\ldots, A_n$
- Matrica A_i je dimenzija $p_{i-1} x p_i$
- Treba postaviti zagrade u umnošku $A_1 \ldots A_n$ tako da broj operacija množenja skalara bude minimalan

Opis problema

- Neka je dan niz matrica $A_1, ..., A_{n-1}$
- Matrica A_i je dimenzija $p_{i-1} x p_i$
- Treba postaviti zagrade u umnošku $A_1 ... A_n$ tako da broj operacija množenja skalara bude minimalan

Napomena: cilj nam je samo odrediti poredak kojim množimo matrice, nećemo ih množiti!

Koliko imamo takvih poredaka?

- Označimo s P(k) broj načina na koji možemo u potpunosti postaviti zagrade za k matrica.

Koliko imamo takvih poredaka?

- Označimo s P(k) broj načina na koji možemo u potpunosti postaviti zagrade za k matrica.
- Neka je $k \in \{1, \ldots, n\}$ proizvoljan. Postavimo zagrade na sljedeći način.
- $(A_1 \ldots A_k \ldots A_n)$

Koliko imamo takvih poredaka?

- Označimo s P(k) broj načina na koji možemo u potpunosti postaviti zagrade za k matrica.
- Neka je $k \in \{1, \ldots, n\}$ proizvoljan. Postavimo zagrade na sljedeći način.
- $((A_1 \ldots A_k)(A_{k+1} \ldots A_n))$

Koliko imamo takvih poredaka?

- Za svaki k ukupan broj načina je P(k)P(n-k)

Koliko imamo takvih poredaka?

- Za svaki k ukupan broj načina je P(k)P(n-k)
- Zbog proizvoljnosti od k imamo da P(n)= $\sum_{k=1}^{n} P(k)P(n-k)$

$$P(1)=1$$

Riješenje rekurzije – Catalanovi brojevi

- Vrijedi:
- $$P(n) = \frac{1}{n} \times \binom{2n-2}{n-1} \approx \frac{4^n}{4\sqrt{\pi}\,n^{1.5}}$$

Riješenje rekurzije – Catalanovi brojevi

- Vrijedi:
- $P(n) = \frac{1}{n} \times \binom{2n-2}{n-1} \approx \frac{4^n}{4\sqrt{\pi}\, n^{1.5}}$

- Dakle, $C_n = P(n-1)$ pri čemu su s C_n označeni Catalanovi brojevi

Riješenje rekurzije – Catalanovi brojevi

- Vrijedi:
- $$P(n)=\frac{1}{n}\times\binom{2n-2}{n-1}\approx\frac{4^n}{4\sqrt{\pi}n^{1.5}}$$
- Dakle, C_n=P(n-1) pri čemu su s C_n označeni Catalanovi brojevi

$$C_n=\Omega\left(\frac{4^n}{n^{1.5}}\right)$$

Rješenje rekurzije – Catalanovi brojevi

- Pronalazak broja množenja skalara za pojedini način postavljanja zagrada je složenosti $\Theta(n)$

Rješenje rekurzije – Catalanovi brojevi

- Pronalazak broja množenja skalara za pojedini način postavljanja zagrada je složenosti $\Theta(n)$
- Pa je ukupna složenost pronalaska rješenja

$$\Omega\left(\frac{4^n}{\sqrt{n}}\right)$$

Dinamičko programiranje

- Pretpostavimo da smo za neki k dobili sljedeće:

  $A_1 \ldots A_k = A_{1 \ldots k}$

  $A_{k+1} \ldots A_n = A_{k+1 \ldots n}$

Dinamičko programiranje

- Pretpostavimo da smo za neki k dobili sljedeće:

  $A_1 \ldots A_k = A_{1\ldots k}$

  $A_{k+1} \ldots A_n = A_{k+1\ldots n}$

- Kako odabrati pravi k? Kako postaviti zagrade u umnošcima $A_1 \ldots A_k$ i $A_{k+1} \ldots A_n$?

Dinamičko programiranje

- Rješenja problema čuvamo u tablici

		j		
i		**m[i][j]**		

Minimalan broj množenja za umnožak $A_i \ldots A_j$

m[i][i]=0

Dinamičko programiranje

- Tablicu popunjavamo po sljedećoj rekurziji

$$m[i][j]=\min_{i<=k<j}(m[i][k]+m[k+1][j]+p_{i-1}p_k p_j)$$

Dinamičko programiranje

```
Matrix-Chain(array p[1,..., n]) {
        array s[1,...,n-1, 2,..., n]
        for i=1 to n do m[i][i]=0;
        for L=2 to n do {
            for i=1 to n-L+1 do {
             j=i+L-1
            m[i][j]=INFINITY;
            for k=1 to j-1 do {
                q=m[i][k]+m[k+1][j]+p[i-1]p[k]p[j];
                if(q<m[i][j]) {
                    m[i][j]=q;
                    s[i][j]=k;
      }}}}
    return m[1][n] and s;
}
```

Dinamičko programiranje-ilustracija

- Primjer: Neka su A_1, A_2, A_3, A_4 matrice dimenzija redom 10x30, 30x5, 5x60, 60x70. Brojevi m[i][j] se računaju na sljedeći način:

Dinamičko programiranje-ilustracija

0			
	0		
		0	
			0

Dinamičko programiranje-ilustracija

0	1500		
	0	9000	
		0	21000
			0

Dinamičko programiranje-ilustracija

0	1500	4500	
	0	9000	31500
		0	21000
			0

Dinamičko programiranje-ilustracija

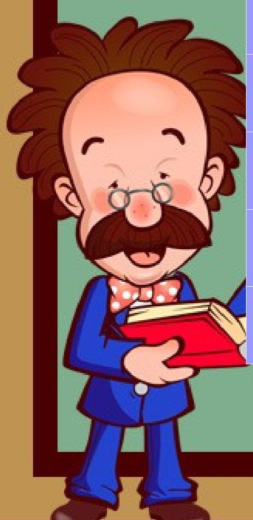
0	1500	4500	26000
	0	9000	31500
		0	21000
			0

Dinamičko programiranje-složenost

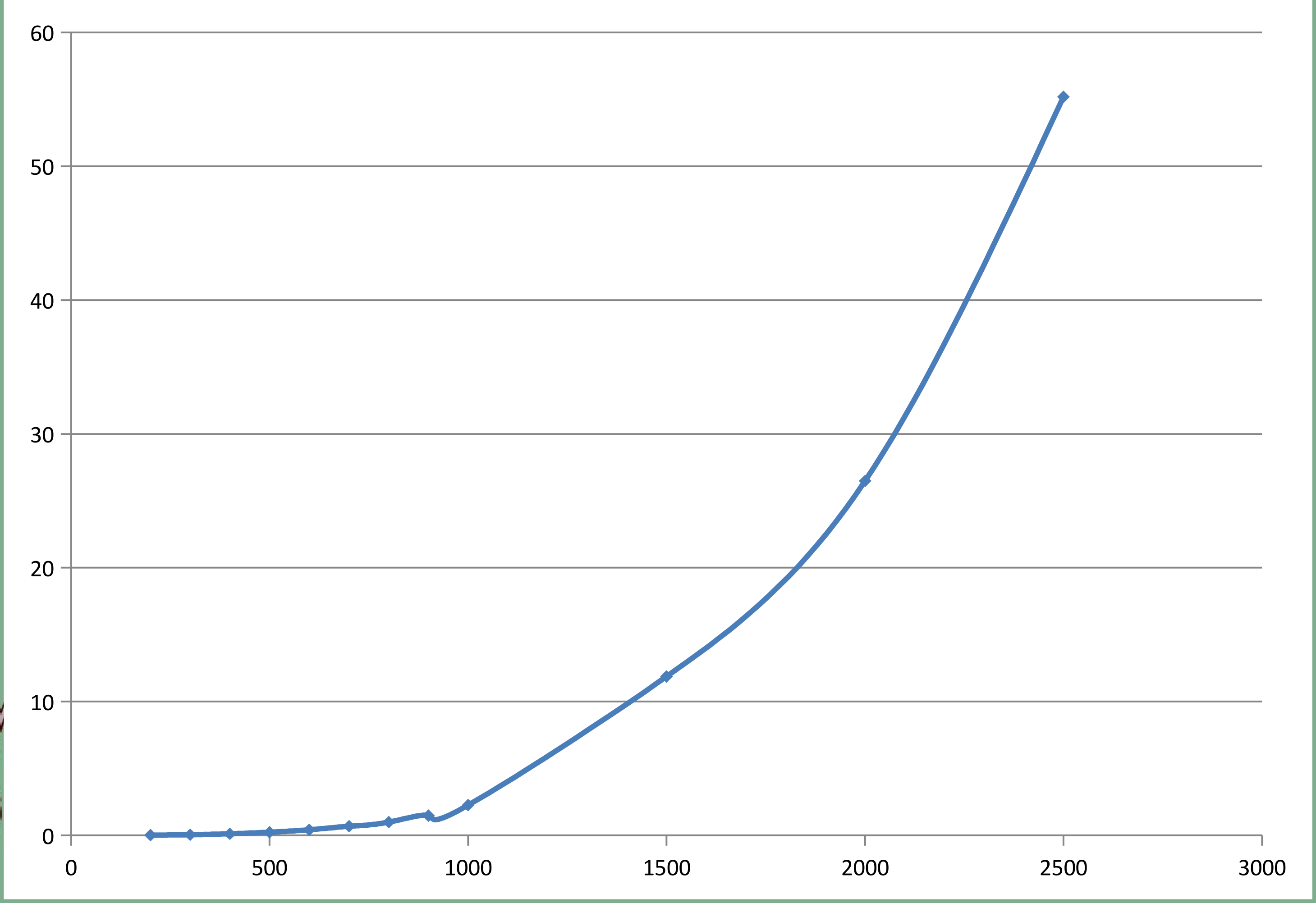
- Prostorna složenost je $O(n^2)$
- Vremenska složenost je $O(n^3)$ zato što imamo tri petlje od po maksimalno n prolazaka

Mjerenja

n	Vrijeme (s)
200	0.02
300	0.05
400	0.12
500	0.24
600	0.42
700	0.68
800	0.99
900	1.47
1000	2.26
1500	11.87
2000	26.48
2500	55.18



Graf



Može li bolje?

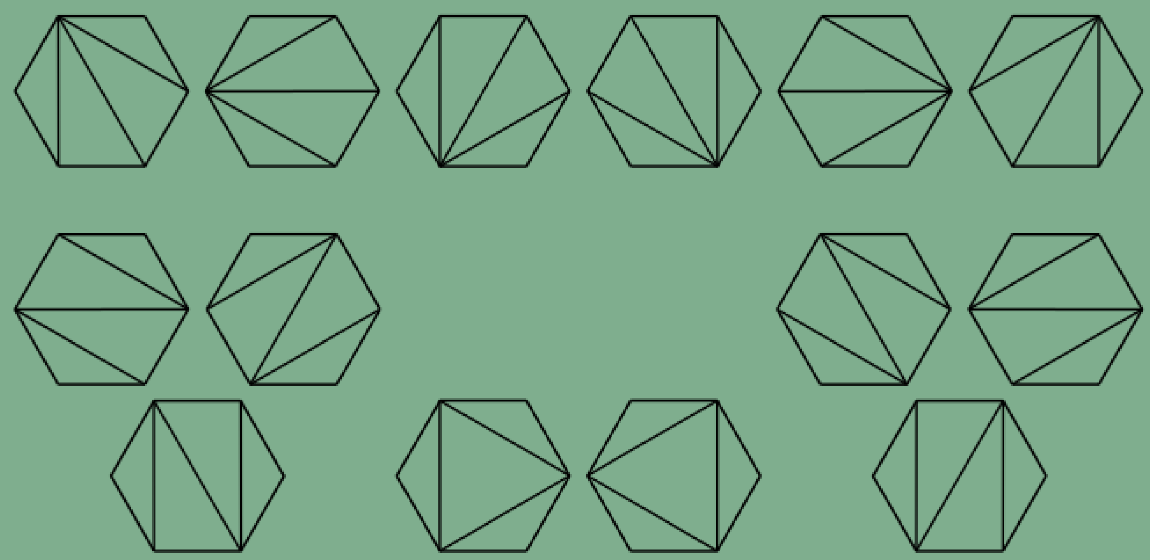
- Postoje algoritmi koji imaju bolju složenost od $O(n^3)$

Može li bolje?

- Postoje algoritmi koji imaju bolju složenost od $O(n^3)$
- Jedan od njih je algoritam kojeg su objavili Hu i Shing, kompleksnosti $O(n\log(n))$

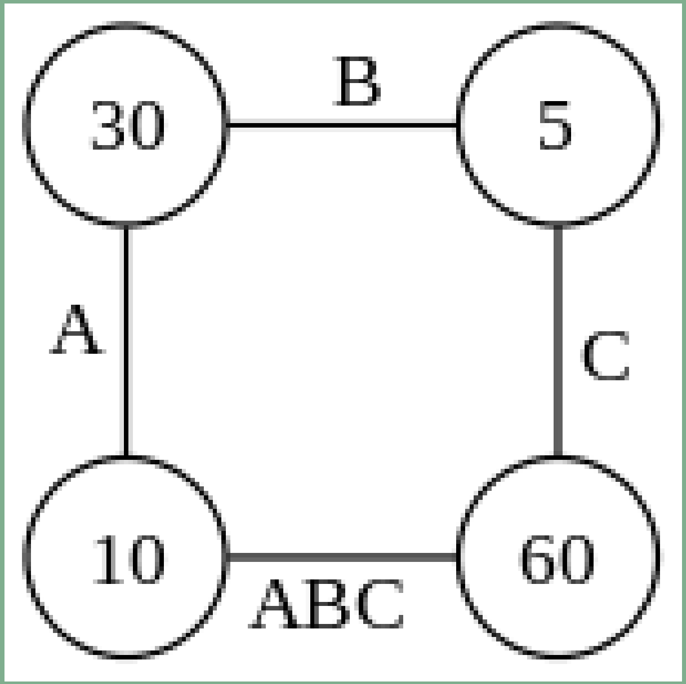
Hu & Shing

- Algoritam povezuje problem ulančanog množenja matrica s problemom podjele pravilnog mnogokuta u trokute.



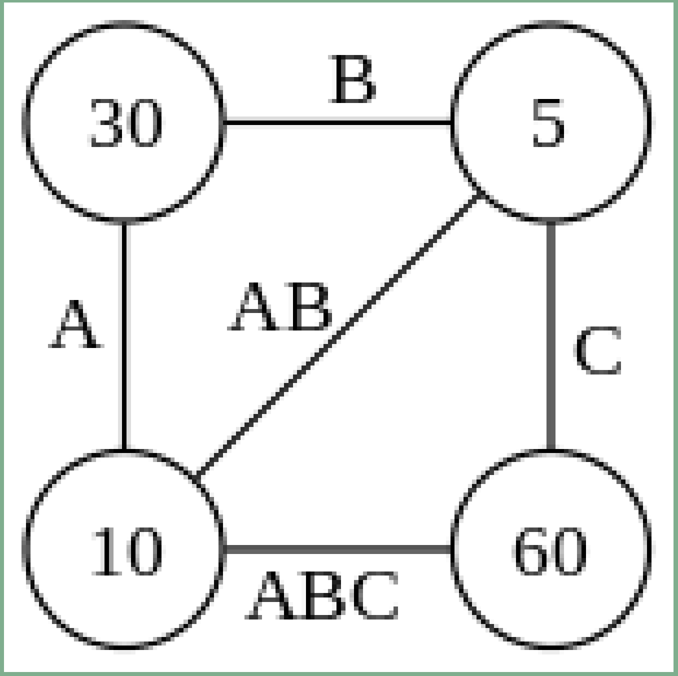
Primjer

- Dane su matrice A, B, C i konačan rezultat ABC dimenzija redom 10x30, 30x5, 5x60 i 10x60



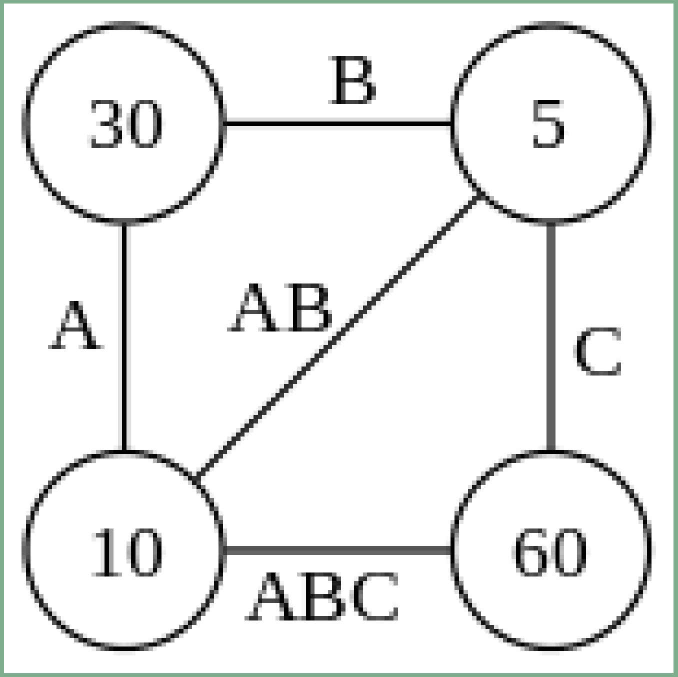
Primjer

- Jedna moguća podjela kvadrata je:



Primjer

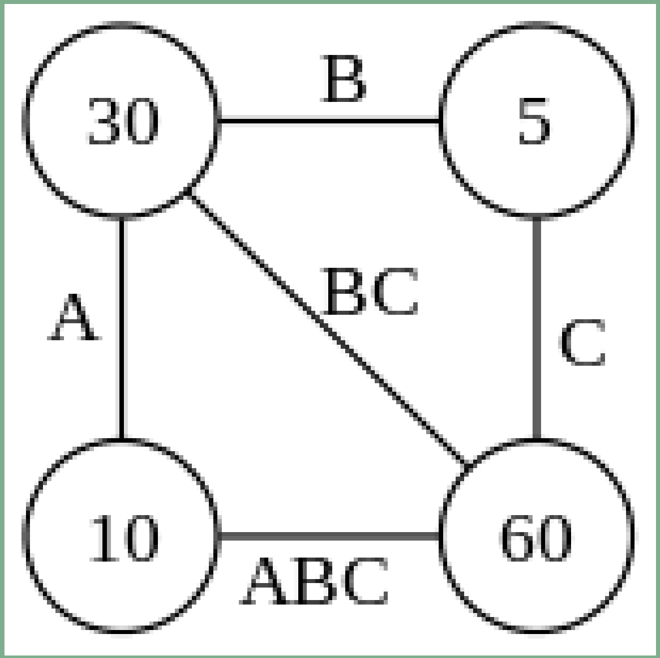
- Jedna moguća podjela kvadrata je:



4500
operacija

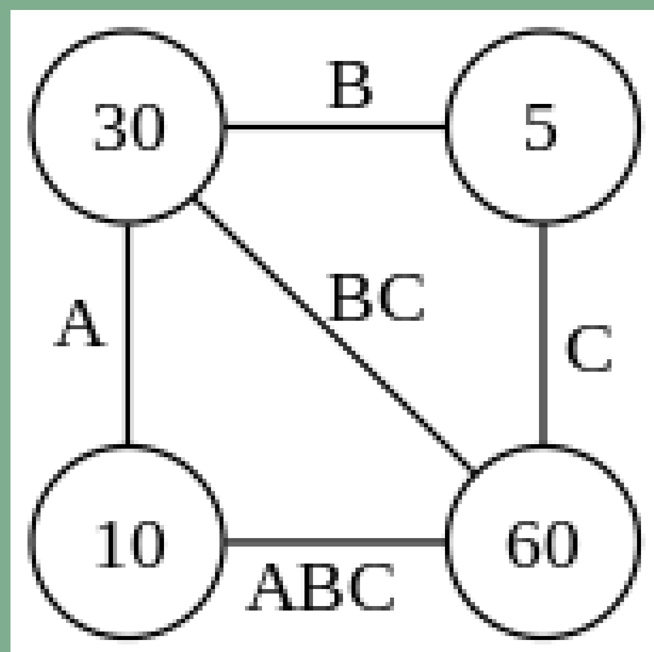
Primjer

- Druga mogućnost:



Primjer

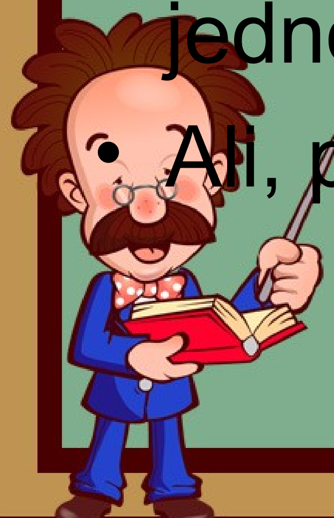
- Druga mogućnost:



27000
operacija

Zaključak

- Brute force metoda ima eksponencijalnu složenost i puno puta ponavlja računanje određenih vrijednosti
- Dinamičko programiranje daje polinomijalnu složenosti i relativno jednostavan algoritam
- Ali, postoje algoritmi i bolje složenosti



Literatura

- D. M. MOUNT, Design and analysis of computer algorithms, Department of Computer Science, University of Maryland, 2008., 111-115.
- Thomas H.Cormen, Charles E. Leiserson, Ronals L. Rivest, Clifford Stein, Introduction to Algorithms, Second Edition, The MIT Press, Cambridge, Massachussets London, England, 284-290.

Hvala na pažnji!