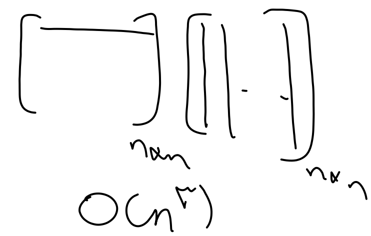
ضرب ماتریس ها

الگوریتم کلاسیک:

```
for (i=1;  i<= n; i++)
    for (j=1;   j <= n; j++) {
        C[i][j] = 0;
        for (k=1; k <=n;  k++)
            C[i][j] = C[i][j] + A[i][k] * B[k][j];
    }
```



تعداد ضرب ها: n^3

تعداد جمع ها: n^3

با تغییر جزیی زیر می توان تعداد جمع ها را کاهش داد:

```
S = 0;
S = S + A;                          S = A;
S = S + B;     ──── راه بهتر ───>    S = S + B;
S = S + C;                          S = S + C;
```

```
for (i=1;  i<= n; i++)
    for (j=1;   j <= n; j++) {
        C[i ,j] = A[i][1] * B[1][j] ;
        for (k=2; k <=n;  k++)
            C[i][j] = C[i][j] + A[i][k] * B[k][j];
    }
```

تعداد ضرب ها: n^3

تعداد جمع ها: $n^3 - n^2$

ایده استراسن:

$$\begin{aligned}A&=x\cdot y;\\ B&=x\cdot y\cdot z;\end{aligned}\quad\xrightarrow{\text{راه بهتر}}\quad\begin{aligned}A&=x\cdot y;\\ B&=A\cdot z;\end{aligned}$$

مثال:

$$\begin{bmatrix}c_{11} & c_{12}\\ c_{21} & c_{22}\end{bmatrix}=\begin{bmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\end{bmatrix}\times\begin{bmatrix}b_{11} & b_{12}\\ b_{21} & b_{22}\end{bmatrix}.$$

$$\begin{aligned}
m_1 &= (a_{11} + a_{22})(b_{11} + b_{22})\\
m_2 &= (a_{21} + a_{22})b_{11}\\
m_3 &= a_{11}(b_{12} - b_{22})\\
m_4 &= a_{22}(b_{21} - b_{11})\\
m_5 &= (a_{11} + a_{12})b_{22}\\
m_6 &= (a_{21} - a_{11})(b_{11} + b_{12})\\
m_7 &= (a_{12} - a_{22})(b_{21} + b_{22}),
\end{aligned}$$

$$C=\begin{bmatrix}m_1 + m_4 - m_5 + m_7 & m_3 + m_5\\ m_2 + m_4 & m_1 + m_3 - m_2 + m_6\end{bmatrix}.$$

حالت کلی:

$$\begin{array}{cc} & \leftarrow n/2 \rightarrow \\ \updownarrow n/2 & \left[\begin{array}{c|c} C_{11} & C_{12}\\ \hline C_{21} & C_{22}\end{array}\right]\end{array} = \left[\begin{array}{c|c} A_{11} & A_{12}\\ \hline A_{21} & A_{22}\end{array}\right] \times \left[\begin{array}{c|c} B_{11} & B_{12}\\ \hline B_{21} & B_{22}\end{array}\right]$$

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} M_1 + M_4 - M_5 + M_7 & M_3 + M_5 \\ M_2 + M_4 & M_1 - M_2 + M_3 + M_6 \end{bmatrix}.$$

$$\begin{aligned}
M_1 &= (A_{11} + A_{22})(B_{11} + B_{22}); \\
M_2 &= (A_{21} + A_{22})B_{11}; \\
M_3 &= A_{11}(B_{12} - B_{22}); \\
M_4 &= A_{22}(B_{21} - B_{11}); \\
M_5 &= (A_{11} + A_{12})B_{22}; \\
M_6 &= (A_{21} - A_{11})(B_{11} + B_{12}); \\
M_7 &= (A_{12} - A_{22})(B_{21} + B_{22}),
\end{aligned}$$

```
void strassen (int n
               n × n_matrix A,
               n × n_matrix B,
               n × n_matrix& C)
{
  if (n <= threshold)
    compute C = A × B using the standard algorithm;
  else {
    partition A into four submatrices A11, A12, A21, A22;
    partition B into four submatrices B11, B12, B21, B22;
    compute C = A × B using Strassen's Method;
      // example recursive call; strassen(n/2, A11 + A22, B11 + B22, M1)
  }
}
```

تعداد ضربها:

$$T(n) = 7T\left(\frac{n}{2}\right) \qquad \text{for } n > 1,\ n \text{ a power of } 2$$

$$T(1) = 1$$

$$T(n) = n^{\lg 7} \approx n^{2.81} \in \Theta(n^{2.81}).$$

تعداد جمع و تفریق ها:

$$T(n) = 7T\left(\frac{n}{2}\right) + 18\left(\frac{n}{2}\right)^2 \qquad \text{for } n > 1,\ n \text{ a power of } 2$$

$$T(1) = 0$$

با جایگزینی 2^k بجای n داریم:

$$T(2^k) = 7T(2^{k-1}) + 18(2^{k-1})^2.$$

حال قرار دهید $t_k = T(2^k)$ پس داریم

$$t_k = 7t_{k-1} + 18(2^{k-1})^2.$$

یا

$$
\begin{aligned}
t_k &= 7t_{k-1} + 18(2^{k-1})^2 \\
&= 7t_{k-1} + 18(4^{k-1}) \\
&= 7t_{k-1} + 4^k\left(\frac{18}{4}\right).
\end{aligned}
$$

جواب این معادله بصورت زیر است:

$$t_k = c_1 7^k + c_2 4^k.$$

پس

$$T(2^k) = c_1 7^k + c_2 4^k. \qquad n = ۲^k$$

و

$$
\begin{aligned}
T(n) &= c_1 7^{\lg n} + c_2 4^{\lg n} \\
&= c_1 n^{\lg 7} + c_2 n^2.
\end{aligned}
$$

چون $T(1) = 0$ پس

$T(۱) = c_۱ + c_۲ = ۰ \Rightarrow c_۱ = -c_۲$

$T(۲) = c_۱\, ۲^{\lg ۷} + ۴c_۲ = ۱۸$

$c_۱\, ۲^{\lg ۷} - ۴c_۱ = ۱۸$

$c_۱(۲^{\lg ۷} - ۴) = ۱۸ \Rightarrow c_۱ = ۶$

$$T(n) = 6n^{\lg 7} - 6n^2 \approx 6n^{2.81} - 6n^2 \in \Theta(n^{2.81}).$$

نکته: اگر تعداد سطرها و ستونهای ماتریس توانی از 2 نباشد یک راه ساده اضافه کردن سطرها و ستون های صفر به ماتریس است. روش دیگر اضافه کردن یک سطر و ستون در فراخوانی بازگشتی است وقتی که تعداد سطر و ستون فرد باشد.

مقایسه روش استاندارد و استراسن

عمل	روش استاندارد	روش استراسن
ضرب	n^3	$n^{2.81}$
جمع / تفریق	$n^3 - n^2$	$6n^{2.81} - 6n^2$

ضرب اعداد صحیح بزرگ (روش Karatsuba)

هر عدد صحیح u با n رقم را می توان بصورت زیر نشان داد:

$$\underbrace{u}_{n \text{ digits}} = \underbrace{x}_{\lceil n/2 \rceil \text{ digits}} \times 10^m + \underbrace{y}_{\lfloor n/2 \rfloor \text{ digits}}$$

که در آن

$$m = \left\lfloor \frac{n}{2} \right\rfloor$$

$$\underbrace{567{,}832}_{6 \text{ digits}} = \underbrace{567}_{3 \text{ digits}} \times 10^3 + \underbrace{832}_{3 \text{ digits}}$$

$$\underbrace{9{,}423{,}723}_{7 \text{ digits}} = \underbrace{9423}_{4 \text{ digits}} \times 10^3 + \underbrace{723}_{3 \text{ digits}}$$

برای دو عدد صیحیح n رقمی

$$u = x \times 10^m + y$$
$$v = w \times 10^m + z,$$

پس

$$\begin{aligned} uv &= (x \times 10^m + y)(w \times 10^m + z) \\ &= xw \times 10^{2m} + (xz + wy) \times 10^m + yz. \end{aligned}$$

همانطور که ملاحظه می شود برای ضرب دو عدد u و v چهار ضرب از اعداد کوچکتر لازم است.

مثال:

$$
\begin{aligned}
567{,}832 \times 9{,}423{,}723 &= (567 \times 10^3 + 832)(9423 \times 10^3 + 723) \\
&= 567 \times 9423 \times 10^6 + (567 \times 723 + 9423 \times 832) \\
&\quad \times 10^3 + 832 \times 723
\end{aligned}
$$

همانطور که ملاحظه می شود برای ضرب دو عدد 6 رقمی به چهار ضرب 3 رقمی نیاز است.

Large Integer Multiplication

Problem: Multiply two large integers, u and v.

Inputs: large integers u and v.

Outputs: *prod*, the product of u and v.

```
large_integer prod (large_integer u, large_integer v)
{
   large_integer x, y, w, z;
   int n, m;


  n = maximum(number of digits in u, number of digits in v)
  if (u == 0 || v == 0)
     return 0;
  else if (n <= threshold)
     return u × v obtained in the usual way;
  else {
     m = ⌊n/2⌋;
     x = u divide 10^m; y = u rem 10^m;
     w = v divide 10^m; z = v rem 10^m;
    return prod(x,w) × 10^2m + (prod(x,z) + prod(w,y)) × 10^m + prod(y,z);
  }
}
```

تعداد عملیات در بدترین حالت: 4 بار فراخوانی خود تابع به علاوه سایر عملیات که مجموع آنها cn است:

$$T(n) = 4T\left(\frac{n}{2}\right) + cn, n > s, T(s) = 0$$

که در آن s عددی است که پس از آن روش تقسیم و حل را ادامه نمی دهیم. داریم

$$T(n) = \theta(n^2)$$

در الگوریتم فوق برای محاسبه

$$xw, \qquad xz + yw, \qquad \text{and} \qquad yz,$$

مقادیر

$$xw, \qquad xz, \qquad yw, \qquad \text{and} \qquad yz.$$

محاسبه می شوند که 4 ضرب لازم دارد. اما این کار را می توان با 3 ضرب انجام داد:

$$r = (x + y)(w + z) = xw + (xz + yw) + yz,$$

$$xz + yw = r - xw - yz.$$

در بدترین حالت داریم:

$$T(n) = 3T\left(\frac{n}{2}\right) + cn,, n > s, T(s) = 0$$

پس

$$T(n) = \theta\left(n^{\log 3}\right) = \theta(n^{1.58})$$

ضرب چند جمله ای ها

یک چند جمله ای درجه n-1 بصورت زیر در نظر بگیرید:

$$A(x) = \sum_{j=0}^{n-1} a_j x^j$$
$$= a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1}$$

محاسبه آن در یک نقطه داده شده با استفاده از قاعده هورنر به $\theta(n)$ عملیات نیاز دارد:

$$A(x_0) = a_0 + a_1 x_0 + a_2 (x_0)^2 + \cdots + a_{n-1}(x_0)^{n-1}$$
$$= a_0 + x_0\left(a_1 + x_0\left(a_2 + \cdots + x_0\left(a_{n-2} + x_0(a_{n-1})\right)\cdots\right)\right)$$

جمع دو چند جمله ای نیز به $\theta(n)$ عملیات نیاز دارد:

$$C(x) = A(x) + B(x)$$

$$A(x) = \sum_{j=0}^{n-1} a_j x^j \text{ and } B(x) = \sum_{j=0}^{n-1} b_j x^j .$$

داریم

$$C(x) = \sum_{j=0}^{n-1} c_j x^j$$

که

$$c_j = a_j + b_j \text{ for } 0 \le j \le n-1$$

اما برای ضرب داریم

$$C(x) = A(x)B(x)$$

که

$$C(x) = \sum_{j=0}^{2n-2} c_j x^j,$$

و

$$c_j = \sum_{k=0}^{j} a_k b_{j-k} \text{ for } 0 \le j \le 2n-2 .$$

محاسبه c نیاز به $\theta(n^2)$ عملیات دارد. اما با ایده تقسیم و حل

$$A(x)=\sum_{j=0}^{n-1} a_j x^j=\sum_{j=0}^{\frac{n}{2}-1} a_j x^j + x^{\frac{n}{2}}\sum_{j=0}^{\frac{n}{2}-1} a_{\frac{n}{2}+j} x^j = A_1(x) + x^{\frac{n}{2}} A_2(x)$$

$$B(x)=\sum_{j=0}^{n-1} b_j x^j=\sum_{j=0}^{\frac{n}{2}-1} b_j x^j + x^{\frac{n}{2}}\sum_{j=0}^{\frac{n}{2}-1} b_{\frac{n}{2}+j} x^j = B_1(x) + x^{\frac{n}{2}} B_2(x)$$

$$\begin{aligned} C(x) &= A(x)B(x) \\ &= A_1(x)B_1(x) + x^{\frac{n}{2}}[A_1(x)B_2(x) + A_2(x)B_1(x)] + x^n\ A_2(x)B_2(x) \end{aligned}$$

اما

$$A_1(x)B_2(x) + A_2(x)B_1(x)$$
$$= [A_1(x) + A_2\ (x)][B_1(x) + B_2\ (x)] - A_1(x)B_1(x) - A_2(x)B_2(x)$$

که $\mathrm{O}\left(n^{\log_2 3}\right) = \mathrm{O}(n^{1.59})$ عملیات لازم دارد.

در چه مواقعی نباید از تقسیم و حل استفاده کرد؟ تعیین مقادارآستانه!

مثال: فرض کنید الگوریتمی به روش معمول $\frac{n(n-1)}{2}\ \mu s$ زمان و با روش تقسیم و حل

$$T\left(\left\lfloor\frac{n}{2}\right\rfloor\right)+T\left(\left\lceil\frac{n}{2}\right\rceil\right)+32n\ \mu s$$

تا چه بعدی استفاده از روش تقسیم و حل بهتر است؟

جواب:

$$T\left(\left\lfloor\frac{t}{2}\right\rfloor\right)+T\left(\left\lceil\frac{t}{2}\right\rceil\right)+32t=\frac{t(t-1)}{2}$$

چون $\left\lfloor\frac{t}{2}\right\rfloor$ و $\left\lceil\frac{t}{2}\right\rceil$ هر دو از t کوچکترند پس

$$\frac{\lfloor t/2\rfloor(\lfloor t/2\rfloor-1)}{2}+\frac{\lceil t/2\rceil(\lceil t/2\rceil-1)}{2}+32t=\frac{t(t-1)}{2}.$$

برای t زوج t=128 و برای t فرد t=128.008.

- پیچیدگی زمانی یک الگوریتم تقسیم و حل بر روی یک کامپیوتر خاص:

$$T(n) = 3T\left(\left\lceil \frac{n}{2} \right\rceil\right) + 16n \ \mu s$$

- پیچیدگی زمانی یک الگوریتم تکراری برای حل نمونه ای به اندازه n:

$$T(n) = n^2 \ \mu s$$

- تعیین مقدار آستانه بهینه t:

$$3T\left(\left\lceil \frac{t}{2} \right\rceil\right) + 16t = t^2 \ \& \ T\left(\left\lceil \frac{T}{2} \right\rceil\right) = \left\lceil \frac{t}{2} \right\rceil^2 \Rightarrow$$

$$3\left\lceil \frac{t}{2} \right\rceil^2 + 16t = t^2$$

الف) اگر t زوج باشد، آنگاه $t = 64$

ب) اگر t فرد باشد، آنگاه $t = 70.04$

چون دو مقدار برابر نیستند، آستانه بهینه وجود ندارد؛ یعنی:

- اگر n یک عدد صحیح زوج بین ۶۴ و ۷۰ باشد، بهتر است یک بار دیگر تقسیم شود
- اگر n یک عدد صحیح فرد بین ۶۴ و ۷۰ باشد، فراخوانی الگوریتم جانشین کارآیی بیشتری دارد
- اگر n کوچکتر از ۶۴ باشد، همواره فراخوانی الگوریتم جانشین کارآیی بیشتری دارد
- اگر n بزرگتر از ۷۰ باشد، همواره تقسیم دوباره نمونه کارآتر خواهد بود.

مقایسه کارآیی الگوریتم بازگشتی و جانشین به ازاء مقادیر مختلف n:

n	n^2	$3\left\lceil \frac{n}{2} \right\rceil^2 + 16n$
62	3844	3875
63	3969	4080
64	4096	4096
65	4225	4307
68	4624	4556
69	4761	4779
70	4900	4795
71	5041	5024

چه مواقعی نباید از روش تقسیم و حل استفاده کرد

- یک نمونه به اندازه n به دو یا چند نمونه تقسیم شود به طوری که اندازه هر یک از این نمونه ها تقریبا برابر اندازه نمونه اصلی باشد. ← نمایی
  - مثال: دنباله فیبوناچی
- یک نمونه به اندازه n تقریبا به n نمونه با اندازه های n/c تقسیم شود به طوری که c یک عدد ثابت باشد. ← $\Theta(n^{\lg n})$