برنامه نویسی پویا

ضرب زنجیره‌ای ماتریس‌ها (Chained Matrix Multiplication)

پرسش: تعداد ضربهای مورد نیاز ؟

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 7 & 8 & 9 & 1 \\ 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 \end{bmatrix} = ?$$

پاسخ:

$$A_{2\times 3} B_{3\times 4} = C_{2\times 4}, \quad c_{ij} = \sum_{k=1}^{3} a_{ik} b_{kj}$$

So each element requires **3 multiplications**. The total number of multiplications for C is

$$2 \times 3 \times 4 = 24.$$

در حالت کلی **تعداد ضربهای مورد نیاز** $A_{i\times j} B_{j\times k}$ برابر $\boldsymbol{i \times j \times k}$ است.

مثال: تعداد ضربهای موردنیاز برای $A_{20\times 2} B_{2\times 30} C_{30\times 12} D_{12\times 8}$ حساب کنید.

نخست باید ترتیب‌های ممکن را ایجاد و سپس در ترتیب‌های ممکن، تعداد حاصلضرب را حساب کنیم.

$((AB)C)D:\ 20\times 2\times 30 + 20\times 30\times 12 + 20\times 12\times 8 = 10320$

$(A(BC))D:\ 2\times 30\times 12 + 20\times 2\times 12 + 20\times 12\times 8 = 3120$

$A(B(CD)):\ 30\times 12\times 8 + 2\times 30\times 8 + 20\times 2\times 8 = 3680$

$(AB)(CD):\ 20\times 2\times 30 + 30\times 12\times 8 + 20\times 30\times 8 = 8880$

$A((BC)D):\ 2\times 30\times 12 + 2\times 12\times 8 + 20\times 2\times 8 = 1232$

مساله اصلی یافتن ترتیبی از محاسبه حاصلضرب‌هاست که کمترین تعداد ضرب را نیاز داشته باشد.

پرسش: ترتیب‌های ممکن برای حاصلضرب پنج ماتریس $A_1 \times A_2 \times A_3 \times A_4 \times A_5$ چیست؟ ترتیب‌های ممکن را نشان دهید.

رویکرد برنامه‌نویسی پویا

$$\begin{cases} M[i][j] = \min\limits_{i \le k \le j-1} \{M[i][k] + M[k+1][j] + d_{i-1} d_k d_j\}, & \text{for } i < j \\ M[i][i] = 0 \end{cases}$$

پرسش: آیا اصل بهینگی برای مساله یافتن ترتیب بهینه(کمترین تعداد ضرب)، برقرار است؟

مثال:

$$\begin{array}{cccccc} A_1 \times & A_2 \times & A_3 \times & A_4 \times & A_5 \times & A_6 \\ 5\times 2 & 2\times 3 & 3\times 4 & 4\times 6 & 6\times 7 & 7\times 8 \\ d_0\, d_1 & d_1\, d_2 & d_2\, d_3 & d_3\, d_4 & d_4\, d_5 & d_5\, d_6 \end{array}$$

$M[1][2] = 5\times 2\times 3 = 30$

$M[2][3] = 2\times 3\times 4 = 24$

$M[3][4] = 3\times 4\times 6 = 72$

$M[4][5] = 4\times 6\times 7 = 168$

$M[5][6] = 6\times 7\times 8 = 336$

$$M[1][3] = \min_{1 \le k \le 2}\{M[1][k] + M[k+1][3] + d_0 d_k d_3\}$$
$$= \min\{M[1][1] + M[2][3] + d_0 d_1 d_3,$$
$$M[1][2] + M[3][3] + d_0 d_2 d_3\}$$
$$= \min\{0 + 24 + 5 \times 2 \times 4,\ 30 + 0 + 5 \times 3 \times 4\}$$
$$= 64$$

$$M[1][4] = \min_{1 \le k \le 3}\{M[1][k] + M[k+1][4] + d_0 d_k d_4\}$$
$$= \min\{M[1][1] + M[2][4] + d_0 d_1 d_4,$$
$$M[1][2] + [3][4] + d_0 d_2 d_4,$$
$$M[1][3] + [4][4] + d_0 d_3 d_4\}$$
$$= 132$$

Problem: Determining the min no. of multiplications needed to multiply *n* matrices .
Inputs : no. of matrices, *n*, array of integers *d* where
$d[i-1]$x$d[i]$ is the dimension of the *i*th matrix.
Outputs: *minmult*, the min no. of multiplications.

```
void minmult (int n, const int d[], index P[][])
{
   index i, j, k, diagonal;
   int M[1..n][1..n];

  for (i=1; i ≤ n; i++)
      M[i][i]=0;

   for (diagonal=1; diagonal ≤ n–1; diagonal++) {
        for (i=1; i ≤ n– diagonal; i++)
          {
            j = i+ diagonal;
            M[i][j] = min_{i≤k≤j−1} {M[i][k] + M[k+1][j] + d[i−1]d[k]d[j]}
            P[i][j] = a value of k that gave the minimum;
          }
       return M[1][n]; }

}
```

Alg. 6-1

Time Complexity?

$$T(n) \in \Theta(n^3)$$

Problem: Print the optimal order for multiplying *n* matrices.

Inputs: positive integer *n* and the array *P*.

Output: the optimal order for multiplying the matrices.

چاپ ترتیب بهینه برای پرانتزگذاری ماتریس‌ها

```
void order (index i, j)
 {
   if (i = = j)
        cout  << "A" << i;
     else {
             k=P[i][j] ;
             cout  << "(";
             order(i, k);
             order(k+1, j);
             cout  << ")";
           }
 }
```

Alg. 6-2

Time Complexity?

$$T(n) \in \Theta(n)$$

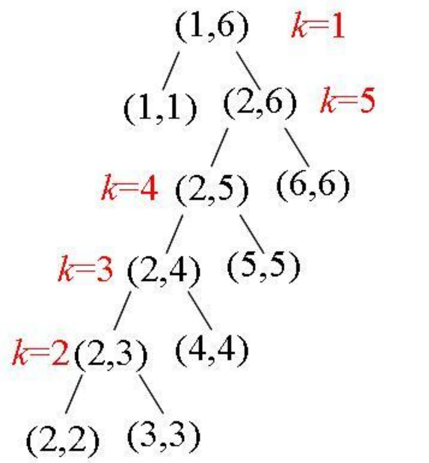
پرسش: چگونه از ماتریس P، ترتیب پرانتزگذاری ماتریس‌ها(شکل ۶-۱)، جهت محاسبه حاصلضرب، به‌دست‌می‌آید؟

$$
\begin{array}{c|cccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & 0 & 30 & 64 & 132 & 226 & 348 \\
2 & & 0 & 24 & 72 & 156 & 268 \\
3 & & & 0 & 72 & 198 & 366 \\
4 & & & & 0 & 168 & 392 \\
5 & & & & & 0 & 336 \\
6 & & & & & & 0
\end{array}
$$

ماتریس M

$$
P = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ & 0 & 2 & 3 & 4 & 5 \\ & & 0 & 3 & 4 & 5 \\ & & & 0 & 4 & 5 \\ & & & & 0 & 5 \\ & & & & & 0 \end{bmatrix}
$$

ماتریس P



$$(A_1((((A_2A_3)A_4)A_5)A_6))$$

شکل ۶-۱: درخت بازگشتی

پرسش: کمترین تعداد ضرب و ترتیب بهینه را برای چهار ماتریس $A_{2\times3} \times B_{3\times4} \times C_{4\times6} \times D_{6\times7}$ با روش برنامه نویسی پویا، بیابید.

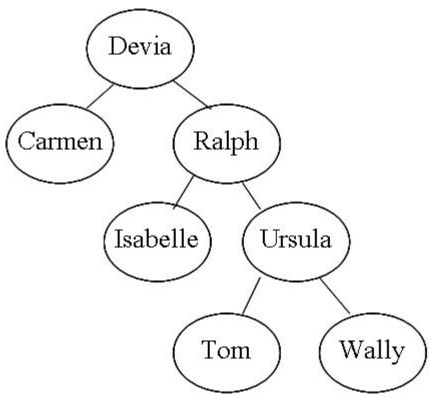
درخت جستجوی دودویی بهینه

(Optimal Binary Search Trees = OBST)

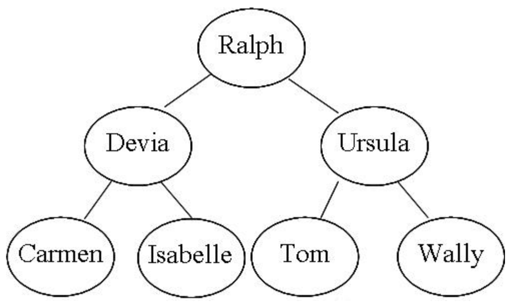
یادآوری از ساختمان داده‌ها

Definition of Binary Tree: A binary tree is the tree that each node has at most two children.

1. Depth (Level): the number of edges from the root to the node.
2. Left Subtree: the subtree whose root is the left child of the node.
3. Right Subtree: the subtree whose root is the right child of the node.
4. Balanced: the depth of the two subtrees of every node never differs by more than 1.



شکل ۶-۳: درخت دودویی نامتوازن



شکل ۶-۲: درخت دودویی متوازن

درخت جستجوی دودویی (Binary Search Trees = BST)

A binary search tree is a binary tree of items (keys) that come from an ordered set, such that:

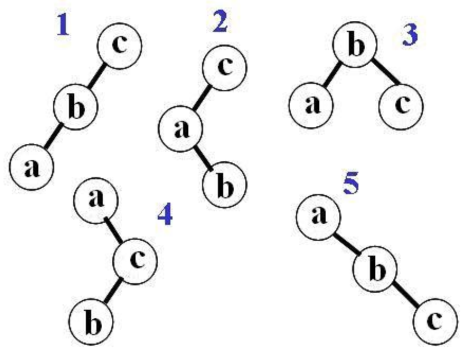
1. Each node contains one key.
2. The keys in the left subtree of a given node are less than the key in that node.
3. The keys in the right subtree of a given node are greater than or equal to the key in that node.

مثال:

Five different trees where n=3. Let p_a=0.7, p_b=0.2 and p_c=0.1. What is the average search time?

درخت	a	b	c

1. 3(0.7)+2(0.2)+1(0.1) = 2.6
2. 2(0.7)+3(0.2)+1(0.1) = 2.1
3. 2(0.7)+1(0.2)+2(0.1) = 1.8
4. 1(0.7)+3(0.2)+2(0.1) = 1.5
5. 1(0.7)+2(0.2)+3(0.1) = 1.4



شکل ۶-۴: درخت‌های جستجوی دودویی ممکن

یافتن درخت جستجوی دودویی بهینه

A more efficient algorithm is **Dynamic Programming**. The optimal tree for Key_i through Key_j is arranged in a tree that minimizes $A[i][j]=\sum_{m=i}^{j} c_m p_m \quad , \; A[i][i]=p_i$

c_m: the number of comparisons to locate Key_m. p_m: the probability that Key_m is the search key.

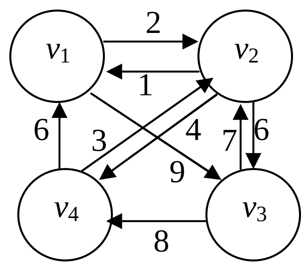
$$A[i][j]=\min_{i\le k\le j}\{A[i][k-1]+A[k+1][j]\}+\sum_{m=i}^{j} p_m, \quad \text{for } i<j$$

نکته: الگوریتم برنامه‌نویسی پویا برای یافتن درخت جستجوی دودویی بهینه، مشابه الگوریتم یافتن ترتیب بهینه در حاصلضرب زنجیره‌ای ماتریس‌هاست.

مساله فروشنده دوره‌گرد (Traveling Salesperson Problem = TSP)

```
void travel (int n, const number W[][], index P[][], number& minlength )
{ index i, j, k;
  number D[1..n][subset of V–{v1}];
  for (i = 2; i ≤ n; i++)
      D[i][ϕ]=W[i][1] ;
  for (k = 1; k ≤ n–2; k++)
      for (all subsets A⊆ V–{v1} containing k vertices)
          for (i such that i ≠1 and vi is not in A) {
              D[i][A]=min {W[i][j]+D[j][A–{vj}]} ;
              P[i][A]= value of j that gave the minimum;}
  D[1][V–{v1} ]=min {W[i][1]+D[j][V–{v1,v2}]} ;
  P[i][V–{v1} ]= value of j that gave the minimum ;
  minlength=D[1][V–{v1}];
}
```

Alg. 6-3



شکل ۶-۵: گراف مثال

Length[$v1,v2,v3,v4,v1$] = 22
Length[$v1,v3,v2,v4,v1$] = 26
Length[$v1,v3,v4,v2,v1$] = 21

$$T(n)=\sum_{k=1}^{n-2}(n-1-k)k\binom{n-1}{k}=(n-1)\sum_{k=1}^{n-2}k\binom{n-2}{k}$$
$$=(n-1)(n-2)2^{n-3}\in\Theta(n^2 2^n)$$

If there are n vertices, how many possible tours?

$(n-1)(n-2)(n-3)\ldots 1=(n-1)!$

مثال: برای ۲۰ شهر، مساله فروشنده دوره‌گرد را بررسی کنید.

Brute-force: 19! µs = 3857 years ☹

Dynamic Programming: $(20-1)(20-2)2^{20-3}$ µs = 45 sec ☺!!

تمرین‌ها

تمرین ۶-۱: نشان دهید الگوریتم۶-۱، دارای تکرار $T(n)=\dfrac{n(n-1)(n+1)}{6}\in\Theta(n^3)$ است.

تمزین ۶-۲: ترتیب‌های ممکن برای حاصلضرب شش ماتریس $A_1\times A_2\times A_3\times A_4\times A_5\times A_6$ بیابید؟