11.串

（c5）KMP算法：分摊分析

邓俊辉

deng@tsinghua.edu.cn

失之东隅，收之桑榆

Ω(n * m)?

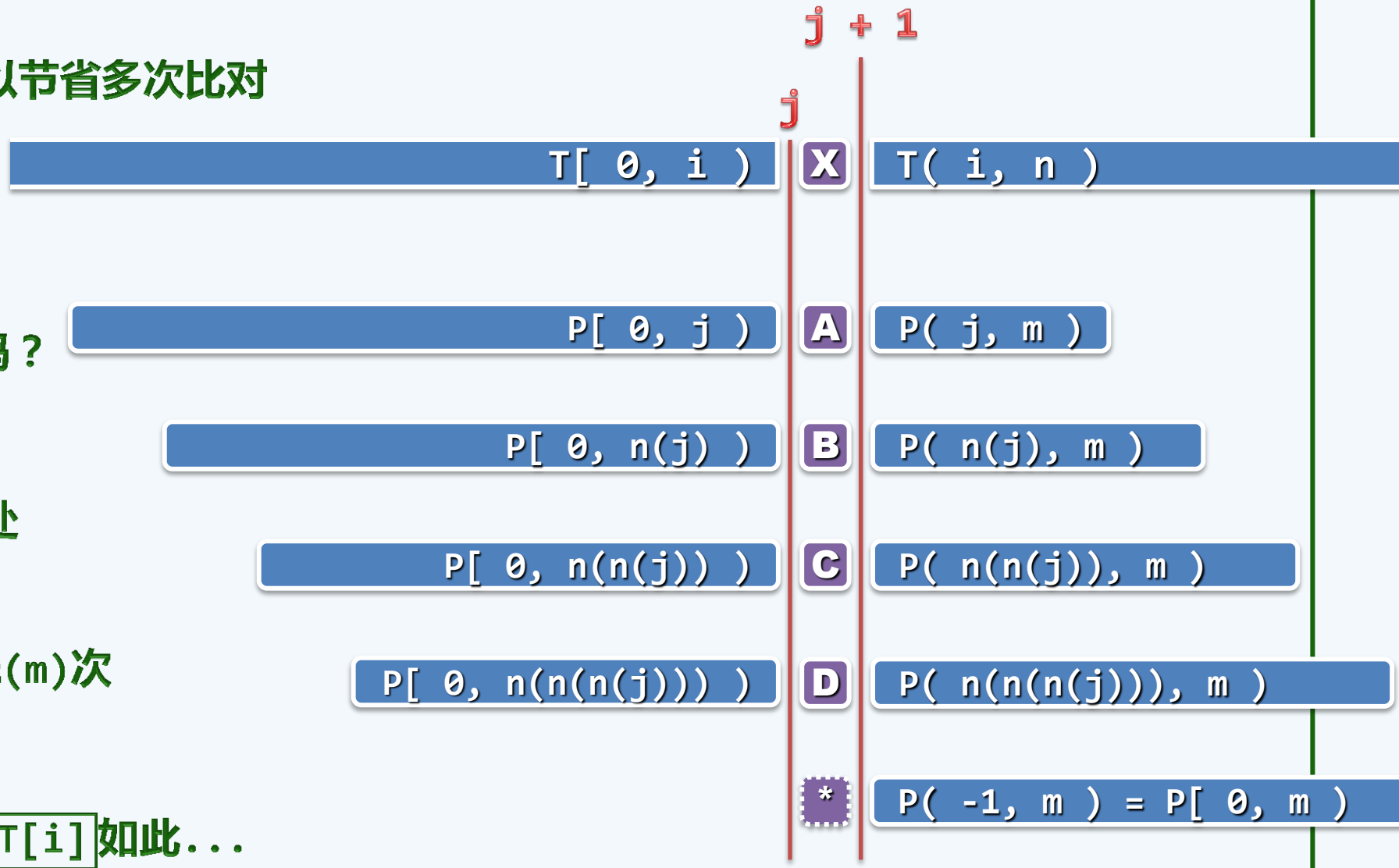
- 观察： KMP算法的确可以节省多次比对
- 然而： 就渐进意义而言

  有实质的节省吗？

- 观察： 在每一个T[i]处

  P都可能比对Ω(m)次

- 于是， 倘若有Ω(n)个T[i]如此...



Ω(n * m)?

- 难道， 总体复杂度还是

  Ω(n * m)?

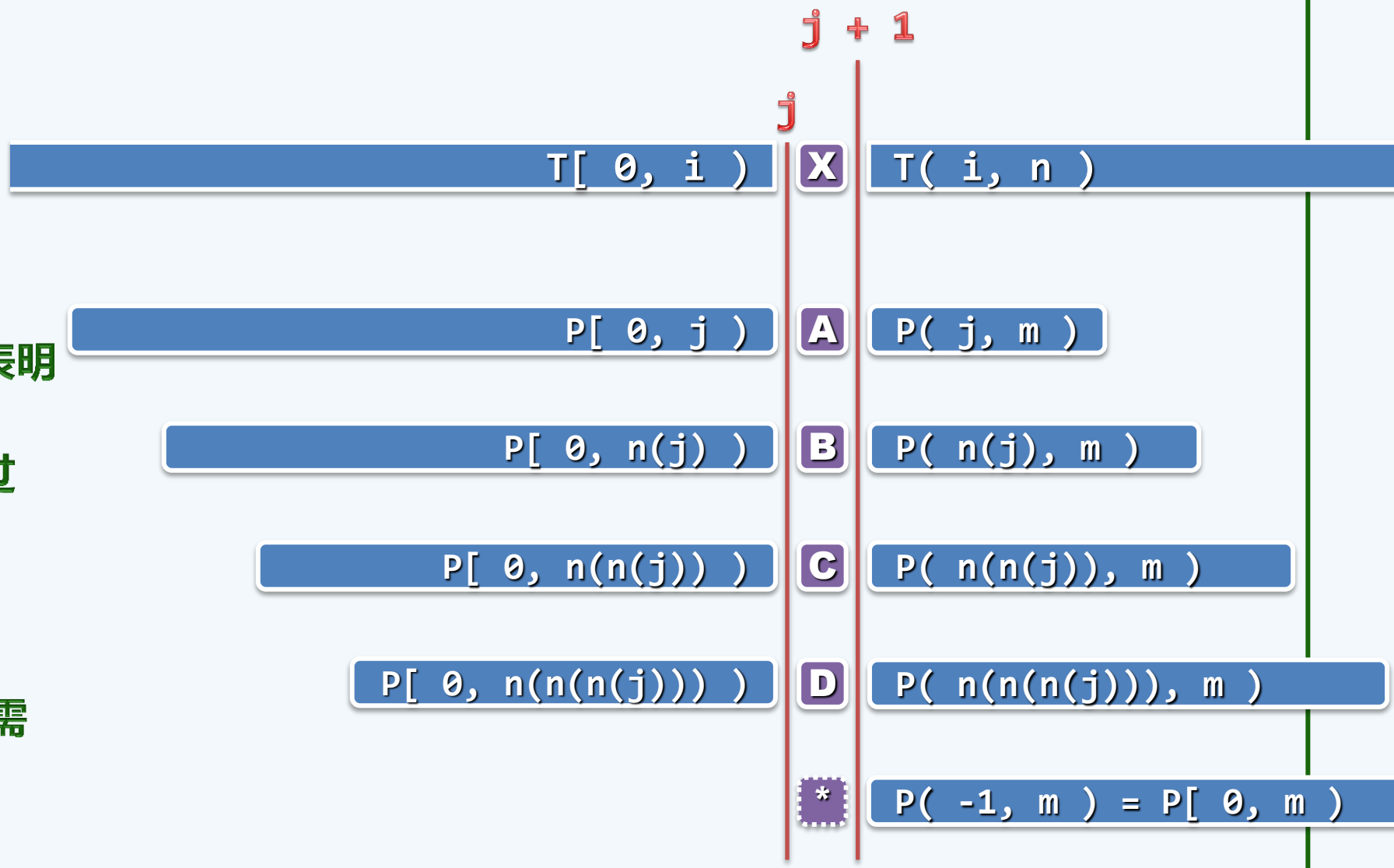
- 然而，更细致的分析将表明

  即便是最坏情况，也不过

  O(n)时间

- 同理，建立next[]也只需

  O(m)时间



O(n + m)！

❖ 令k = 2*i - j　　//具体含义，详见习题[11-4]

```
while ( j < m && i < n ) //k必随迭代而单调递增，故也是迭代步数的上界
   if ( 0 > j || T[i] == P[j] )
      { i ++;  j ++; } //i、j同时加1，故k恰好加1
   else
      j = next[j]; //i不变，j至少减1，故k至少加1
```

❖ k的初值为0；算法结束时，必有：

$$k = 2*i - j \le 2(n - 1) - (-1) = 2n - 1$$