01-E2

绪论

迭代与递归：分而治之

凡治众如治寡，分数是也。

邓俊辉

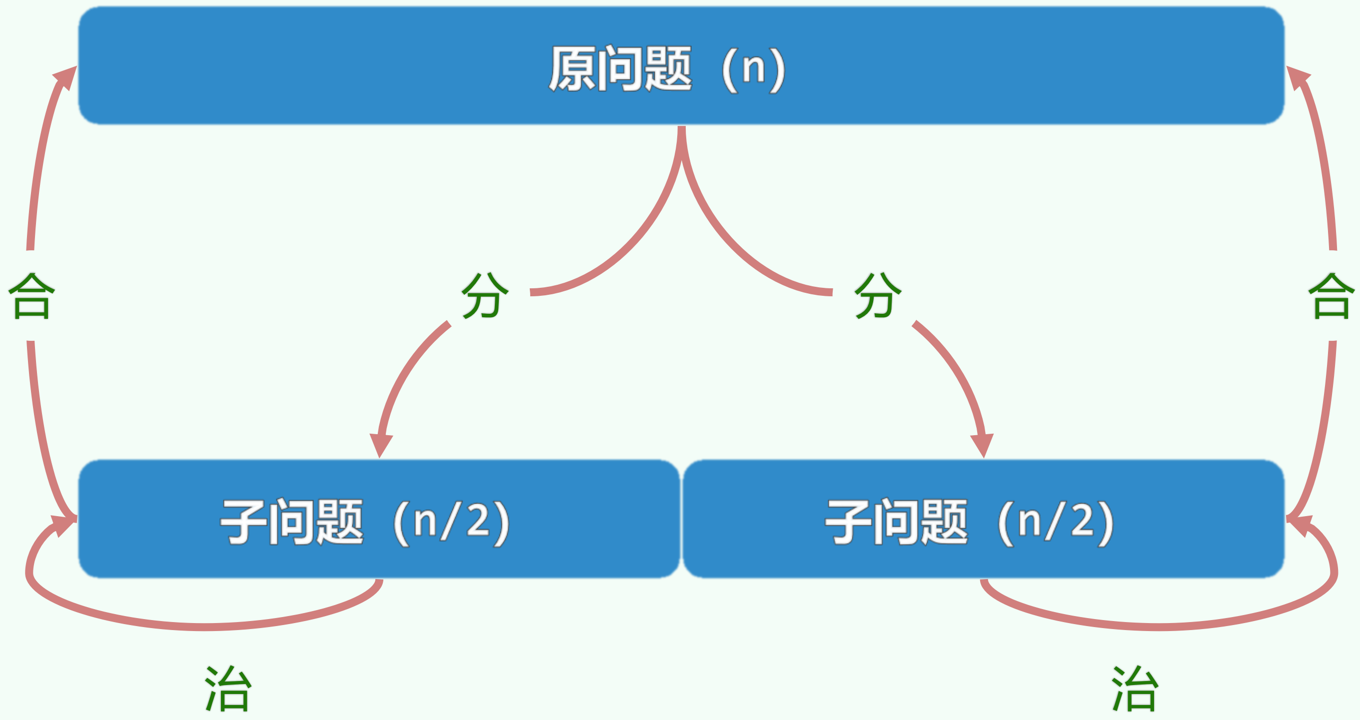
deng@tsinghua.edu.cn

Divide-and-Conquer

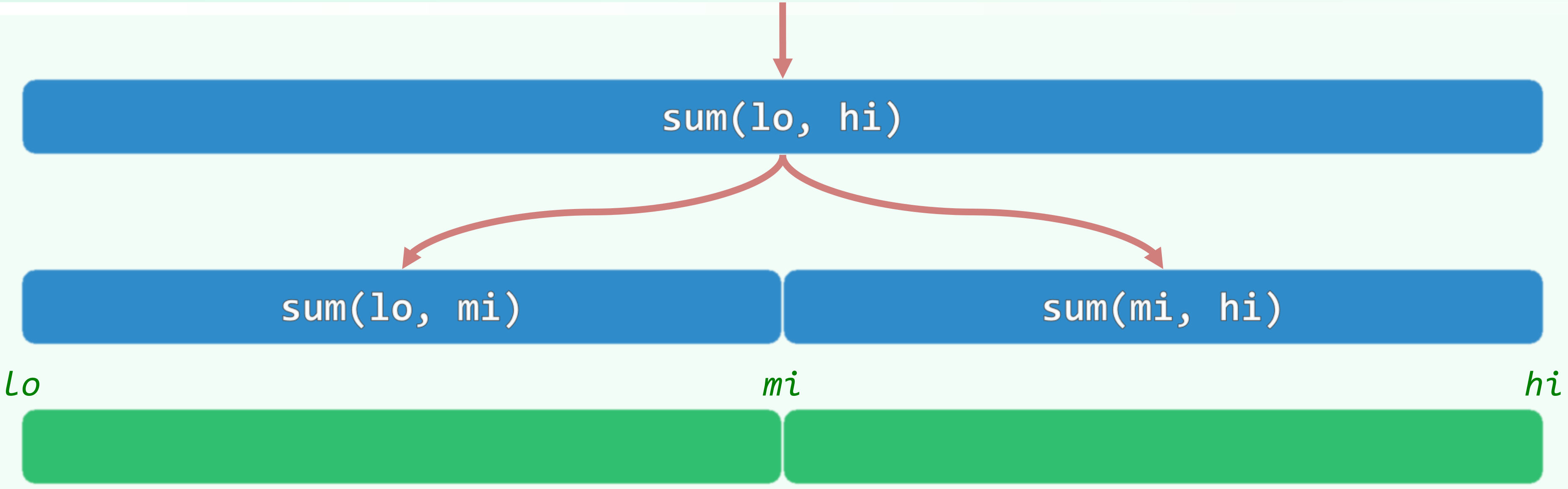
- ❖ 为求解一个大规模的问题，可以...
- ❖ 将其**划分**为若干子问题

  （通常两个，且规模**大体相当**）
- ❖ 分别求解子问题
- ❖ 由子问题的解

  合并得到原问题的解

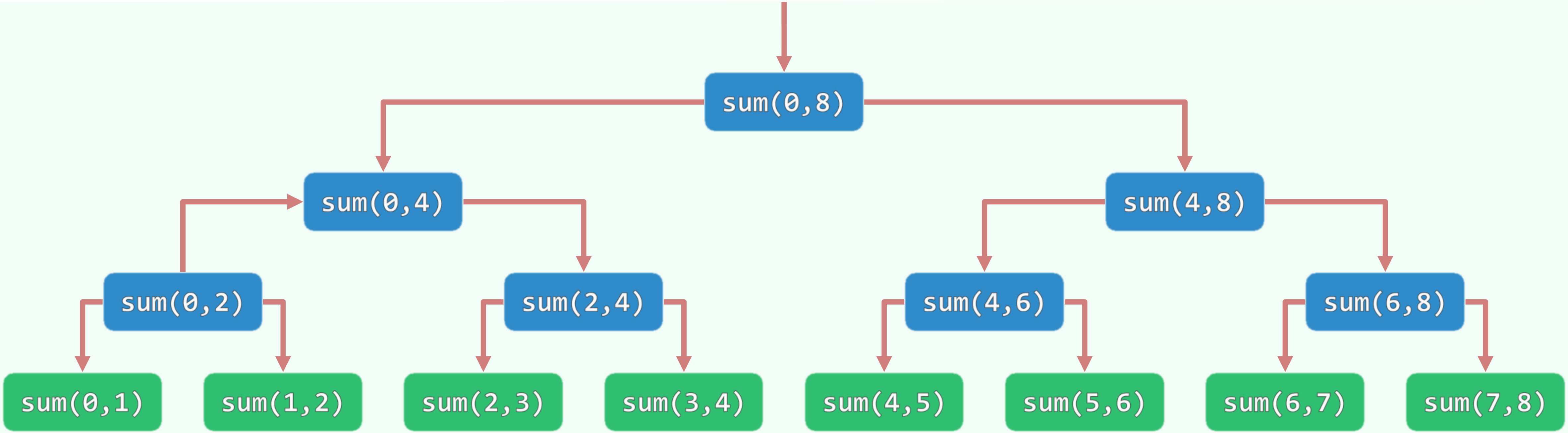


Binary Recursion




```
sum( int A[], int lo, int hi ) { //区间范围A[lo, hi)
   if ( hi - lo < 2 ) return A[lo];
   int mi = (lo + hi) >> 1;   return sum( A, lo, mi ) + sum( A, mi, hi );
} //入口形式为sum( A, 0, n )
```

Binary Recursion: Trace



T(n) = **各层递归实例所需时间之和** //递归跟踪

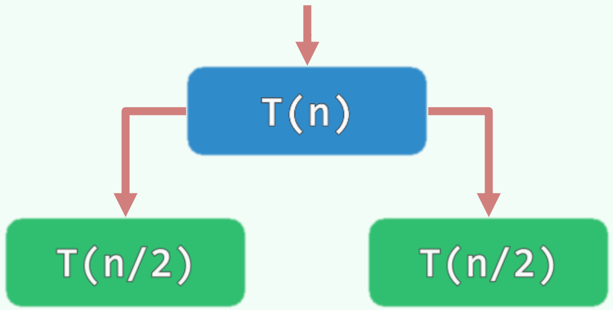
$$= \mathcal{O}(1) \times (2^0 + 2^1 + 2^2 + \cdots + 2^{\log n})$$

$$= \mathcal{O}(1) \times (2^{1+\log n} - 1) = \mathcal{O}(n)$$ //更快捷地，作为**几何**级数...

Binary Recursion: Recurrence

❖ 从递推的角度看，为求解sum(A, lo, hi)，需要

- 递归求解sum(A, lo, mi)和sum(A, mi, hi)，进而 //2*T(n/2)
- 将子问题的解累加 //$\mathcal{O}(1)$



❖ 递推方程：

$$T(n) = 2 \cdot T(n/2) + \mathcal{O}(1)$$

$$T(1) = \mathcal{O}(1) \quad \text{//base: sum(A, k, k)}$$

❖ 求解：

$$T(n) = 4 \cdot T(n/4) + \mathcal{O}(3) = 8 \cdot T(n/8) + \mathcal{O}(7) = 16 \cdot T(n/16) + \mathcal{O}(15) = \ldots$$

$$= n \cdot T(1) + \mathcal{O}(n-1) = \mathcal{O}(2n-1) = O(n)$$

Master Theorem (1/2)

分治策略对应的递推式，通常形如： //原问题被分为a个规模均为n/b的子任务

$$T(n) = a \cdot T(n/b) + \mathcal{O}(g(n))$$

//任务的划分、解的合并，总共耗时g(n)

❖ 若 $g(n) = \Omega(n^{\log_b a + \epsilon})$ ，则 $T(n) = \Theta(g(n))$

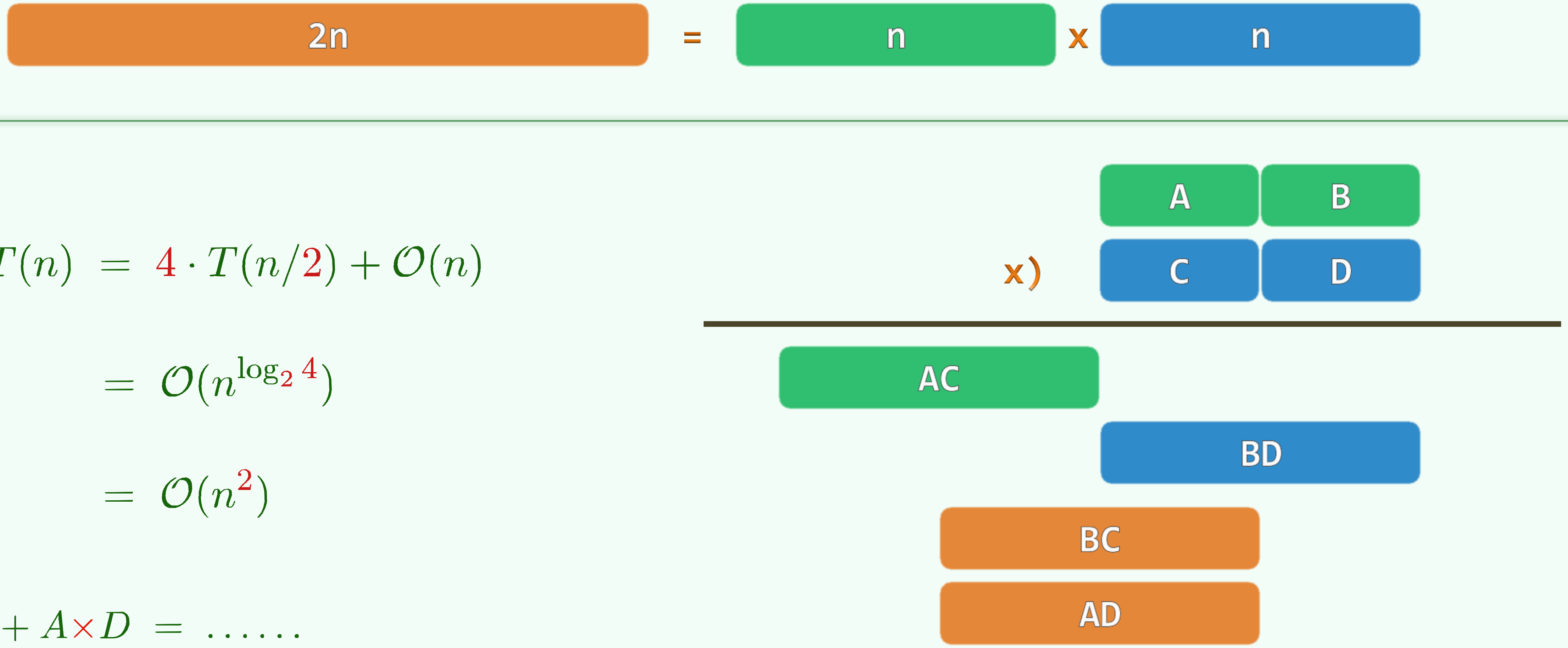
- quickSelect (average case): $T(n) = 1 \cdot T(n/2) + \mathcal{O}(n) = \mathcal{O}(n)$

❖ 若 $g(n) = \mathcal{O}(n^{\log_b a - \epsilon})$ ，则 $T(n) = \Theta(n^{\log_b a})$

compare >>

- recursive sum: $T(n) = 2 \cdot T(n/2) + \mathcal{O}(1) = \mathcal{O}(n)$
- kd-search: $T(n) = 2 \cdot T(n/4) + \mathcal{O}(1) = \mathcal{O}(\sqrt{n})$
- large integer multiplication: $T(n) = 3 \cdot T(n/2) + \mathcal{O}(n) = \mathcal{O}(n^{\log_2 3})$

Large Integer Multiplication: Naive + DAC



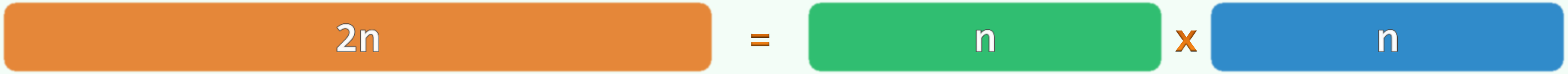
$$T(n) = 4 \cdot T(n/2) + \mathcal{O}(n)$$

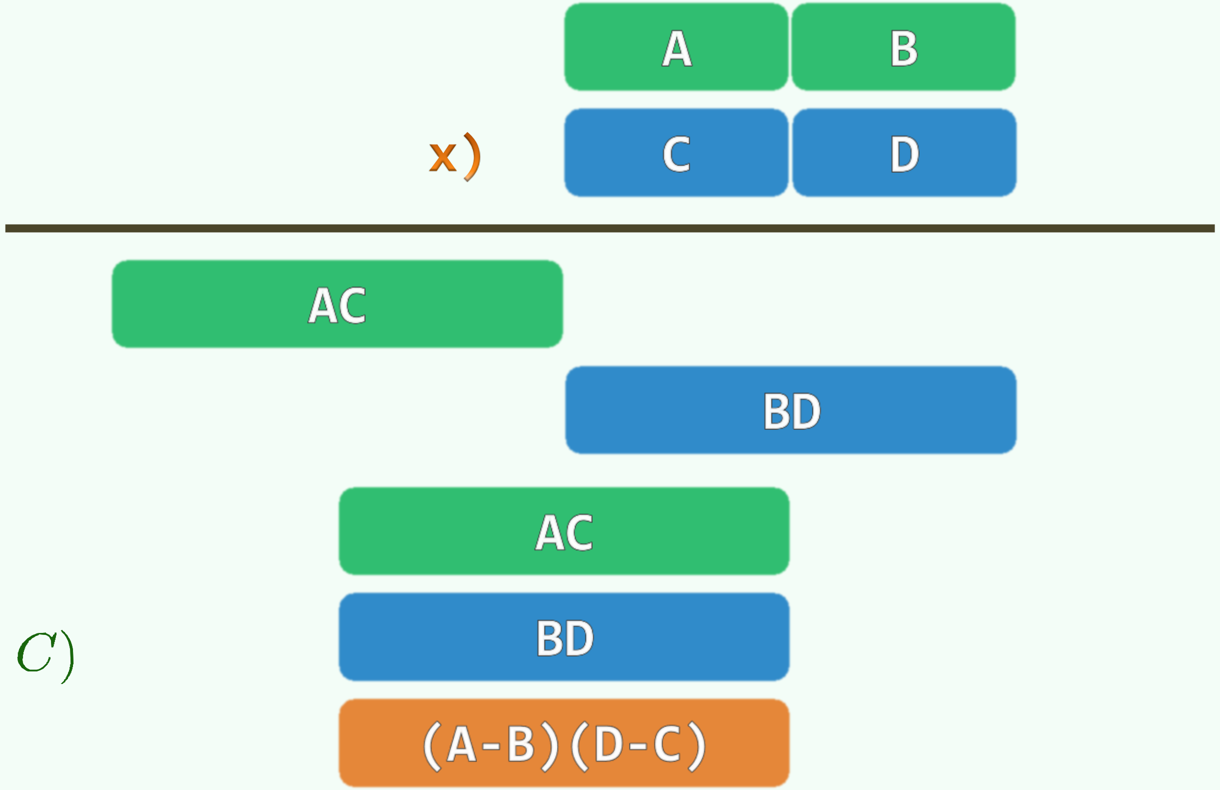
$$= \mathcal{O}(n^{\log_2 4})$$

$$= \mathcal{O}(n^2)$$

$$B \times C + A \times D = \ldots\ldots$$

Large Integer Multiplication: Optimal





$$T(n) = 3 \cdot T(n/2) + \mathcal{O}(n)$$

$$= \mathcal{O}(n^{\log_2 3})$$

$$\approx \mathcal{O}(n^{1.585})$$

$$B \times C + A \times D = A \cdot C + B \cdot D + (A - B) \times (D - C)$$

Master Theorem (2/2)

分治策略对应的递推式，通常形如： //原问题被分为a个规模均为n/b的子任务

$$T(n) \;=\; a \cdot T(n/b) + \mathcal{O}(g(n))$$ //任务的划分、解的合并，总共耗时g(n)

- 若 $g(n) = \Theta(n^{\log_b a} \cdot \log^k n)$ 且 $0 \le k$，则 $T(n) = \Theta(g(n) \cdot \log n) = \Theta(n^{\log_b a} \cdot \log^{k+1} n)$
  - binary search: $T(n) \;=\; 1 \cdot T(n/2) + \mathcal{O}(1) \;=\; \mathcal{O}(\log n)$
  - mergesort: $T(n) \;=\; 2 \cdot T(n/2) + \mathcal{O}(n) \;=\; \mathcal{O}(n \cdot \log n)$
  - STL mergesort: $T(n) \;=\; 2 \cdot T(n/2) + \mathcal{O}(n \cdot \log n) \;=\; \mathcal{O}(n \cdot \log^2 n)$ << compare

- 要是...不巧落在...这三种情况之间的**缝隙**呢？

  或者...子任务的...规模**不均等**呢？

Akra-Bazzi Theorem

分治策略对应的递推式，更一般地形如： //k组：各含a_i个规模为$b_i n+h_i(n)$的子任务

$$T(n) = \sum_{i=1}^{k} a_i \cdot T(b_i \cdot n + h_i(n)) + \mathcal{O}(g(n))$$ //任务的划分、解的合并，总共耗时g(n)

- $0 < a_i$，$0 < b_i < 1$，$|h_i(n)| = \mathcal{O}(n/\log^2 n)$
- $0 \leq g(n)$ **且有正的常数d使得** $|g'(n)| = \mathcal{O}(n^d)$ //多项式增长

❖ **只要取p使得** $\sum_{i=1}^{k} a_i \cdot b_i^p = 1$，**则** $T(n) = \Theta(n^p \cdot (1 + \int_1^n g(u)/u^{p+1} du))$ //与$h_i(n)$无关

❖ **以第14章的linearSelect算法为例**

- **坏选择：** $T(n) = 1 \cdot T(\underline{3/4} \cdot n) + 1 \cdot T(\underline{1/4} \cdot n) + \mathcal{O}(n) = \mathcal{O}(n \log n)$ $// \ p = 1$
- **好选择：** $T(n) = 1 \cdot T(\underline{3/4} \cdot n) + 1 \cdot T(\underline{1/5} \cdot n) + \mathcal{O}(n) = \mathcal{O}(n)$ $// \ p < 1$