绪论

迭代与递归：分而治之

01-E2

邓俊辉
deng@tsinghua.edu.cn

凡治众如治寡，分数是也

Divide-and-Conquer

❖ 为求解一个大规模的问题，可以...

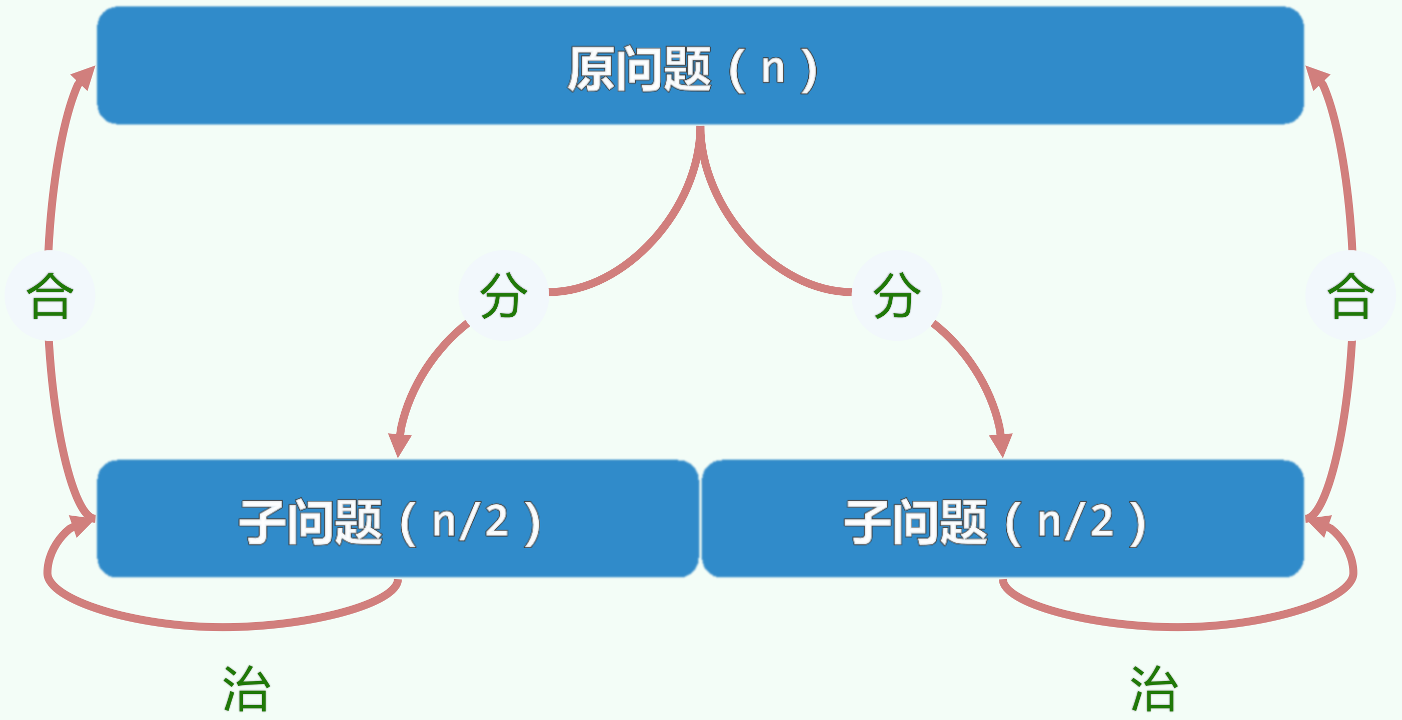
❖ 将其**划分**为若干子问题

（通常两个，且规模**大体相当**）

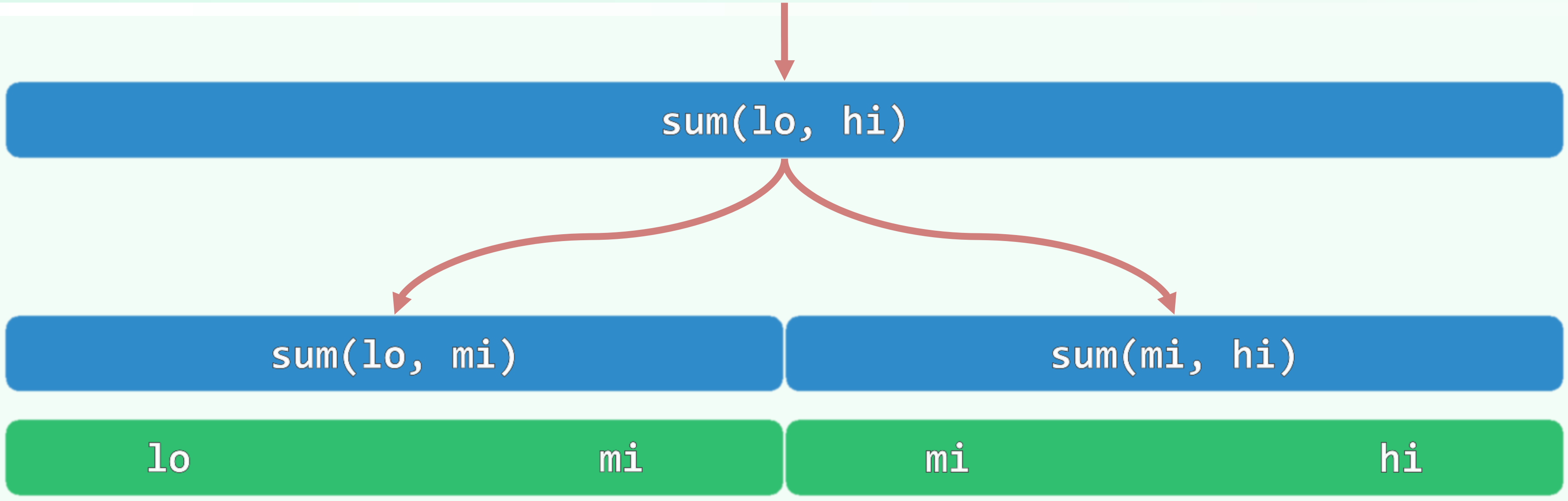
❖ 分别求解子问题

❖ 由子问题的解

合并得到原问题的解

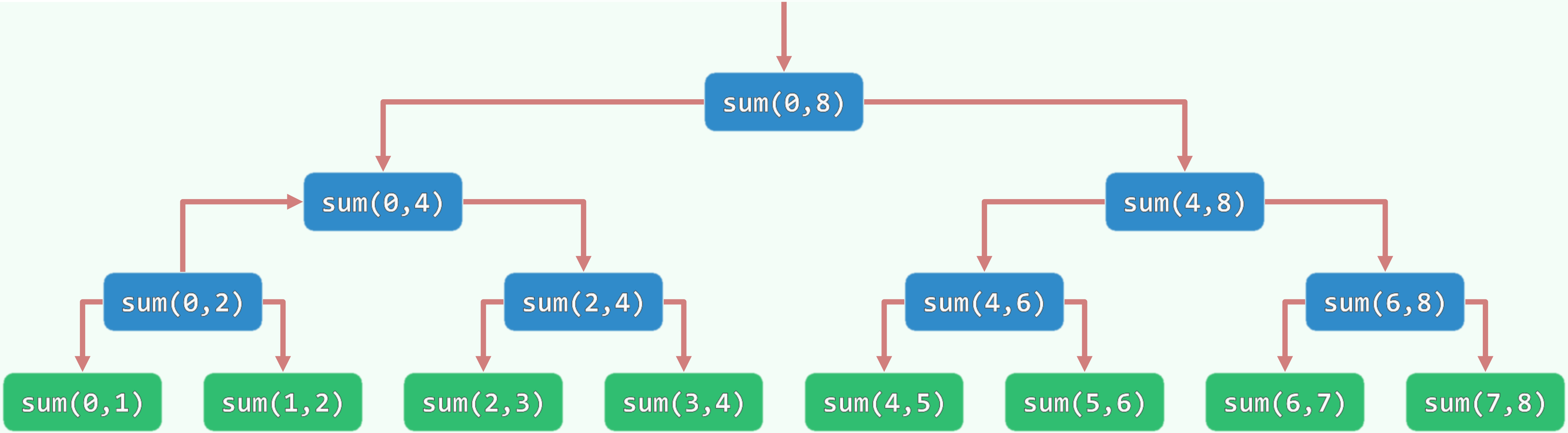


Binary Recursion




```
❖ 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

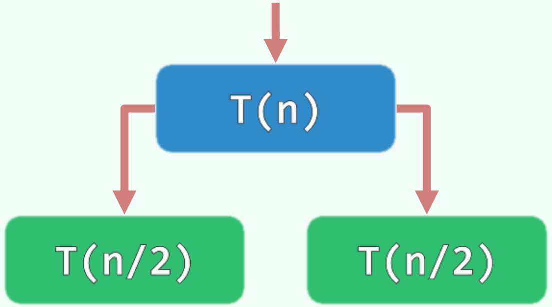
$$= \mathcal{O}(1) \times (2^0 + 2^1 + 2^2 + \cdots + 2^{\log n})$$

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

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



❖ 递推方程：$T(n) \;=\; 2 \cdot T(n/2) + \mathcal{O}(1)$

$T(1) \;=\; \mathcal{O}(1)$ //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

❖ 分治策略对应的递推式通常（尽管不总是）形如：$T(n) \;=\; a \cdot T(n/b) + \mathcal{O}(f(n))$

（原问题被分为a个规模均为n/b的子任务；任务的划分、解的合并耗时f(n)）

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

- kd-search: $T(n) \;=\; 2 \cdot T(n/4) + \mathcal{O}(1) \;=\; \mathcal{O}(\sqrt{n})$

❖ 若 $f(n) = \Theta(n^{\log_b a} \cdot \log^k n)$，则 $T(n) = \Theta(n^{\log_b a} \cdot \log^{k+1} n)$

- 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: $T(n) \;=\; 2 \cdot T(n/2) + \mathcal{O}(n \cdot \log n) \;=\; \mathcal{O}(n \cdot \log^2 n)$

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

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