01-C3

绪论

渐近复杂度：指数

慌得那拿盘的小怪，战兢兢跑去报道：“难，难，难！难，难，难！”
老妖道：“怎么有许多难？”

“你是什么东西？”太太说。四虎子也楞住了，他自己
不知道他是什么东西——这本是世上最难答的一个问题

邓俊辉
deng@tsinghua.edu.cn

$\mathcal{O}(2^n)$: exponential

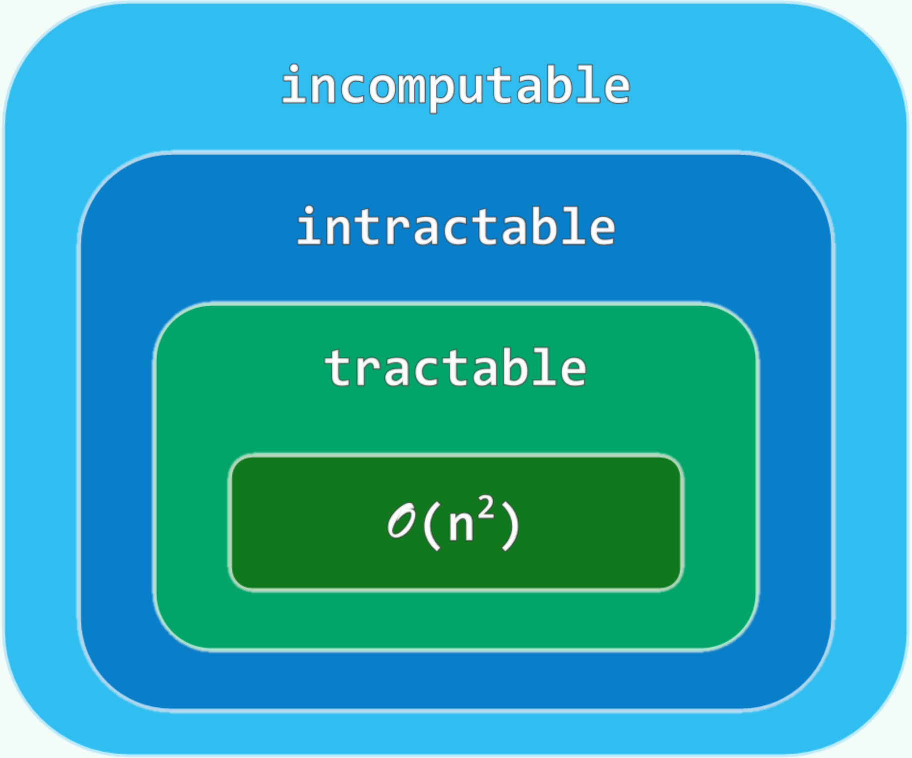
❖ 指数： $T(n) = \mathcal{O}(a^n), \quad a > 1$

$$\because \; e^n \; = \; 1 + n + n^2/2! + n^3/3! + n^4/4! + \ldots$$

$$\therefore \forall\, c > 1, \; n^c = \mathcal{O}(2^n)$$

$$n^{1000\ldots01} = \mathcal{O}(1.000\ldots01^n) = \mathcal{O}(2^n)$$

$$1.000\ldots01^n = \Omega(n^{1000\ldots01})$$



❖ 这类算法的计算成本增长极快，通常被认为**不可忍受**

❖ 从 $\mathcal{O}(n^c)$ 到 $\mathcal{O}(2^n)$，是从**有效算法**到**无效算法**的分水岭

❖ $\mathcal{O}(2^n)$ 算法往往显而易见，然而设计出 $\mathcal{O}(n^c)$ 算法却**极其不易**，有时甚至注定是**徒劳无功**

❖ 更糟糕的是，这类问题要远比我们想象的**多**得多...

SubsetSum：问题

$$\forall\, S = \{\, a_1,\, a_2,\, \dots,\, a_n \,\} \subset \mathbb{Z}^+$$

$$0 \le t \le s = \sum_{k=1}^{n} a_k$$

$$\exists\, T \subseteq S \ \text{ s.t. } \sum_{a \in T} a = t\ ?$$

❖ 从那堆石头里，曹冲真能挑出几块 //s

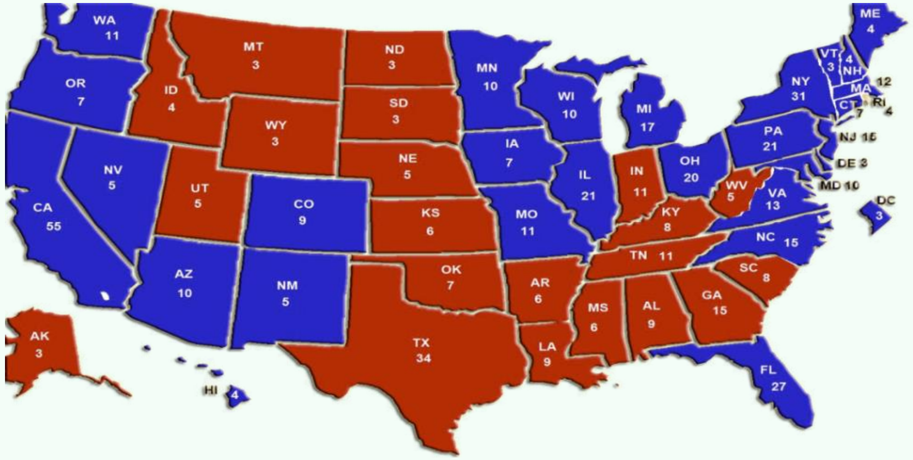
恰好与大象一样重？ //t

❖ 选举人团投票制：51个选区，共538票 //n, 2t

若仅两位候选人，会否恰好各得269票？ //t

可视作SubsetSum的特例：$s = \sum_{k=1}^{n} a_k = 2t$

55	California	11	Indiana	7	Connecticut	4	Idaho
34	Texas	11	Missouri	7	Iowa	4	Maine
31	New York	11	Tennessee	7	Oklahoma	4	New Hampshire
27	Florida	11	Washington	7	Oregon	4	Rhode Island
21	Illinois	10	Arizona	6	Arkansas	3	Alaska
21	Pennsylvania	10	Maryland	6	Kansas	3	Delaware
20	Ohio	10	Minnesota	6	Mississippi	3	D. C.
17	Michigan	10	Wisconsin	5	Nebraska	3	Montana
15	Georgia	9	Alabama	5	Nevada	3	North Dakota
15	New Jersey	9	Colorado	5	New Mexico	3	South Dakota
15	North Carolina	9	Louisiana	5	Utah	3	Vermont
13	Virginia	8	Kentucky	5	West Virginia	3	Wyoming
12	Massachusetts	8	South Carolina	4	Hawaii		538 = Σ



SubsetSum: ~~算法~~ 程序

❖ 直觉上，似乎并不难：

逐一枚举S的每一子集，分别统计总和并核对

❖ sSum(S = {a_1, a_2, ..., a_n}, t)

  if (t == 0) return true;

  if (n == 0) return false;

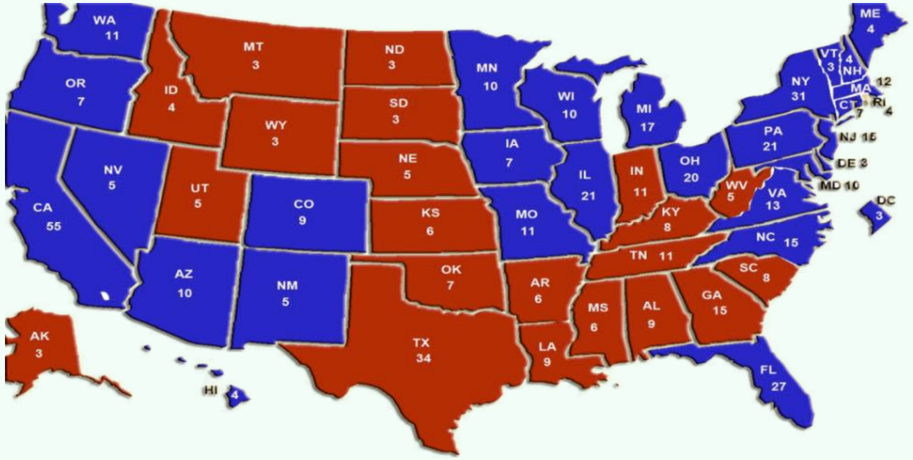
  S = S\{a_n}; //classification

  return sSum(S, t) || sSum(S, t-a_n);

❖ 最坏情况下，需要检视每一个子集，然而...

$$\left|2^S\right| = 2^{|S|} = 2^n$$

55	California	11	Indiana	7	Connecticut	4	Idaho
34	Texas	11	Missouri	7	Iowa	4	Maine
31	New York	11	Tennessee	7	Oklahoma	4	New Hampshire
27	Florida	11	Washington	7	Oregon	4	Rhode Island
21	Illinois	10	Arizona	6	Arkansas	3	Alaska
21	Pennsylvania	10	Maryland	6	Kansas	3	Delaware
20	Ohio	10	Minnesota	6	Mississippi	3	D. C.
17	Michigan	10	Wisconsin	5	Nebraska	3	Montana
15	Georgia	9	Alabama	5	Nevada	3	North Dakota
15	New Jersey	9	Colorado	5	New Mexico	3	South Dakota
15	North Carolina	9	Louisiana	5	Utah	3	Vermont
13	Virginia	8	Kentucky	5	West Virginia	3	Wyoming
12	Massachusetts	8	South Carolina	4	Hawaii	538 = $\sum$	



SubsetSum: NPC

❖ 故严格地讲，这类方法只能算作**程序**，而非**算法**

❖ 还是直觉：应该有更好的办法吧？

比如...转化为...背包问题...？

很遗憾，这里对于整数的**取值范围**未作任何假定

❖ SubsetSum is **NP-complete** —— 什么意思？

❖ 意即：就目前的计算模型而言

不存在可在多项式时间内解决此问题的算法

上述的直觉算法，居然就是最优的

55	California	11	Indiana	7	Connecticut	4	Idaho
34	Texas	11	Missouri	7	Iowa	4	Maine
31	New York	11	Tennessee	7	Oklahoma	4	New Hampshire
27	Florida	11	Washington	7	Oregon	4	Rhode Island
21	Illinois	10	Arizona	6	Arkansas	3	Alaska
21	Pennsylvania	10	Maryland	6	Kansas	3	Delaware
20	Ohio	10	Minnesota	6	Mississippi	3	D. C.
17	Michigan	10	Wisconsin	5	Nebraska	3	Montana
15	Georgia	9	Alabama	5	Nevada	3	North Dakota
15	New Jersey	9	Colorado	5	New Mexico	3	South Dakota
15	North Carolina	9	Louisiana	5	Utah	3	Vermont
13	Virginia	8	Kentucky	5	West Virginia	3	Wyoming
12	Massachusetts	8	South Carolina	4	Hawaii	538 = Σ