串

Karp-Rabin算法：串即是数

13-F1

All things are numbers.

- Pythagoras (570 ~ 495 BC)

God made the integers; all else is the work of man.

- L. Kronecker (1823 ~ 1891)

邓俊辉

deng@tsinghua.edu.cn

凡物皆数：Gödel Numbering


Kurt Gödel 1906-1978

- 逻辑系统的符号、表达式、公式、命题、定理、公理等

  均可以不同的自然数来标识

- 每个有限维的自然数向量（包括字符串），都唯一对应于某个自然数

- 素数序列：p(k) = 第k个素数 = 2, 3, 5, 7, 11, 13, 17, 19, ...

$$\langle a_1, a_2, \ldots, a_n \rangle \rightleftharpoons p(1)^{1+a_1} \times p(2)^{1+a_2} \times \cdots \times p(n)^{1+a_n}$$

$$\langle \boxed{3} \quad \boxed{1} \quad \boxed{4} \quad \boxed{1} \quad \boxed{5} \quad \boxed{9} \quad \boxed{2} \quad \boxed{6} \rangle$$

$$2^{\boxed{4}} \times 3^{\boxed{2}} \times 5^{\boxed{5}} \times 7^{\boxed{2}} \times 11^{\boxed{6}} \times 13^{\boxed{10}} \times 17^{\boxed{3}} \times 19^{\boxed{7}}$$

- "godel" = $2^{1+7} \times 3^{1+15} \times 5^{1+4} \times 7^{1+5} \times 11^{1+12}$ = 13986956031066481708794391920000

- 若果真如RAM模型所假设的字长无限，则只需一个寄存器即可...

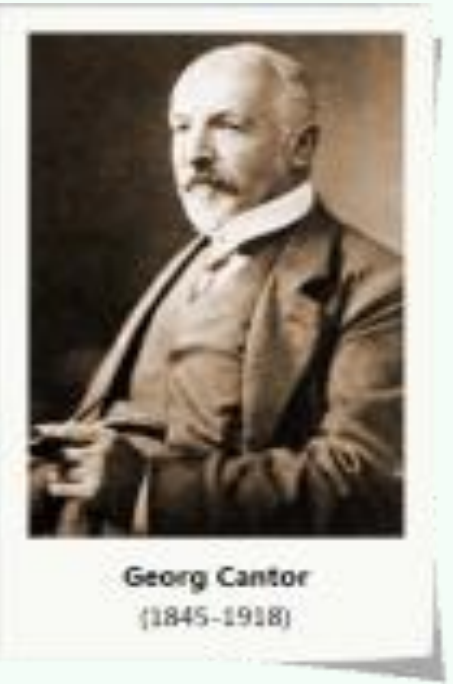
凡物皆数：Cantor Numbering

$$cantor_2(i,j) = [(i+j)^2 + 3\cdot i + j]/2$$

$$cantor_2(2,3) = [(2+3)^2 + 3\cdot 2 + 3]/2 = 17$$

$$cantor_2(3,2) = [(3+2)^2 + 3\cdot 3 + 2]/2 = 18$$

0	1	3	6	10	15
2	4	7	11	16	
5	8	12	17		
9	13	18			
14	19				
20					



$$cantor_{n+1}(\, a_1,\, \ldots; a_{n-1},\, a_n,\, a_{n+1}\,) =$$

$$cantor_n(\, a_1,\, \ldots; a_{n-1},\, cantor_2(a_n,\, a_{n+1}\,)\,)$$

凡物皆数：Cantor Numbering

❖ 长度有限的字符串，都可视作d=1+|Σ|进制的自然数

"decade" = $453145_{(10)}$

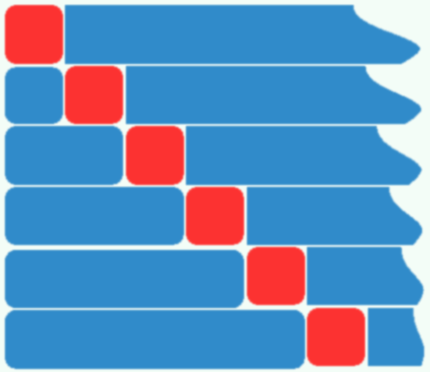
//d = 1 + ('i'-'a') = 10

❖ 长度无限的字符串，都可视作[0,1)内的d进制小数

"bgahbhahbhdei..." = 0.2718281828459...

"fade" = 0.6145 = 0.614499999... = "faddiii..."

❖ Cantor's Diagonal：$\aleph_0 < \aleph_1$



串亦为数

- 十进制串，可直接视作自然数 //指纹（fingerprint），等效于多项式法

  P = "82818"　　T = 271 82818 284590452353602874713527

- 一般地，随意对字符编号{ 0, 1, 2, ..., d - 1 } //设d = |Σ|

  于是，每个字符串都对应于一个 d 进制自然数 //尽管不是单射

  "CAT" = 2 0 19 $_{(26)}$ = $1371_{(10)}$ //Σ = { A, B, C, ..., Z }

  "ABBA" = 0 1 1 0 $_{(26)}$ = $702_{(10)}$

- P在T中出现 仅当 T中某一子串与P相等 //为什么不是“当”？

- 这，不已经就是一个算法了吗？！ //具体如何实现？

- 问题似乎解决得很顺利，果真如此简单吗？ //复杂度？