

向量

有序向量：插值查找

02-D6

邓俊辉

deng@tsinghua.edu.cn

.....那么在最后剩下的一万个猎手中，肯定有人会做出这样的选择：向那个位置开一枪试试...

原理与算法

❖ 假设：已知有序向量中各元素随机分布的**规律**

比如：**独立且均匀**的随机分布

❖ 于是：[lo, hi]内各元素应**大致呈线性趋势增长**

$$\frac{mi - lo}{hi - lo} \approx \frac{e - A[lo]}{A[hi] - A[lo]}$$

❖ 因此：通过**猜测**轴点mi，可以极大地提高收敛速度

$$mi \approx lo + (hi - lo) \cdot \frac{e - A[lo]}{A[hi] - A[lo]}$$

❖ 以英文词典为例：**binary**大致位于**2/26**处

search大致位于**19/26**处

[lo]	0	A	1	[1,53)
	1	B	74	[53,104)
	2	C	158	[104,156)
	3	D	292	[156,208)
	4	E	368	[208,259)
	5	F	409	[259,311)
	6	G	473	[311,363)
	7	H	516	[363,414)
	8	I	562	[414,466)
	9	J	607	[466,518)
	10	K	617	[518,569)
	11	L	628	[569,621)
	12	M	681	[621,673)
	13	N	748	[673,724)
	14	O	771	[724,776)
	15	P	806	[776,827)
	16	Q	915	[827,879)
	17	R	922	[879,931)
	18	S	1002	[931,982)
	19	T	1176	[982,1034)
	20	U	1253	[1034,1086)
	21	V	1271	[1086,1137)
	22	W	1289	[1137,1189)
	23	X	1337	[1189,1241)
	24	Y	1338	[1241,1292)
	25	Z	1341	[1292,1344)
[hi]	26		1344	

实例

❖ 查找目标 : $e = 50$

5	10	12	14	19	23	29	36	39	41	44	51	54	59	72	79	82	86	92
---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

❖ $lo = 0, hi = 18$

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

插值 : $mi = 0 + (18 - 0) * (50 - 5) / (92 - 5) = 9$

比较 : $A[9] = 41 < e$

❖ $lo = 10, hi = 18$

5	10	12	14	19	23	29	36	39	41	44	51	54	59	72	79	82	86	92
---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

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

插值 : $mi = 10 + (18 - 10) * (50 - 44) / (92 - 44) = 11$

比较 : $A[11] = 51 > e$

❖ $lo = hi = 10$

5	10	12	14	19	23	29	36	39	41	44	51	54	59	72	79	82	86	92
---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

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

插值 : $mi = 10$

比较 : $A[10] = 44 < e$, 故返回 : NOT_FOUND

性能

❖ 最坏： $hi - lo = \mathcal{O}(n)$

//具体实例？

❖ 平均：每经一次比较，待查找区间宽度由 n 缩至 \sqrt{n} // [Yao76, PIA78]，习题解析[2-24]

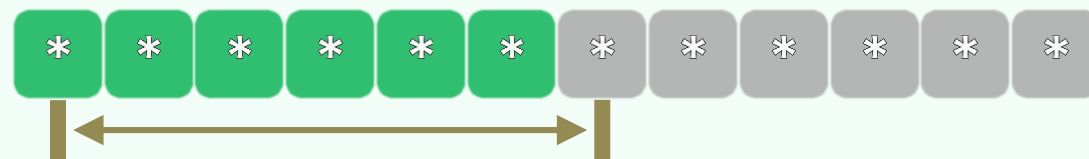
$$n \rightarrow \sqrt{n} \rightarrow \sqrt{\sqrt{n}} \rightarrow \sqrt{\sqrt{\sqrt{n}}} \rightarrow \dots \rightarrow 2$$

$$\underbrace{n \rightarrow n^{1/2^1} \rightarrow n^{1/2^2} \rightarrow n^{1/2^3} \rightarrow \dots \rightarrow 2}_{\mathcal{O}(\log \log n)}$$

❖ 每经一次比较，

待查找区间宽度的数值 n 开方，有效字长 $\log n$ 减半

- 插值查找 = 在字长意义上的折半查找
- 二分查找 = 在字长意义上的顺序查找



综合评价

❖ 从 $O(\log n)$ 到 $O(\log \log n)$ ，优势并不明显

(除非查找表极长，或比较操作成本极高)

比如， $n = 2^{(2^5)} = 2^{32} = 4G$ 时

- $\log_2(n) = 32$

- $\log_2(\log_2(n)) = 5$

❖ 须引入乘法、除法运算

❖ 易受小扰动的干扰和“蒙骗”

❖ 实际可行的方法

- 首先通过插值查找

迅速将查找范围缩小到一定的尺度

- 然后再进行二分查找

进一步缩小范围

- 最后（当数据项只有200~300时）

使用顺序查找