111502544

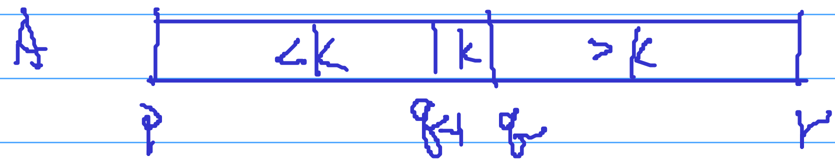
演算法 HW3 資工2B 李崇安

[9.1-1]

Step 1 : find the minimum → $n-1$ times

Step 2 : divide & conquer

let minimum as pivot k



$$\begin{cases} \text{find } A[p:q-1] , < k \\ \text{find } A[q:r] , > k \end{cases}$$

keep doing recursively, every recursion step will divide the length of array half

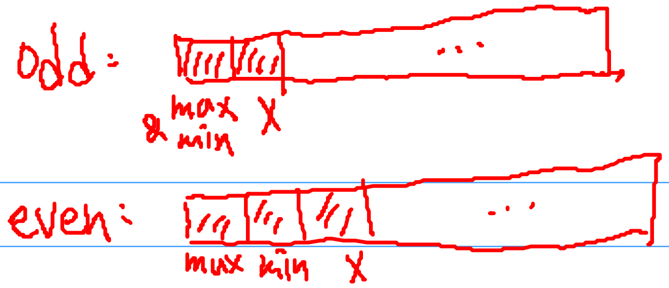
⟹ until find the second smallest, cost $\lceil \lg n \rceil - 1$ (k don't need to compare)

Hence,

$$\text{Total comparison} = n-1 + \lceil \lg n \rceil - 1$$
$$= n + \lceil \lg n \rceil - 2$$

※

[9,1-2]




```
start, max, min, x
if n is odd                                          ⎤
    start=2, max=min=1, x=A[2]                       ⎥
else                                                 ⎥
    if A[1] > A[2]                                   ⎥
        max=A[1], min=A[2]                           ⎥  init
    else                                             ⎥
        max=A[2], min=A[1]                           ⎥
    start=3, x=A[3]                                  ⎦

for i=start to n step 2
    if x == max                                      ⎤  not max
        x = max                                      ⎥     &
    else if x == min                                 ⎥
        x = min                                      ⎦  not min
    if A[i] > A[i+1]
        exchange A[i] with A[i+1]
    if A[i] < min
        min = A[i]
    if A[i+1] > max
        max = A[i]
```

Smallest #comparison $= 5n/2 - 4$ ※

$$\begin{cases} \text{odd}: & 5\lfloor n/2 \rfloor \\ \text{even}: & 5(n-2)/2 + 1 = 5n/2 - 4 \end{cases}$$

[9.2-3]

minimum

A = ⟨2, 3, 0, 5, 7, 9, 1, 8, 6, 4⟩

Worst case : $T(n) = T(n-1) + \Theta(n) \rightarrow \Theta(n^2)$

pivot

⟨2, 3, 0, 5, 7, 9, 1, 8, 6, 4⟩

⟨2, 3, 0, 5, 7, 4, 1, 8, 6, 9⟩

⟨2, 3, 0, 5, 7, 4, 1, 6, 8, 9⟩

⟨2, 3, 0, 5, 6, 4, 1, 7, 8, 9⟩

⟨2, 3, 0, 5, 1, 4, 6, 7, 8, 9⟩

⟨2, 3, 0, 4, 1, 5, 6, 7, 8, 9⟩

⟨2, 3, 0, 1, 4, 5, 6, 7, 8, 9⟩

⟨2, 1, 0, 3, 4, 5, 6, 7, 8, 9⟩

⟨0, 1, 2, 3, 4, 5, 6, 7, 8, 9⟩

total comparison = 7 + 9 + 6 + 6 + 5 + 5 + 3 + 2
= 43

[9.3-3]

```
1. QuickSort (A, p, r)
2.     if p < r
3.     q = Select (A, p, r, (p+r)/2)
4.     q = PARTITION (A, p, r, q)
5.     QuickSort (A, p, q-1)
6.     QuickSort (A, q, r)
```

$\because$ Line 2, q is always the middle

$\therefore$ Quicksort won't be the worst case $\Theta(n^2)$

$$T(n) = \Theta(n) + \Theta(n) + 2T(n/2)$$
$$= 2T(n/2) + \Theta(n)$$

$$n^{\log_n n} = n = n$$

By master theorem,

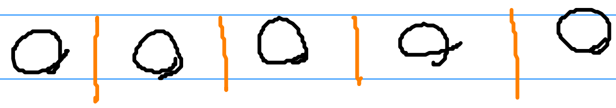
$$T(n) = \Theta(n \lg n)$$

[9.3-5]

```
1  while (r − p + 1) mod 5 ≠ 0
2      for j = p + 1 to r                    // put the minimum into A[p]
3          if A[p] > A[j]
4              exchange A[p] with A[j]
5      // If we want the minimum of A[p : r], we're done.
6      if i == 1
7          return A[p]
8      // Otherwise, we want the (i − 1)st element of A[p + 1 : r].
9      p = p + 1
10     i = i − 1
```

從(p+1)為base，找(i-1)大的

maybe use in 9.1-1 ?

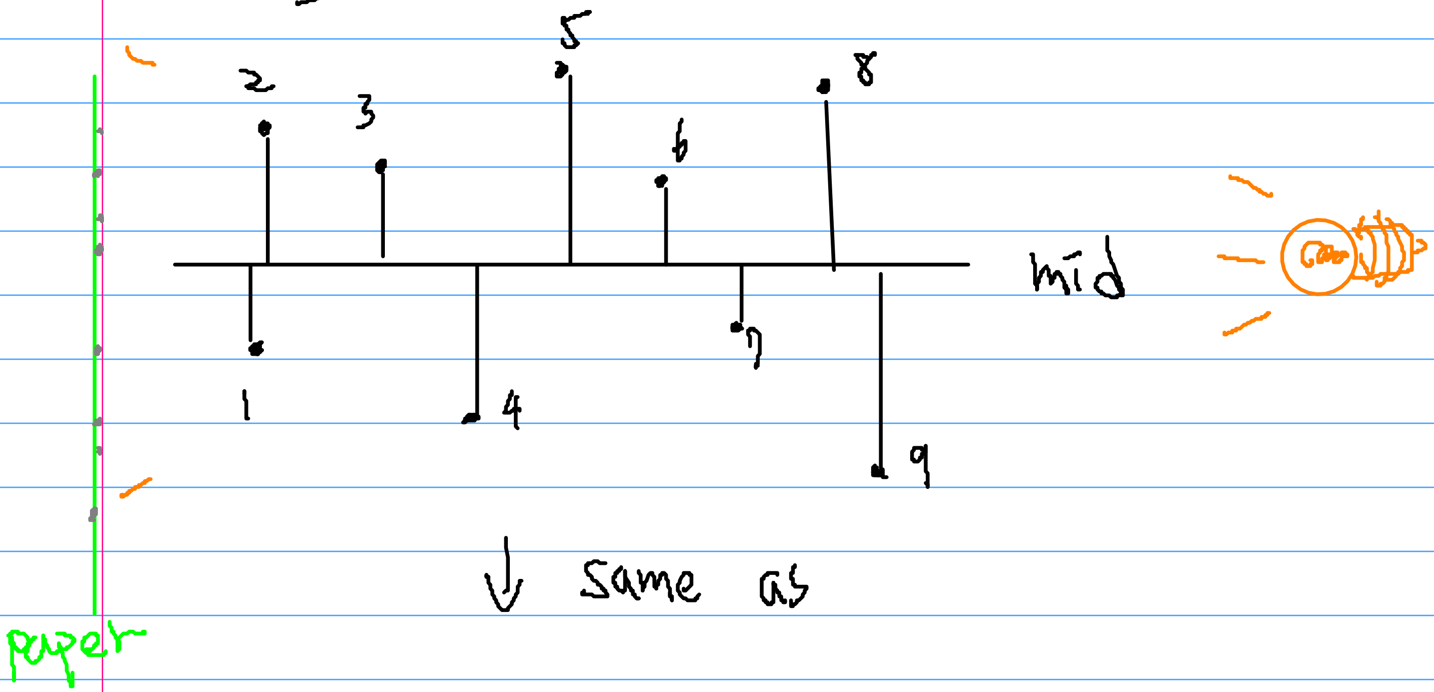


Line 1-10 : put the smallest to A[p]

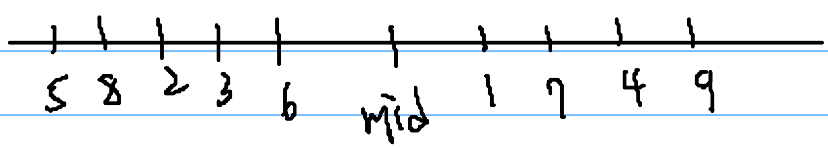
(P, i)	sorted	# comparison (Line 3)
(1, 3)	A[1:1]	3
(2, 2)	A[1:2]	2
(3, 1)	A[1:3]	1

total : 3+2+1 = 6 #

[9,3-7]



↓ same as



find the mid

Solution:

use Select(A,p,r,i) & y-coordinates to determine the mid point in linear time.

Let spot 5 as the basis coordinate (=0), then calculate each spots' coordinate by subtract the basis, |spot i − basis|, {i | i spot exists}.

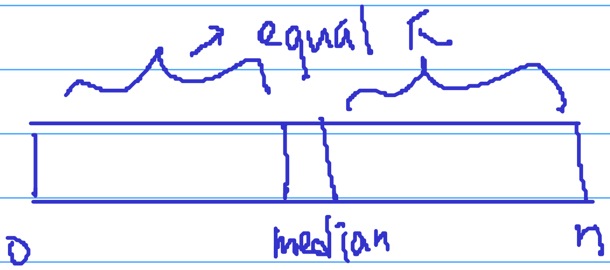
[9.3]

(a)

total weight sum $= \frac{1}{n} \cdot n = 1$

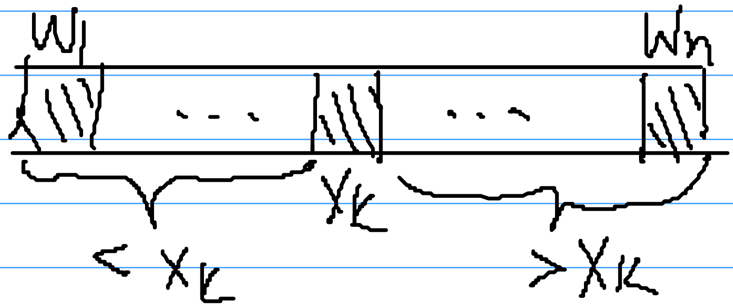
$\because$ each elements has same weight $\frac{1}{n}$

$\therefore$ weight median = median of $x_1, \dots, x_n$



Sorted list

(b)



sorted

Step1 : using quick sort $\Theta(n \lg n)$

Step2 : find the index that $\sum_{x_j < x_k} w_j \geq \frac{1}{2}$.

then $x_k = x_{j-1}$ $\Theta(n)$

$T(n) = \Theta(n \lg n) + \Theta(n) = \Theta(n \lg n)$ #

1. $i = 1$
2. $sum = 0$
3. while $sum \geq \frac{1}{2}$
4. $sum \mathrel{+}= W_i$
5. $i{+}{+}$

(c)

Modify_SELET

(1) Compare: value $\xrightarrow{\text{instead}}$ weight
weight $\longrightarrow$ $1/n$

$\because$ By (a), we know the weight median equals to median when $W_i = 1/n$

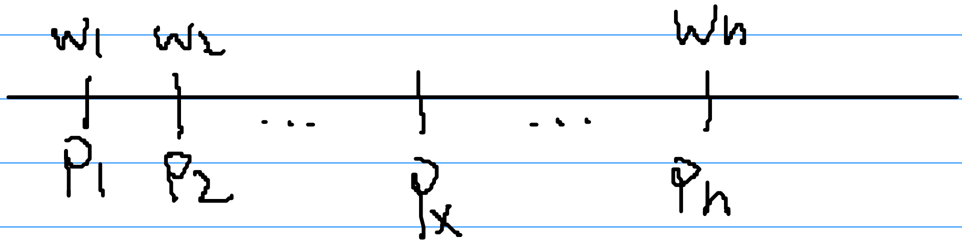
$\therefore$ We just need to find the median of $\{0, 1, \dots, n\}$ using Modify-SELET,

Modify_SELET $(A, p, r, \lfloor n/2 \rfloor)$

Finally, the running time in $\Theta(n)$ in worst case.

$$\sum_{i=1}^{n} w_i\, d(p, p_i)$$

(d)



(1) $x_i < x_k$

$$W_1 \cdot d(P_1, P_x) + W_2 \cdot d(P_2, P_x) + \cdots + W_{x-1} \cdot d(P_{x-1}, P_x)$$

$$\le W_1 \cdot d(P_1, P_x) + W_2 \cdot d(P_1, P_x) + \cdots + W_{x-1} \cdot d(P_1, P_x)$$

$$= \underbrace{(W_1 + W_2 + \cdots + W_{x-1})} \cdot d(P_1, P_x)$$

$$< \frac{1}{2} \cdot d(P_1, P_x) \qquad \sum_{x_i < x_k} W_i < \frac{1}{2}$$

(2) $x_i > x_k$

$$W_{x+1} \cdot d(P_{x+1}, P_x) + W_{x+2} \cdot d(P_{x+2}, P_x) + \cdots + W_n \cdot d(P_n, P_x)$$

$$\le W_{x+1} \cdot d(P_n, P_x) + W_{x+2} \cdot d(P_n, P_x) + \cdots + W_n \cdot d(P_n, P_x)$$

$$= \underbrace{(W_{x+1} + W_{x+2} + \cdots + W_n)} \cdot d(P_n, P_x)$$

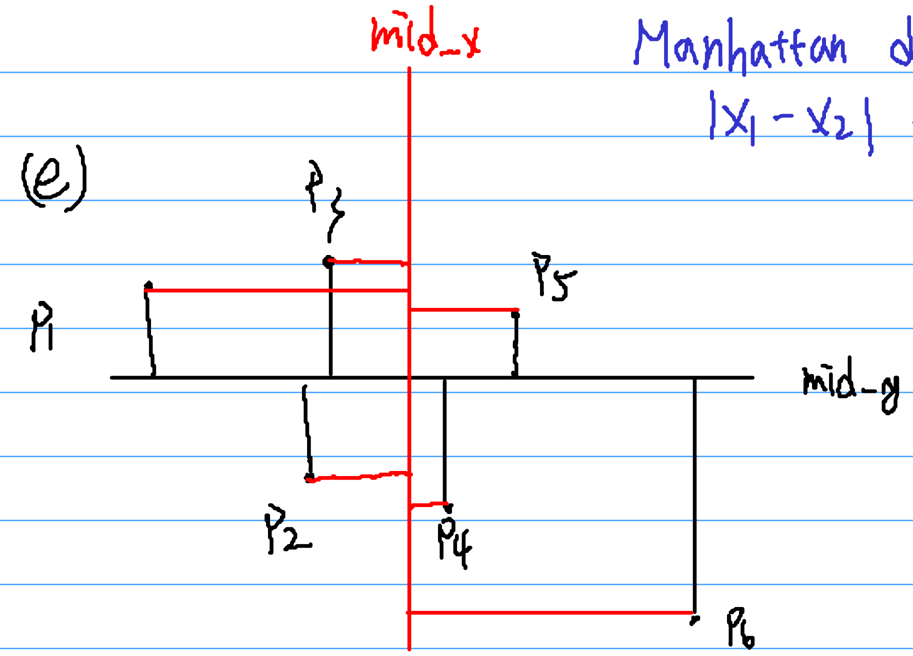
$$\le \frac{1}{2} \cdot d(P_n, P_x) \qquad \sum_{x_i > x_k} W_i \le \frac{1}{2}$$

Conclusion:

If $d(P_1, P_x) \approx d(P_n, P_x)$, then $\sum_{i=1}^{n} W_i \cdot d(P_x, P_i)$

$\approx 2 \cdot \left[\frac{1}{2} \cdot d(P_1, P_x)\right] = d(P_1, P_x)$, which is the best case.

(e)



Manhattan distance $= |x_1 - x_2| + |y_1 - y_2|$

Best solution:

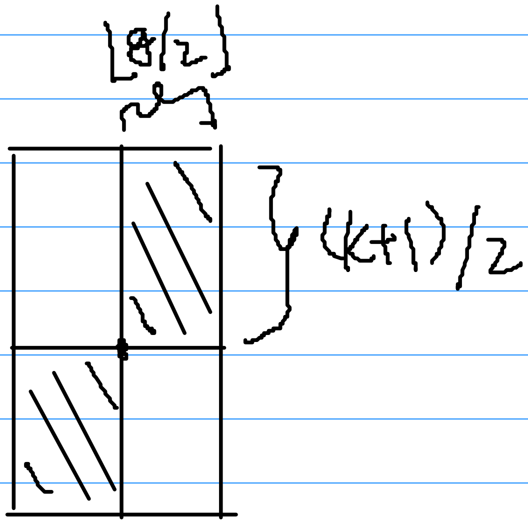
find mid_x, mid_y, then

$$\sum_{i=1, j=1}^{n} |x_i - mid\_x| + |y_i - mid\_y|$$

will be the smallest. (By the conclusion of (d))

[9.6]

(a)



Assume k is odd,

$g = n/k$

low side $= (k+1)/2 \cdot \lfloor g/2 \rfloor \geq (k+1)g/4$

high side $= (k+1)/2 \cdot \lfloor (g+1)/2 \rfloor \geq (k+1)g/4$

$$kg - (k+1)g/4 = \frac{3k-1}{4} g = \frac{3k-1}{4k} n$$

$$T(n) = T(n/k) + T\left(\frac{3k-1}{4k} n\right) + \Theta(n)$$

c: suitably large

$$\leq c(n/k) + c\left(\frac{3k-1}{4k} n\right) + \Theta(n)$$

$c > 0, \forall n > 0$

$$\leq \frac{3k+3}{4k} cn + \Theta(n)$$

If we want run in linear time, then

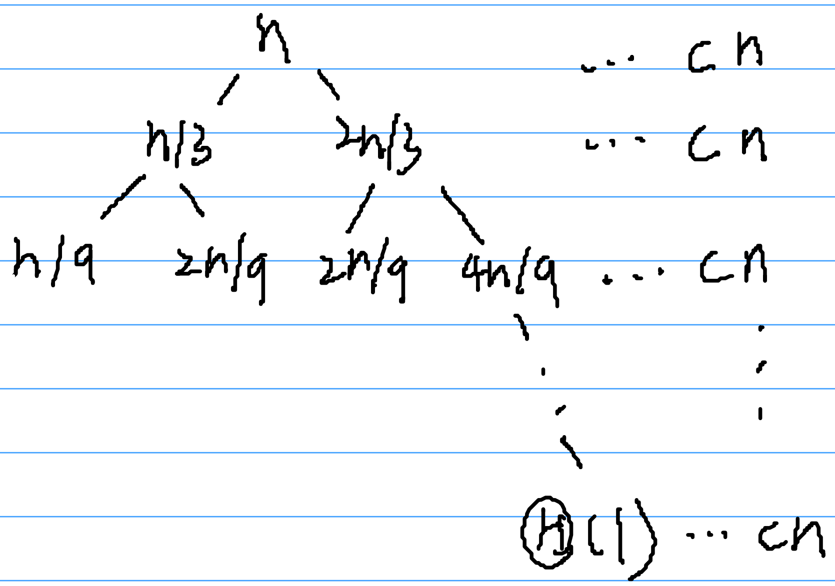
$$\frac{3k+3}{4k} < 1, \quad k > 3$$

\#

(b)

Same steps in (a), when $n=3$, we get

$$T(n) = T(n/3) + T(2n/3) + \Theta(n)$$



$$n/(3/2)^k = 1, \quad k = \lg_{3/2} n$$

$$T(n) = cn \cdot \lg_{3/2} n$$

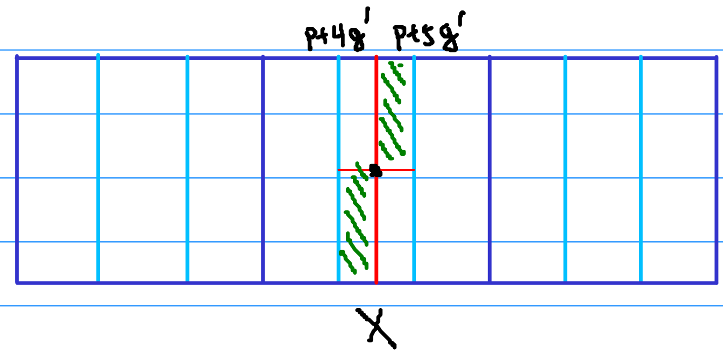
$$= O(n \lg n)$$

#

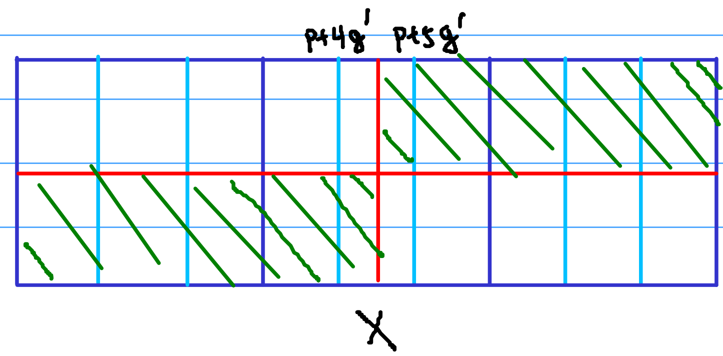
(c)

By (b), time complexity is $O(n\lg n)$ when $n=3$. In order to reach $O(n)$, we need to let $n>3$. And the fastest way to reach it is multipy by 3, $n' = 3\times3 = 9$ (odd).

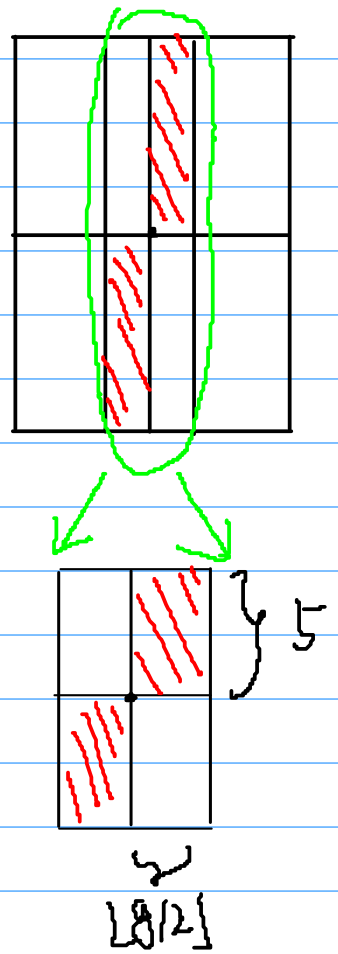
- Line 1 : change to mod 9 ($\because$ 9 groups)
- Line 2-13 : same
- Line 15-17 : 3 groups $\rightarrow$ 9 groups
- Line 20 : find x = middle of $A[p+4g' : p+5g'-1]$



- Line 21-28 : partition to find the ith largest



(d)



$g = n/9$

low side = high side $\geq 5g/2$

$9g - 5g/2 = 13g/2 \geq 13n/18$

$T(n) = T(n/9) + T(13n/18) + \Theta(n)$

c: suitably large

$c > 0, \forall n > 0$

$\leq c(n/9) + c(13n/18) + \Theta(n)$

$= \frac{15}{18} cn + \Theta(cn)$

$= cn - \frac{3}{18} cn + \Theta(cn)$

$= \Theta(cn)$ #

[11.1-1]

```
max = T[1]
for i = 2 to m
    if (T[i] != NIL) & (T[i] > max)
        max = T[i]
```

Worst case: $\Theta(m)$ #

[11.1-2]

Use Direct-Address

Let the hash table size = m

hash function: h

① $h: \{0, 1, \dots, m-1\} \rightarrow \{0, 1, \dots, m-1\}$

② $\begin{cases} 0, & NIL \\ 1, & \text{occupied} \end{cases}$

Once the condition satisfied ① & ②,

we can make sure dictionary operations will run in $O(1)$ time.

[11.2-3]

(i) successful search、unsuccessful search

Result: faster

Reason:

∵ list is sorted

∴ use binary search $\Theta(\lg n)$

(ii) insert、deletion

Result: slower

Reason:

∵ need to find the correct position to maintain sorted order

∴ take much more time

[11.2-5]

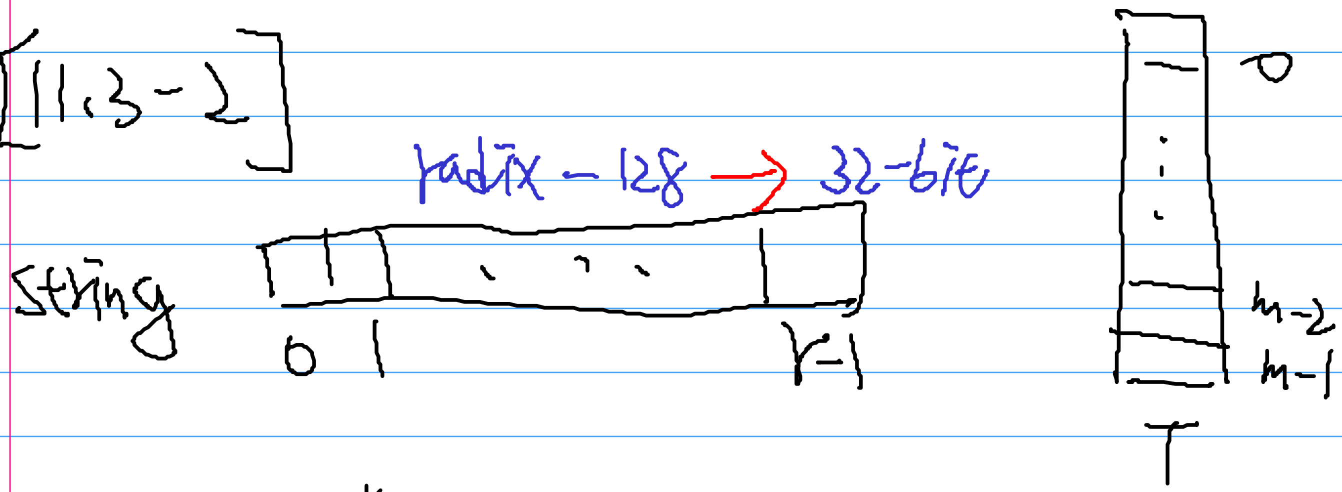
By thm. 11.2, we know

successful search take $\Theta(1+\alpha)$

$$\Theta(1+\alpha) = \Theta\left(1+\frac{|U|}{m}\right) > \Theta\left(1+\frac{(n-1)m}{m}\right) = \Theta(n)$$

#

[11.3-2]



$h(k) = (k * 128) \bmod m \;, \quad 0 \le k < r$

[11.3-3]

Let x & y be two strings that can be converted to each other by permuting their characters

$\because$ order of characters in x & y won't effect the result of division

$\therefore$ x & y will collide in same slot

Not desirable example: Bank-password hashing

Goal: small change → significant different hash value

⇒ prevent attacker trying different permutations

[11-3-4]

k	$h(k)$
61	$\lfloor 1000(61\cdot(\sqrt{5}-1)/2 \bmod 1)\rfloor = 754$
62	$\lfloor 1000(62\cdot(\sqrt{5}-1)/2 \bmod 1)\rfloor = 766$
63	$\lfloor 1000(63\cdot(\sqrt{5}-1)/2 \bmod 1)\rfloor = 778$
64	$\lfloor 1000(64\cdot(\sqrt{5}-1)/2 \bmod 1)\rfloor = 791$
65	$\lfloor 1000(65\cdot(\sqrt{5}-1)/2 \bmod 1)\rfloor = 803$

[11.4-1]

key = {10, 22, 31, 4, 15, 28, 17, 88, 59}

m = 11

(1) Linear probe : $h(k,i) = (k+i) \bmod m$

k	i	h(k,i)
10	1	$(10+1) \bmod 11 = 0$
22	1	$(22+1) \bmod 11 = 1$
31	1	$(31+1) \bmod 11 = 10$
4	1	$(4+1) \bmod 11 = 5$
15	1	$(15+1) \bmod 11 = 5$
28	1	$(28+1) \bmod 11 = 7$
17	1	$(17+1) \bmod 11 = 7$
88	1	$(88+1) \bmod 11 = 1$
59	1	$(59+1) \bmod 11 = 5$

key	slot
10	0
22	1
88	2
	3
	4
4	5
15	6
28	7
17	8
59	9
31	10

collide

15 : slot 5 $\xrightarrow{+1}$ slot 6

17 : slot 7 $\xrightarrow{+1}$ slot 8

88 : slot 1 $\xrightarrow{+1}$ slot 2

59 : slot 5 $\xrightarrow{+1}$ slot 6 $\xrightarrow{+1}$ slot 7 $\xrightarrow{+1}$
slot 8 $\xrightarrow{+1}$ slot 9

(2) double hashing: $h_1(k) = k$

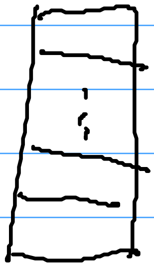
$h_2(k) = 1 + (k \bmod (m-1))$

$h_1(k)+h_2(k)$	k	$h_1(k)$	$h_2(k)$
10	10	10	$1+(10\%10) = 2 \to 0$
9	22	$22 \to 9$	$1+(22\%10) = 3 \to 0$
8	31	$31 \to 8$	$1+(31\%10) = 2 \to 0$
7	4	4	$1+(4\%10) = 5 \to 3$
6	15	$15 \to 6$	$1+(15\%10) = 6 \to 0$
5	28	$28 \to 5$	$1+(15\%10) = 6 \to 0$
4	17	$17 \to 4$	$1+(17\%10) = 8 \to 0$
3	88	$88 \to 3$	$1+(88\%10) = 9 \to 0$
2	59	$59 \to 2$	$1+(59\%10) = 6 \to 0$

	0
	1
59	2
88	3
17	4
28	5
15	6
4	7
31	8
22	9
10	10

[11.4-4]

$$H_m = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{m}$$



$n = m$

全部都會 Hash 到

(search)

$$\Pr\{X\} = \frac{n}{m} \cdot \frac{n-1}{m-1} \cdots 2 \cdot 1$$

$$E[X] = \frac{1}{n} \cdot \sum_{i=1}^{n} \Pr\{X\}$$

$$= \frac{1}{n} \cdot \sum_{i=1}^{n} \left(\frac{n}{m} \cdot \frac{n-1}{m-1} \cdots \frac{2}{i} \cdot \frac{1}{i} \right)$$

$$= \frac{1}{n} \cdot \sum_{i=1}^{n} \left(\prod_{j=1}^{n} j \Big/ \prod_{j=1}^{m} j \right)$$

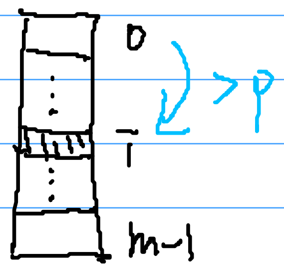
$$= \frac{1}{n} \cdot \sum_{i=1}^{n} \frac{n!}{m!} = \sum_{i=1}^{n} \frac{(n-1)!}{m!}$$

$\because n = m$

$$\therefore \sum_{i=1}^{m} \frac{(m-1)!}{m!} = \sum_{i=1}^{m} \frac{1}{m} = H_m \quad \#$$

[11.1]

(a)



前面至少有p個已 inserted

$\therefore \Pr\{X_i > p\}$

$= \frac{n}{m} \cdot \frac{n-1}{m-1} \cdots \frac{n-p}{m-p}$

$\le \left(\frac{n}{m}\right)^p \quad (\because n \le m/2)$

$\le \left(\frac{m/2}{m}\right)^p$

$= 2^{-p}$

(b)

$p = 2\lg n$ 代入 (a) 的結論

$$\Pr\{X_i > 2\lg n\} \le 2^{-2\lg n} = n^{-2} = O(1/n^2)$$

(c)

$\because$ # insertion & search maximum are independent

$\therefore \Pr\{X > 2\lg z\}$ = sum of each search maximum

By (b),

$$\Pr\{X > 2\lg z\} = n \cdot O(1/n^2) = O(1/n)$$

(d) by (c),

$$E[X] = \sum \Pr\{X > 2\lg n\} \cdot 2\lg n$$

$$= \underbrace{\frac{1}{n} 2\lg n + \cdots + \frac{1}{n} 2\lg n}_{n}$$

$$= \frac{1}{n} \cdot n \cdot 2\lg n$$

$$= 2\lg n = O(\lg n)$$ #

[11-2]

(a)

(1) Build the BST (ensure height = $O(\lg n)$)

$\rightarrow \Theta(n \lg n)$

(2) Worst case:

SEARCH Operation : $\Theta(n \lg n$

(b) By thm. 11.7 we know,

$$\{\#\text{ probe of unsuccessful search}\} \le \frac{1}{1-\alpha}$$

$$\frac{1}{1-\alpha} = \lg n$$

$$\Leftrightarrow \frac{1}{1-n/m} = \lg n$$

$$\Leftrightarrow \lg n - \frac{n}{m}\lg n = 1$$

$$\Leftrightarrow \frac{n}{m}\lg n = \lg n - 1$$

$$\Leftrightarrow m = \frac{n \cdot \lg n}{\lg n - 1}$$

$$O(m-n) = O\left(n - \frac{n \cdot \lg n}{\lg n - 1}\right)$$

$$= O\left(n\left(1 - \frac{\lg n}{\lg n - 1}\right)\right)$$

$$= O\left(n \cdot \left(1 - 1 - \frac{1}{\lg n - 1}\right)\right)$$

$$= O\left(\frac{-n}{\lg n - 1}\right) = O\left(\frac{n}{\lg n}\right)$$

\#