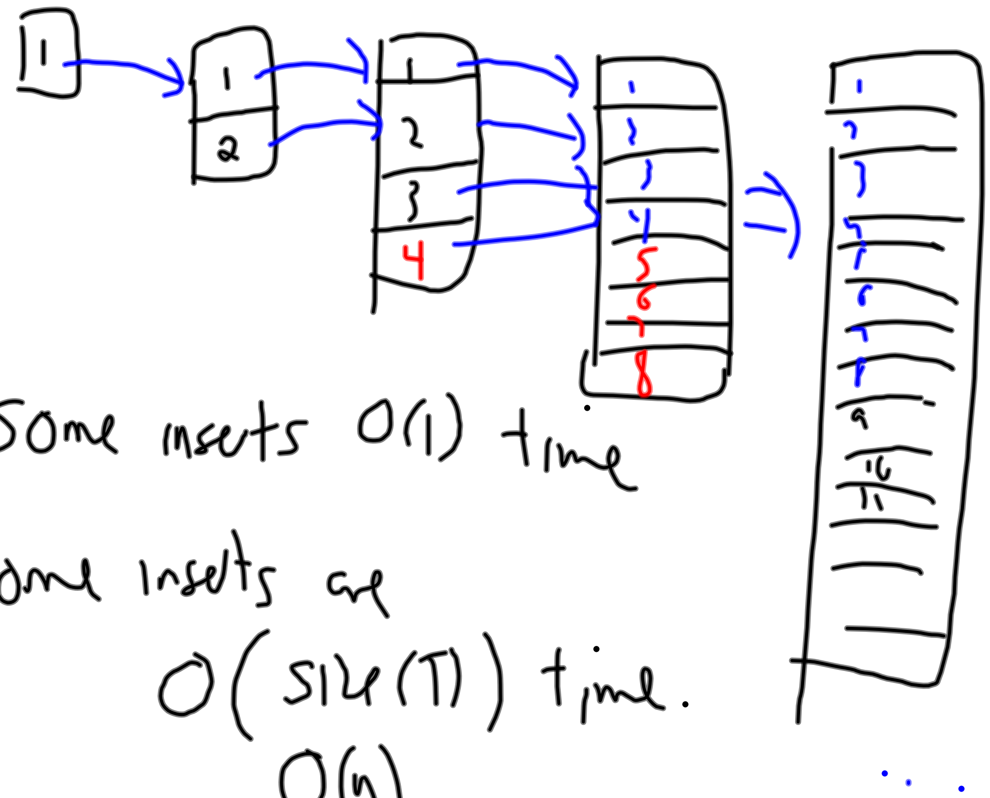
Insert / Delete into a table (array)

How do you fix the size of the array w/o knowing how much data you will see.

table is too small - reallocate
" " " big - wasting memory

reallocation ⇒ copying all data ⇒ time proportional to the data you have

Strategy when full - double size



Some inserts $O(1)$ time

Some inserts are

$O(\text{size}(T))$ time.

$O(n)$

Let c_i = cost i^{th} insert

$$c_i = \begin{cases} i & \text{if } i-1 \text{ is a power of } 2 \\ 1 & \text{o.w.} \end{cases}$$

n inserts

$$\sum_{i=1}^{n} c_i \le \sum_{i=1}^{n} 1 + \sum_{j=0}^{\lg n} 2^j$$

$$\le n + (1 + 2 + 4 + \cdots + n)$$

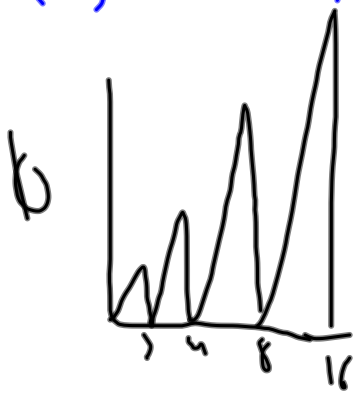
$$\le n + 2n = 3n$$

each inserts is $O(1)$ amortized time.

each insert should build up credit (potential)
potential should be used when doing a reallocation

$$\Phi \quad \begin{cases} num = size/2 & \Phi = 0 \\ num = size & \Phi = num \end{cases}$$

$$\Phi(T) = 2\,num(T) - size(T)$$



$\Phi(T) = 2\,num(T) - size(T)$

Show: $\hat{c}_i = c_i + \Delta\Phi \leq 3$

case 1: table size does not change

$\hat{c}_i = 1 + 2\,num_i - \cancel{size_i} - 2\,num_{i-1} + \cancel{size_{i-1}}$

$size_i = size_{i-1}$

$num_i = num_{i-1} + 1$

$= 1 + 2(num_{i-1} + 1) - 2\,num_{i-1}$

$= 3$

case 2: table size does change

$$\hat{c}_i = c_i + \Delta\phi_i$$

$$= num_i + 2num_i - size_i - 2num_{i-1} + size_{i-1}$$

$$size_i = 2\,size_{i-1}$$
$$num_i = num_{i-1} + 1$$
$$size_{i-1} = num_{i-1}$$

$$= 3(num_{i-1} + 1) - 2size_{i-1} - 2num_{i-1} + size_{i-1}$$

$$= \cancel{num_{i-1}} + 3 - \cancel{size_{i-1}}$$

$$= 3$$

Deletion

Shrink the table if

num << size

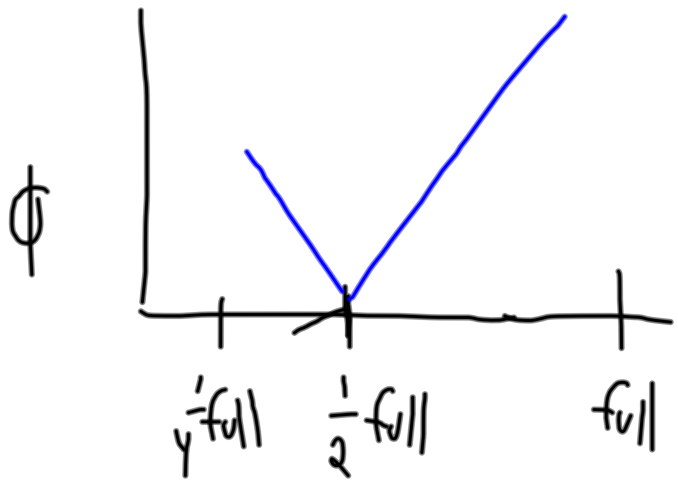
Idea

double size when full

half size when table is ~~half~~ quarter full

alt. ins., del, this makes every

op expensive



$O(1)$ amortized time / op Ins. & Delete.

Disjoint Sets

Items: $X = \{A, B, C, D, E, F, G, H\}$

Maintain sets $S_1 \ldots S_k$

each $x \in X$ is in exactly one set S_i

$S_i \cap S_j = \emptyset \quad \forall i, j, i \neq j$

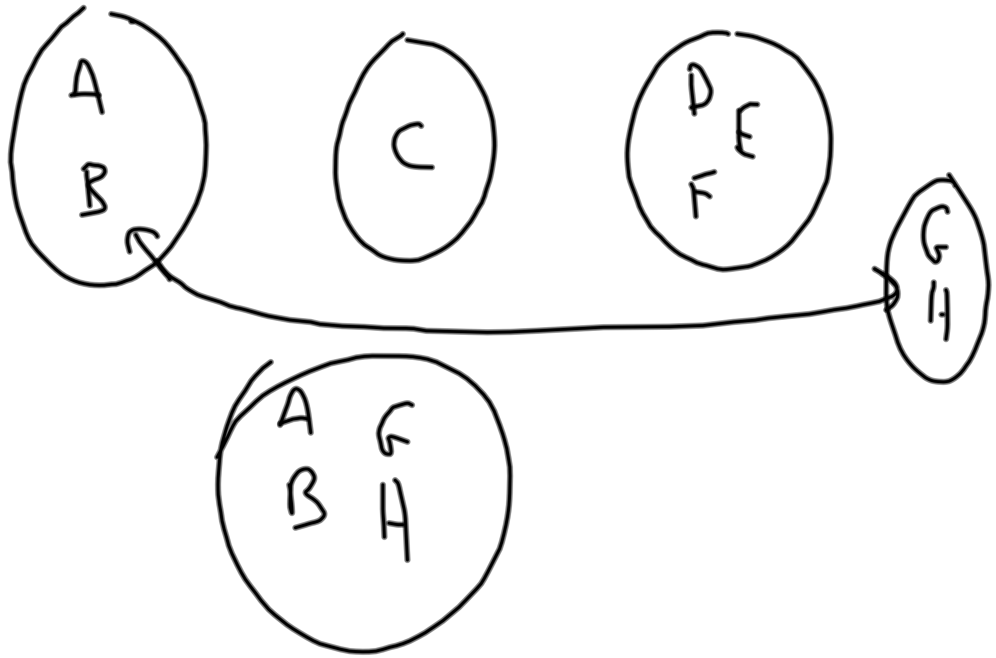
MakeSet(x) - makes a one element set

Find(x) - return the "name" of the set x is in

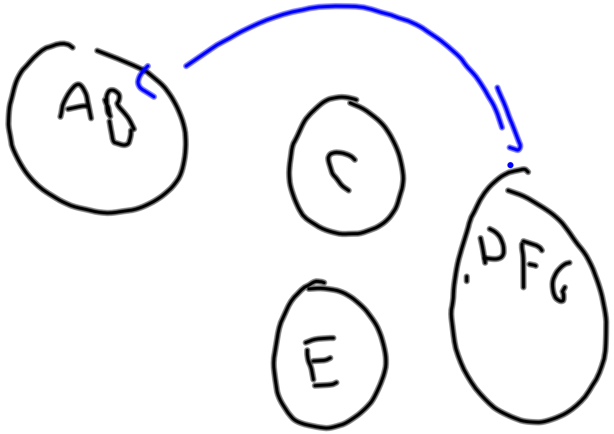
Union(x, y) - merges x's set & y's set

maintain equivalence relation

Network "is connected to"

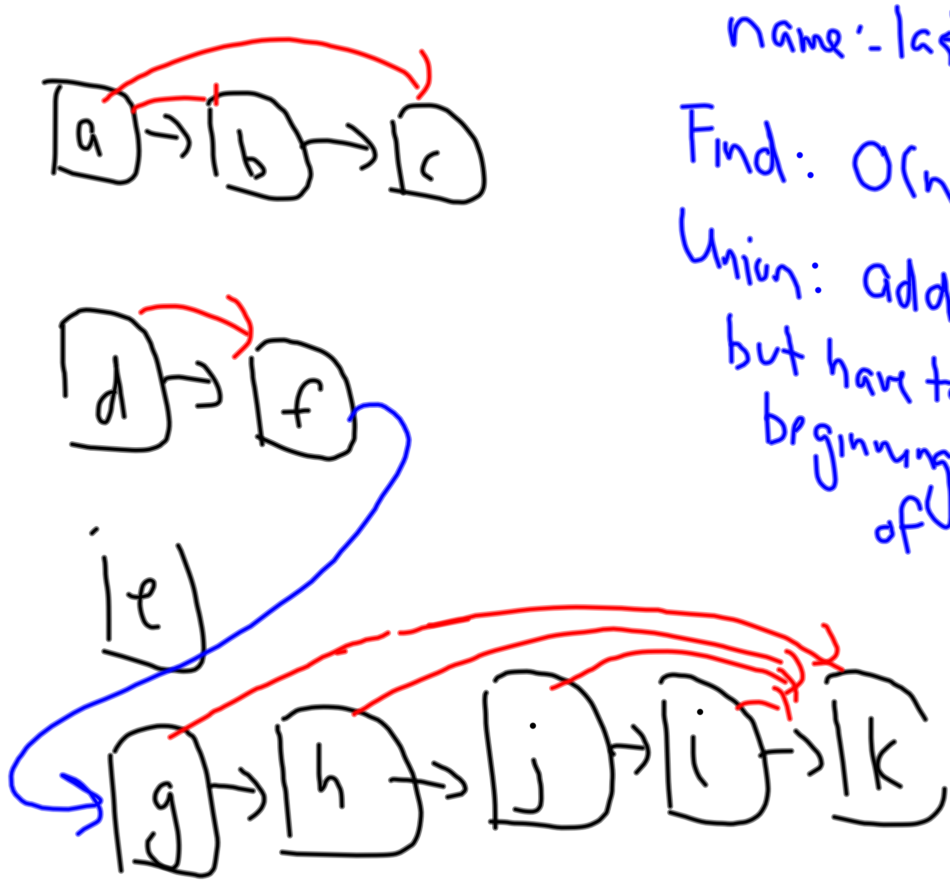


items	name
A	A
B	A
C	C
D	F
E	E
F	F
G	F



Find $O(1)$

Union $O(n)$



name: last elt.

Find: $O(n)$

Union: add 1 ptr.
but have to find beginning & end of list

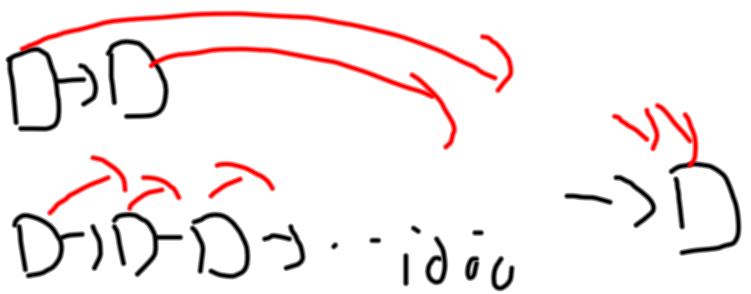
Add ptr to end of list

Find: $O(1)$ Union: $O(n)$

Union by size

(put shorter list before the longer list)

Union: time = |shorter list|



union has $O(\lg n)$ amortized time

smaller set

Union_1 A

Union_2 B

Union_3 $C\ D$

$\vdots$

$B\ C\ D$

$\vdots$

Union_{n-1} (A)

How many times can one element be in the smaller set of a union?

A - B

AB CD

ABCD

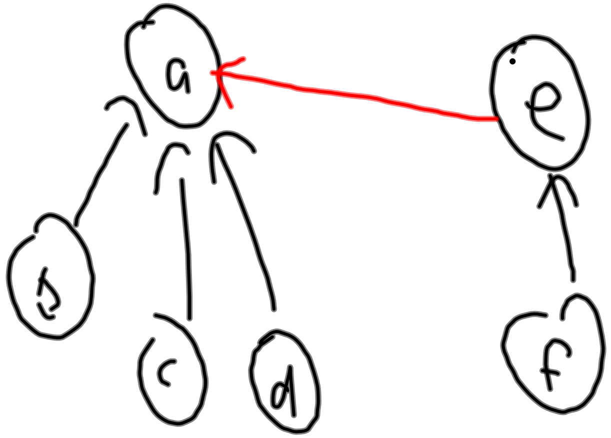
every time an elt. is in the smaller side of a union its set size at least doubles

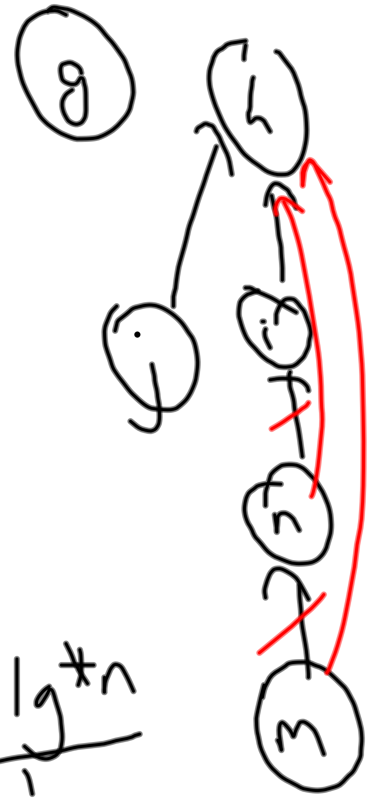
$\Rightarrow$ smaller side $\leq \lg n$ times

Total time for all unions is $O(n \lg n)$

Find $O(1)$

Union $O(\lg n)$





Find: $O(\lg n)$

Union: $O(1)$ + Find

$$O(\lg^* n)$$

n	$\lg^* n$
2	1
2 - 4	2
4 - 16	3
16 - 2^{16}	4
2^{16} - 2^{65536}	5