Lecture 11 Scribbles

Chat Moderators: Junyeob
Eliot

Topics: - Algorithmic Reductions
- Recursions
- Divide & Conquer

What is an algorithm

- a recipe of solving specific problems utilizing
  - primitive instructures
  - set of memory states
  - finite description

Model of Computation: basic computer with assembly

- Unit Cost RAM model
  - basic data types is a integer
  - all it's fit in a word
  - arithmatic operations run $O(1)$ +/−/&/||
  - array allow random access

3 types of problems:

↳ decision problems
↳ search problems
↳ optimization problems

Algorithm Analysis

↳ Correctness
↳ Asymptotic Running Time
↳ Asymptotic Space Usage

Reduction

- reducing a problem into simpler parts

$A[1, \ldots, n]$ uniqueness we want to if $A[i] = A[j]$ for any $i \neq j$

```
for i = 1 → n
  for j = 1 → n
    if A[i] == A[j] & i ≠ j
      return false
return true
```

$O(n^2)$

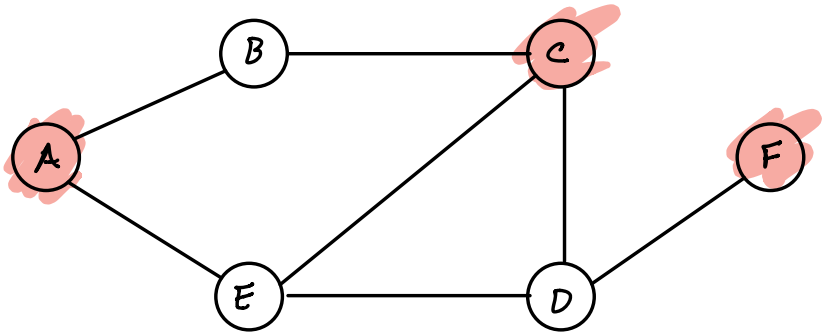
1. hash map $O(n)$

2. Sort A $O(n \log n)$

```
for i = 1 → n
  if Ai == Ai+1   O(n)
    return false
return true
```

Uniqueness problem reduces to sorting problem or hash map problem

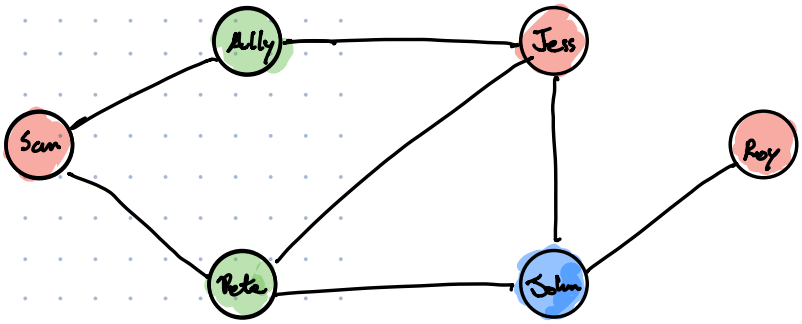
Maximum independent sets



Try to organize carpool

We have six people:

- ◯ John
- ◯ Roy
- ◯ Pete
- ◯ Sam(antha)
- ◯ Jess(ica)
- ◯ Ally



Now this group has a lot of infighting and grudges including:

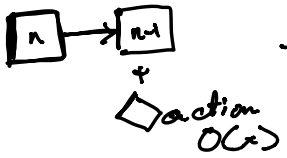
- Ally does not get along with the other girls
- Sam is divorced from Pete
- John borrowed money from Pete and never paid it back
- Jess used to date Pete and John so now things are awkward
- John is mad at Roy for screwing up a Warcraft campaign

Carpool problem reduces to the independent sets problem maximally

```
While (uncolored nodes)
    find ind_sets [B]   O(x)
    color nodes in ind_sets   O(n x)
end
```

Recursion: What is a recursion

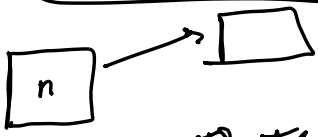
- reducing a problem into a small instance of itself



then the complete is solved

Tower of Hanoi

```
Hanoi (n, src, dest, tmp)
  if (n>0)
    // move n-1 disks to tmp
    Hanoi (n-1, src, tmp, dest)
    // move nth disk from src to dest
    Hanoi (n-1, tmp, dst, src)
```



$T(n) = 2T(n-1) + 1$

If $T(n) = 2^n$

1. Guess

$2^n = 2 \cdot 2^{n-1} + 1$

$2^n = 2^n + 1$

Guess $T(n) 2^n - 1$

$2^n - 1 = 2(2^{n-1} - 1) + 1 = 2 \cdot 2^{n-1} - 2 + 1 = 2^n - 1$

2. Repeated Application

$T(n) = 2T(n-1) + 1 = 2 \cdot 2T(n-2) + 2 + 1$

$= 2 \cdot 2 \cdot 2T(n-3) + 4 + 2 + 1$

$= 2^i T(n-i) + 2^{i-1} + 2^{i-2} + \ldots + 1$

$= 2^{n-1} T(1) + 2^{n-2} + \ldots + 1$

$T(n) = 2^n - 1 = \underbrace{1\,1\,1\,1\,1\,1}_{n-1 \text{ bits}}$

$T(n) = rT(n/c) + f(n)$

3. Characteristic equation and/or annihilators

Divide & Conquer

1. Dividing the problem into smaller parts
2. Taking the results & merging into a larger solution

```
MergeSort (A)
  if (n>1)
    m = n/2
    MergeSort (A[1...m])
    MergeSort (A[m...n])
    Merge (A[1...n], m)  ← O(n)
```

$T(n) = 2T(n/2) + O(n)$ $\qquad O(n \log n)$

$\cdot\, kn$

Quicksort:

What is the recurrence (if $k=1$)

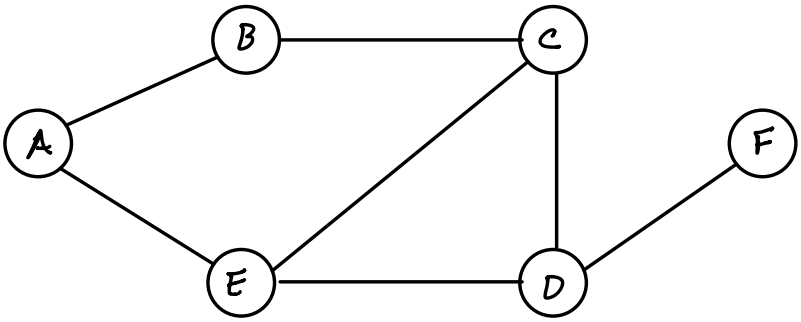
k is the rank of the pivot

$$T(n) = T(k-1) + T(n-k) + O(n)$$

$$T(n) = 2T(n/2) + O(n) = O(n \log n)$$

$$T(n) = T(n-1) + O(n) = O(n^2)$$

Scrap for predrawn figures



We have six people:

- **John**
- **Roy**
- **Pete**
- **Sam(antha)**
- **Jess(ica)**
- **Ally**

Now this group has a lot of infighting and grudges including:

- Ally does not get along with the other girls
- Sam is divorced from Pete
- John borrowed money from Pete and never paid it back
- Jess used to date Pete and John so now things are awkward
- John is mad at Roy for screwing up a Warcraft campaign