Randomized Algorithms

Tutorial 3
Hints for Homework 2

Outline

- Hints for Homework 2
- Randomized Quicksort (Exercise 4.20)
- Michael's Algorithm (optional)[1]
- One of Three (optional) [1]

[1]Probability and Computing, CMU 15-359, Fall 2007.

Hints for Homework 2

Exercise 1

- Find variance in number of fixed points assuming permutation is chosen uniformly at random

- Hint
  - Cannot use linearity to find the variance.
  - Just calculate it directly

Exercise 2

- Weak Law of Large Numbers
  - independent RV $X_1, X_2, \ldots, X_n$
  - Same finite mean μ, finite std-dev σ
- Show

$$\lim_{n\to\infty} \Pr\left(\left| \frac{X_1 + X_2 + \ldots + X_n}{n} - \mu \right| > \varepsilon \right) = 0$$

- Hint
  - Chebyshev's inequality

Exercise 3 (Parameter Estimation)

- Show that

$$\Pr(|\, p - \tilde{p} \,| > \varepsilon p) < \exp\left(\frac{-na\varepsilon^2}{2}\right) + \exp\left(\frac{-na\varepsilon^2}{3}\right)$$

- Show for any δ belongs to (0,1)

$$\text{if } n > \frac{2\ln(2/\delta)}{a\varepsilon^2}, \text{ then } \Pr(|\, p - \tilde{p} \,| > \varepsilon p) < \delta$$

Exercise 4

- Let X_1, X_2, …, X_n be n Poisson trials
- Let a_1, a_2, …, a_n be real in [0,1]
- Let $W = \sum a_i X_i$ and ν =E[W].
- Show that

$$\Pr(W > (1+\delta)\nu) < \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}} \right)^{\nu}$$

- Hint
  - MGF & Markov inequality

Exercise 4

- Somehow, you may need to show this inequality to simplify terms:
- For any $x \in [0,1]$,

$$e^{tx} - 1 \leq x(e^{t} - 1)$$

- Hint: can be proven by calculus

Exercise 5

- Let $X = X_1 + X_2 + \ldots + X_n$,
  each X_i = Geo(0.5)

- Compare X with a sequence of fair coin tosses ➔ Show that

$$\Pr(X > (1+\delta)2n) < \exp\left(\frac{-n\delta^2}{2(1+\delta)}\right)$$

Randomized Quicksort

Quicksort(S) {

1. If $|S| \leqq 1$, return S
2. Else, pick an item x from S
3. Divide S into S_1 and S_2 with
   S_1 = list of all items smaller than x
   S_2 = list of all items greater than x
4. $List_1$ = Quicksort(S_1)
5. $List_2$ = Quicksort(S_2)
6. return $List_1$, x, $List_2$

}

Randomized Quicksort

- In step 2, we choose x by picking an item uniformly at random from S.
- Runtime = expected $O(n \log n)$

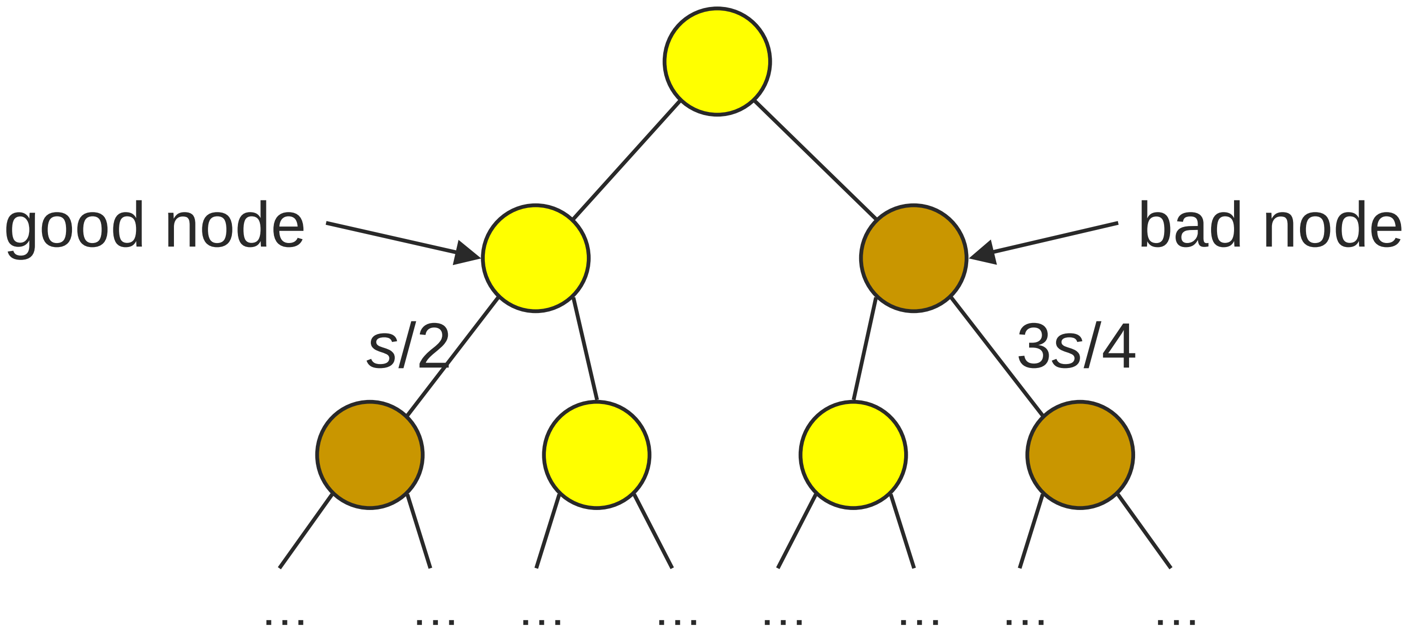
- Can we show it runs in $O(n \log n)$ time with high probability ?

Randomized Quicksort

- Let s = size of the set to be sorted at a particular node

Node:= point which decides on a pivot

Randomized Quicksort



A good node is one whose pivot divides the set into two parts with size of each part not exceeding $2s/3$

Randomized Quicksort

Fact 1: # good nodes in any path is at most $c \log_2 n$, for some c

Proof:

good nodes is at most

$\log n / \log(3/2) = c \log n$

Randomized Quicksort

Fact 2: With probability $\geqq$ 1 - 1/n^2,
nodes in a root-to-leaf path is
at most c' $\log_2 n$, for some c'

Proof:

P := a root-to-leaf path,

l := length of P

B := # bad nodes in P

Randomized Quicksort

Proof (cont.): Now, we know that

$$B \geqq l - c \log n \qquad \text{(why?)}$$

and Pr(bad node) = 2/3

- $\Pr(l > c' \log n)$
  $\leqq \Pr(B > c' \log n - c \log n)$
  $\leqq (2/3)^{c' \log n - c \log n} < n^{-2}$ (for large c')

Randomized Quicksort

Fact 3: With probability $\geqq$ 1-1/n,

nodes in the longest root-to-leaf path is at most $c'\log_2 n$

Proof: Union Bound

Randomized Quicksort

Conclusion:

Runtime of Randomized Quicksort is $O(n \log n)$ with prob at least $1-1/n$

Proof:

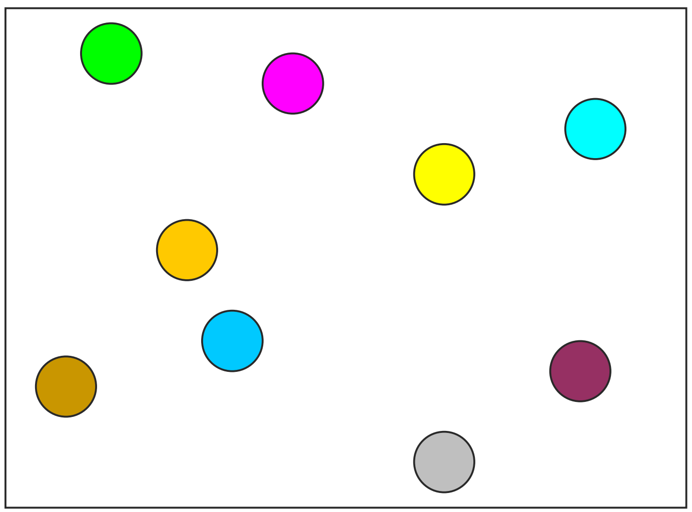
Height of the tree is $O(\log n)$ with probability at least $1-1/n$

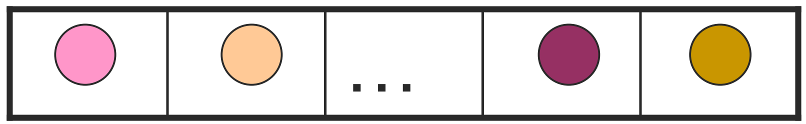
Runtime in this case: $O(n \log n)$

Michael’s Algorithm

Michael's Algorithm

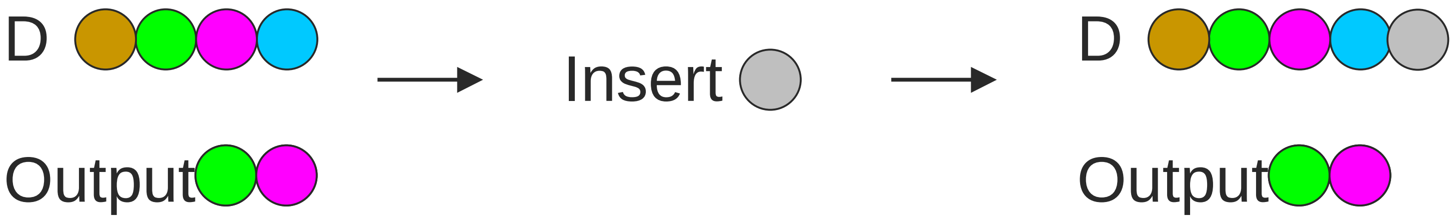
- Input: a set of 2D points
- Determine the closest pair (and its dist)
- Input points are stored in an array





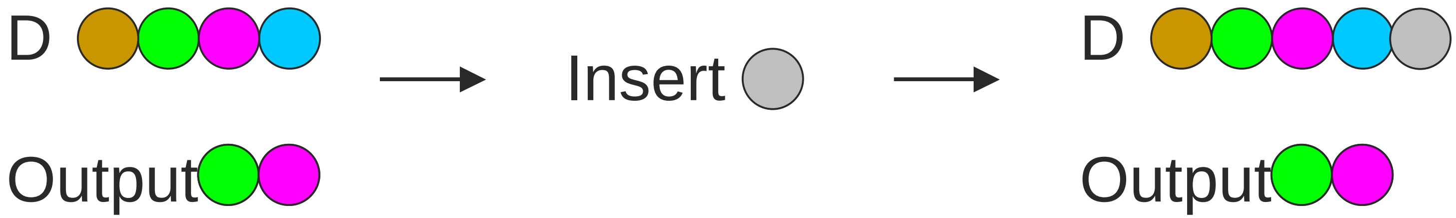
Michael's Algorithm

- Suppose we have a strange storage data structure *D* :
- When we give a point to *D*, it stores the point and outputs the closest pair of points stored in *D*



Michael's Algorithm

- Our knowledge: Insertion time depends on whether the closest pair is changed or not.
- If output is the same: 1 clock tick



Michael’s Algorithm

- If output is not the same: |*D*| clock ticks

Michael's Algorithm

- With random insertion order,

  show that the expected total number of clock ticks used by D is $O(n)$

Proof:

X_i: # clock ticks to insert i th point

X: the total clock ticks

Michael's Algorithm

Proof (cont.):

p = Pr(i th point causes answer change)

= Pr(i th point causes answer change)

= $2/i$

➔ $E[X_i] = i*p + 1*(1-p) = i* 2/i + 1 - 2/i < 3$

➔ $E[X] = O(n)$ by linearity of expectation

One of Three

One of Three

- A company is developing a prediction system by machine learning
- For a given item, the prediction has
  - Pr(success) = p_1
  - Pr(failure) = p_2
  - Pr(not sure) = p_3

One of Three

- The algorithm is run for n items
- Let

  X_1: total # with correct prediction
  X_2: total # with failure prediction

  X_3: total # with not-sure prediction

- Can we compute $E[X_1|X_3=m]$?

One of Three

Answer:

(1) $X_3 = m \Rightarrow X_1 + X_2 = n' = n - m$

(2) Let p' denote:

Pr(i th prediction correct | not not-sure)

$= p_1 / (p_1 + p_2)$

One of Three

The value of X_1 given $X_3=m$ is Bin(n',p')

$$E[X_1|X_3=m]$$
$$= n'p'$$
$$= (n-m)p_1/(p_1+p_2)$$