Today

Proof Strategies

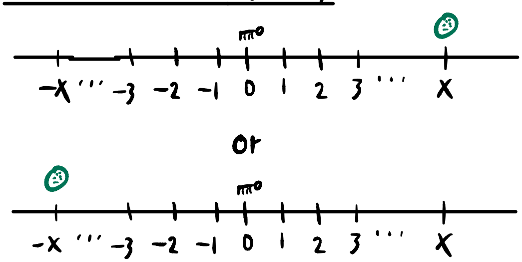
- Guessing
- Charging

- Doubling
- Halving
- Averaging

Guessing: Algorithmic problems easier when parameters of OPT are known

Often don't know parameters but can guess them

Ant on an (Infinite) Log



→ Goal: Minimize distance travelled

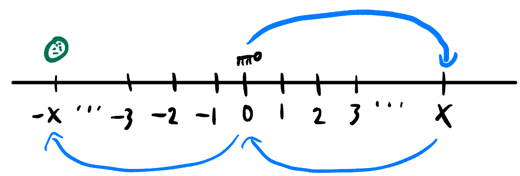
X Known

Until Cookie

Go to X

Go to -X

Analysis



Ant travels $\leq 3X$

X Not Known

Until cookie

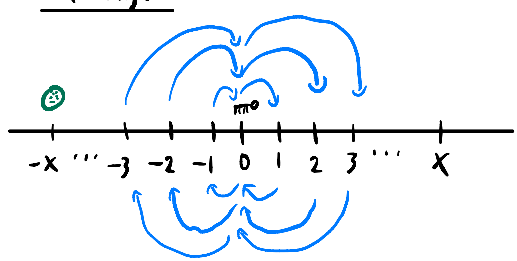
For $\hat{X} = 1, 2, 3, \ldots$ (brute force "guess" X)

Go to $\hat{X}$

Go to $-\hat{X}$

Go to 0

Analysis

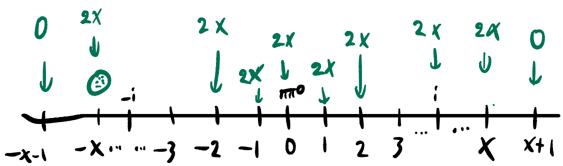


Ant travels $1+1+1+1+2+2+2+2+\ldots+X+X+X+X$

$$= 4\sum_{i=1}^{X} i = O(X^2)$$

A Changing Perspective on Analysis

Charging: to upper bound $y \le z$, create z dollars
distance travelled
Pay for each part of y w/ z dollars



So start w/ $2x+4x^2$ dollars

"charging"

Each time ant goes $i \rightsquigarrow i\pm 1$, reduce # dollars @ i by 1

$\le x$ iterations and \$ at each i reduced by ≤ 2/iteration

so never run out of \$

$\rightarrow$ Distance travelled $\le$ total # dollars $\le 2x+4x^2 = O(x^2)$

X Not Known (assume x a power of 2)

Until cookie (guess x as powers of 2)

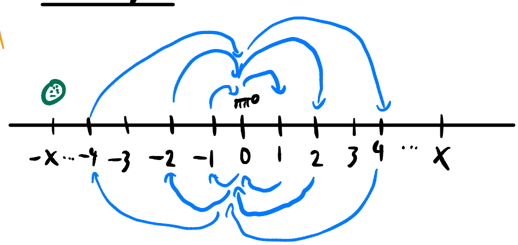
For $\hat{x} = 1, 2, 4, 8, 16, \ldots$

Go to $\hat{x}$

Go to $-\hat{x}$

Go to 0

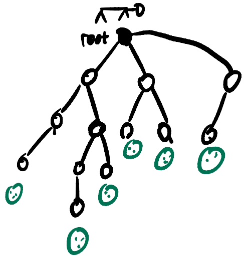
Analysis



Ant travels $\le 1+1+1+1+2+2+2+2+4+4+4+4+\ldots+x+x+x+x$
$+2x$ [illegible]

$$= 4\sum_{i=0}^{\log x} 2^i = 4\sum_{i=0}^{\log x} x/2^i = 4x\sum_{i=0}^{\log x}\frac{1}{2^i} = O(x)$$

Ant on a Tree

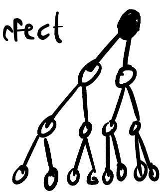


Goal: Minimize distance to a cookie @ leaf

A rooted tree is **d-ary** if all non-leaves has d children

A rooted tree is **d-perfect** if it is d-ary and all leaves at same depth

E.g. 2-perfect



level	# nodes	Subtree Size
0	1	n
1	2	$\le n/2$
2	$2 \cdot 2$	$\le n/4$
3	$2 \cdot 2^2$	$\le n/8$

Doubling: If $y \ge 1$ initially, y only increases over time and $y \le B$ always then # times $y \leftarrow y \cdot d$ is $\le \log_d B$

Halving: If $y \le B$ initially, y only decreases over time and $y \ge 1$ always then # times $y \leftarrow \frac{y}{d}$ is $\le \log_d B$

Strategy for d-Perfect Trees

Until Cookie
  Go to arbitrary child

Doubling Analysis

Let $y :=$ # nodes at ant's level

$y \geq 1$ initially (root)

$y \leq n$ always (n nodes)

After each step $y \leftarrow y \cdot d$

So # steps $\leq \log_d n$

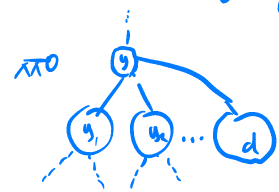
Halving Analysis

Let $y :=$ # nodes in ant's subtree

$y \leq n$ initially

$y \geq 1$ always

After each step $y \leftarrow \frac{y}{d}$ b/c



$y_i :=$ subtree size of ith child

$y = 1 + \sum_{i=1}^{d} y_i = 1 + d y_j \geq d \cdot y_j \quad \forall j$

So $y_j \leq y/d \quad \forall j$

So # steps $\leq \log_d n$

I.e. a d-Perfect tree has depth $\leq \log_d n$

Common Corollaries

d:	2	$\log n$	$2^{\sqrt{\log n}}$	$\sqrt{n}$	$n^{\varepsilon}\ \forall \varepsilon > 0$
Depth:	$\log n$	$\frac{\log n}{\log\log n}$	$\sqrt{\log n}$	2	$1/\varepsilon$

Averaging: Given $y_1, y_2, \ldots, y_d \in \mathbb{R}$, $\exists y_j$ s.t. $y_j \leq \frac{\sum_i y_i}{d}$

Strategy for d-ary Trees

Until Cookie

Go to child j minimizing y_j

Analysis

$$y_j \leq \frac{\sum_i y_i}{d} \leq \frac{y}{d}$$

So take $\leq \log_d n$ steps by a halving argument

Corollary: every d-ary tree has a root $\rightsquigarrow$ leaf path of length $\leq \log_d n$