Tutorial II

06 March 2019 10:04

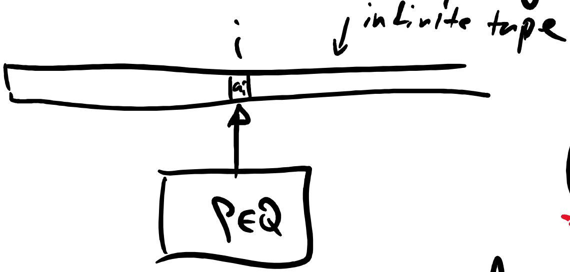
CLASSIFICATION OF RANDOMIZED ALGORITHMS

TURING MACHINE COMPLEXITY CLASSES

↳ Concerns decision problems

DTM - deterministic turing machine

input Σ



$$(Q\times\Sigma)\to(Q\times\Sigma\times\{\leftarrow,\to,\downarrow\})$$

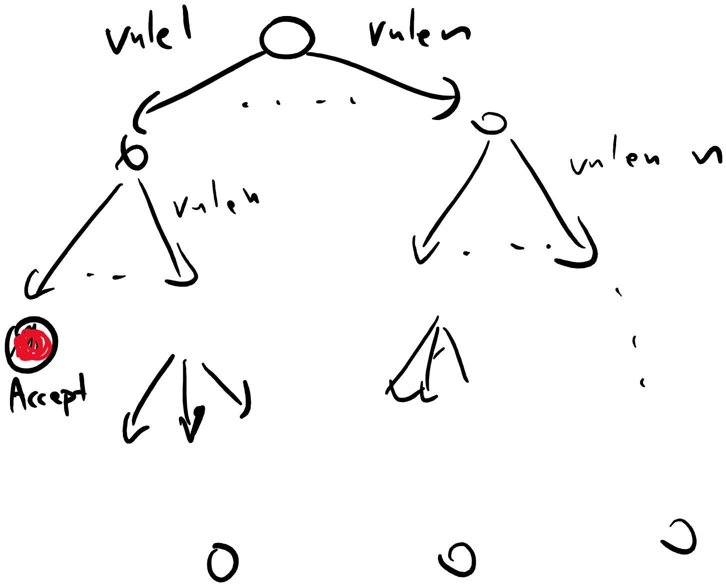
$Acc \subseteq Q$

$x\in L \to$ TM finds Accepting state in polynomial time (P)

NTM - non-deterministic TM.

$$(Q\times\Sigma)\times(Q\times\Sigma\times\{\leftarrow,\to,\downarrow\})$$

↓ informally multiple rules for the same input



$x \in L \rightarrow \exists$ accepting terminating state

PTM - probabilistic Turing Machine

it is NTM where choices of rules are assigned probabilities

random polynomial

RP: $x \in L$: $Pr[TM(x) \text{ accepts}] \geq 1/2 \leftarrow$

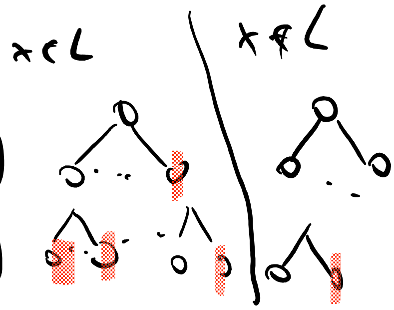
$x \notin L$: $Pr[TM(x) \text{ accepts}] = 0$

co-RP: $x \in L$: $Pr[TM(x) \text{ accepts}] = 1$

$x \notin L$: $Pr[TM(x) \text{ accepts}] \leq 1/2$

BPP: $x \in L$: $Pr[TM(x) \text{ accepts}] \geq 3/4$ $(1/2 + \varepsilon)$

$x \notin L$: $Pr[TM(x) \text{ accepts}] < 1/4$ $(1/2 - \varepsilon)$



PP: $x \in L$: $Pr[TM(x) \text{ accepts}] > 1/2$ $(1/2 + 1/2^n)$ size of the input

$x \notin L$: $Pr[TM(x) \text{ accepts}] \leq 1/2$ $(1/2 - 1/2^n)$

Run PTM on the same input x $2N+1$ times

and output the answer that appears at least $N+1$ times

(Majority voting).

$x \in L$

Yes $3/4$

v

No $1/4$

$x \in L$

$\boxed{Y|Y|Y}$
$\boxed{Y|Y|N}$
$\boxed{Y|N|Y}$
$\boxed{N|Y|Y}$ Yes

$\vee$

$\boxed{N|N|N}$
N N Y
N Y N
Y N N No

$$3 \cdot \left(\tfrac{3}{4}\right)^2 \cdot \tfrac{1}{4} + \left(\tfrac{1}{4}\right)^2 \cdot \left(\tfrac{3}{4}\right) > \tfrac{3}{4}$$

$\vee$

For chosen confidence level (probability of correct answer) how many repetitions are needed?

BPP - polynomially many in n
PP - can be exponential in n
} formally with Chernoff bounds next week

$ZPP = co\text{-}RP \cap RP$

Classification without TM → works for functions as well

Las Vegas algorithm:

Two flavours:

1.) Always polynomial time, Answer is correct or "I don't know" with bounded probability

2.) Answer is always correct and runs in <u>expected</u> polynomial time

Why are those equivalent?

$2 \Rightarrow 1$

if runs <u>too long</u> stop and say "I don't know".

"Expected polynomial" $\Rightarrow$ Vast majority of calculations are "short"

taken over random choices

$1 \Rightarrow 2$ <u>Probability amplification</u>

Let $0 < \varepsilon < \delta < 1$

$\Pr(LV(x) \text{ gives answer}) = \varepsilon$

$\Pr(LV(x) = ??) = 1 - \varepsilon$

How many repetitions are needed for

$\Pr([LV(x)]^k \text{ gives answer}) > \delta$?

$[LV(x)]^k \to$ run LV k times and give an answer if found

$[LV(x)]^k = ??$ only if all k runs didn't find an answer

$\Pr\{(LV(x))^k = ??\} = (1-\varepsilon)^k$

$\Pr\{(LV(x))^k = \text{answer}\} = 1 - (1-\varepsilon)^k$

$$1-(1-\varepsilon)^k \geq \delta$$

$$(1-\delta) \geq (1-\varepsilon)^k \qquad / \log$$

$$\log(1-\delta) \geq \log(1-\varepsilon)^k \qquad \tfrac{1}{4} \leq \tfrac{1}{2} \quad / \log_2$$

$$\log(1-\delta) \geq k \underbrace{\log(1-\varepsilon)}_{\to \text{ negative}} \qquad -2 \leq -1$$

$$\frac{\log(1-\delta)}{\log(1-\varepsilon)} \leq k$$

For $\varepsilon = f(n)$ k depends on n and amplification might not be efficient!

size of the input

if LV algorithm is calculating a decision problem then problem is in $ZPP = co\text{-}RP \cap RP$

RP - Yes answer is always correct in poly-time

co-RP - No answer is always correct in poly-time

RP implies

$$TM_1(x) \to \begin{cases} x \in L & Pr(YES) \geq 1/2 \quad Pr(NO) \leq 1/2 \\ x \notin L & Pr(NO) = 1 \quad Pr(Yes) = 0 \end{cases} \qquad 1/2^{-n}$$

co-RP implies

$$TM_2(x) \begin{cases} (x \in L) \Rightarrow Pr(Yes) = 1 & Pr(NO) = 0 \\ (x \notin L) \Rightarrow Pr(NO) \geq 1/2 & Pr(YES) \leq 1/2 \end{cases}$$

Run $TM_1(x)$ and if Yes, say Yes

if No run ($TM_2(x)$ and if NO) say NO

if No run($TM_2(x)$) and if NO) say NO

1-MC algorithms → these are defined for decision problems only

all problems in RP have 1-MC algorithm

2-MC algorithms correct calculation of function w.p. $\geq 3/4$
incorrect calculation of function w.p. $< 1/4$

if function calculates a decision problem

all problems in BPP have 2-MC algorithm

UMC algorithms → defined for functions
correct calculation of function w.p. $> 1/2$
incorrect calculation of function w.p. $< 1/2$

if function is calculating a decision problem
all problems in PP have UMC algorithms