L10 | the law of iterated logarithm

Only for coin tossing (fair) $X_1, X_2, \ldots$ independent coin tosses, $Pr(X_i = 1) = Pr(X_i = -1) = \frac{1}{2}$.

Theorem (the LIL) $S_n := X_1 + X_2 + \cdots + X_n$

a) $\forall \varepsilon > 0$: $Pr\left(\frac{S_n}{\sqrt{(n/2)\log(\log n)}} \geq 1+\varepsilon \text{ i.o.} \right) = 0$

b) $\forall \varepsilon > 0$: $Pr\left(\frac{S_n}{\sqrt{\frac{n}{2}\log(\log n)}} \geq 1-\varepsilon \text{ i.o.} \right) = 1$

where i.o. = for infinitely many n = infinitely often

Aleksandr Ya. Khinchin (Khinchin) (1894–1959) 1924

Andrej N. Kolmogorov (1903–1987) – 1929

– But this takes place in an uncountable probab. space.

Randomized probability testing

Theorem (Probable primes) $\exists$ an algorithm $\mathcal{A} = \mathcal{A}(n,x) : \{(n,x) \in \mathbb{N}^2 \mid n > 1 \text{ & odd}, 1 < x < n\} \to \{0,1\}$ with the variables $q, n, q, y \in \mathbb{N}$ and $j \in \mathbb{N}_0$, with the failure parameter $x \in \mathbb{N}$.

output 0 ... "n is definitely not prime"
1 ... "n is probably prime". The algorithm:

(1) Main input: an odd number $n = 1 + 2^q q > 1$ where q is odd, to be tested for primality.

(2) Input the failure parameter: $x \in \mathbb{N}$, $1 < x < n$.

(3) Exponentiation: $j := 0$ and take the $y \in [n]$ ("$\{1, \dots, n\}$") such that $y \equiv x^q \pmod n$.

(4) Done?. [now $y \equiv x^{2^j q} \pmod n$.] If $j = 0$ and $y = 1$, or if $y = n-1 \Rightarrow$ output "n is probably prime" and terminate.

(5) Increase j: $j := j+1$. If $j < q$, set $y :=$

$:= q^2 \pmod n$, $q \in [n]$, and go to step 4.

(6) Not prime: output "n is definitely not prime" and terminate.

This algorithm terminates on any input (n, x) in time polynomial in $\log n$.

God's view: If n is prime, $\forall x$ the algor. correctly outputs "n is probably prime". If n is composite, for some less than $\frac{n-2}{4}$ numbers x with $1 < x < n$, known to me, the algor. fails and and outputs "n is probably prime" but for other values of x it correctly outputs "n is definitely not prime".

User's view: the answer of the algor. "n is definitely not prime" is always correct, for any x. The other answer "n is probably prime" may be incorrect for some less than $\frac{n-2}{4}$ numbers x with $1 < x < n$ but of course one does not know which are these x.

Proof. ∎

$r \in \mathbb{N}$, the modified algorithm A_r has inputs $(r, n, x_1, \dots, x_r)$ where the (n, x_i) are r inputs of A, the output is

$A_r(r, n, x_1, \dots, x_r) := \min(A(n, x_1), \dots, A(n, x_r)) \in \{0, 1\}$ (0 def. output, 1 prob. prime)

(r runs of A) $I_n := \{2, 3, \dots, n-1\}$

$$\frac{\#\{(x_1, \dots, x_r) \in I_n^r \mid A_r(r, n, x_1, \dots, x_r) \text{ fails}\}}{\#I_n^r \ (= (n-2)^r)} < \left(\frac{1}{4}\right)^r.$$

n is composite but output is 1.

- Michael O. Rabin (1931) $\doteq$ 1980
- Donald E. Knuth (1938) — Algorithm P in TAOCP, Vol. 2, p. 395.

MARKOV CHAINS

- Andrej A. Markov (1856–1922)

$X_0, X_1, X_2, \ldots$ random variables, $X_t \in \mathbb{Z} =$

$= \{\ldots, -1, 0, 1, \ldots\}$

Def. is a Markov chain if $\forall t \in \mathbb{N}$:

$$Pr(X_t = a_t \mid X_{t-1} = a_{t-1}, X_{t-2} = a_{t-2}, \ldots, X_0 = a_0) =$$

$$= Pr(X_t = a_t \mid X_{t-1} = a_{t-1}) =: p_{a_{t-1}, a_t}$$

Markov property = memoryless property = any dependence on the past is captured in the value of X_t

X_{t-1}.

time step

$X_t \in \mathbb{N}_0 = \{0, 1, \ldots\} \ni i, j$ or $\in \{0, 1, \ldots, n\} =: [n]_0$

the states

Transition probabilities:

$$p_{i,j} := Pr(X_t = j \mid X_{t-1} = i),$$

one-step transition matrix

infinite matrix:

$$P = \begin{pmatrix} p_{0,0} & p_{0,1} & p_{0,2} & \cdots & p_{0,j} & \cdots \\ p_{1,0} & p_{1,1} & p_{1,2} & \cdots & p_{1,j} & \cdots \\ \vdots & \vdots & \vdots & & \vdots & \\ p_{i,0} & p_{i,1} & p_{i,2} & \cdots & p_{i,j} & \cdots \\ \vdots & \vdots & \vdots & & \vdots & \end{pmatrix} \in [0,1]^{\mathbb{N}_0 \times \mathbb{N}_0}$$

or

finite matrix

$$\in [0,1]^{[n]_0 \times [n]_0}$$

$\rightsquigarrow \forall i: \sum_{j \geq 0} P_{i,j} = 1$ (the row sum) (6)

For $t \in \mathbb{N}_0$, $\vec{p}(t) := (p_0(t), p_1(t), p_2(t), \dots) \in [0,1]^{\mathbb{N}_0, [n]}$ where $p_i(t)$ = the prob. that the process is in state i at time t. $\forall t \in \mathbb{N}$:

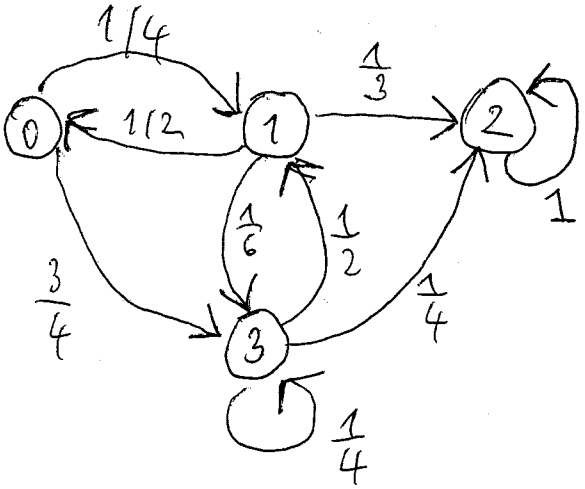
$\rightsquigarrow \boxed{\vec{p}(t) = \vec{p}(t-1) \cdot P.}$ For i,j $m \in \mathbb{N}_0$ we de-

fine $P^m_{i,j} = Pr(X_{t+m} = j \mid X_t = i)$ = the probab. that the chain moves from state i to state j in exactly m steps.

$\rightsquigarrow$ If $P^{(m)} = (P^m_{i,j})_{i,j \in \mathbb{N}_0}$ or $\in [n]_0$ then $\boxed{P^{(m)} = P^m}$ (where $P^0 = I = \begin{pmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{pmatrix}$).

The digraph

new: paths in a weighted digraph



$$P = \begin{pmatrix} 0 & \frac{1}{4} & 0 & \frac{3}{4} \\ \frac{1}{2} & 0 & \frac{1}{3} & \frac{1}{6} \\ 0 & 0 & 1 & 0 \\ 0 & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \end{pmatrix}$$

$$P^{(3)} = \begin{pmatrix} 3/16 & 7/48 & 29/64 & 41/192 \\ 5/48 & 5/24 & 79/144 & 5/36 \\ 0 & 0 & 1 & 0 \\ 1/16 & 13/96 & 107/192 & 47/192 \end{pmatrix} = P^3$$

, so for example $p^3_{0,3} = \frac{41}{192}$

(paths 0103, 0133, 0313 and 0333).

Randomized algorithm for 2-satisfiability

variables: x_1, x_2, x_3, x_4 (in general $x_1, x_2, \dots, x_n$)

(Boolean) formula like — clause — literals

$$F = (x_4 \vee \bar{x}_1) \wedge (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_3) \wedge (x_1 \vee x_2) \wedge (x_4 \vee \bar{x}_3)$$

? $\exists\, x_1, \dots, x_4 \in \{0,1\}$ s.t. $F = 1$ $(= 1 \wedge 1 \wedge 1 \wedge 1 \wedge 1)$

2-SAT Algorithm

① Start with arbitr. truth assignment $S_0: \{x_1, \dots, x_n\} \to \{0,1\}$.

② Repeat up to $2mn^2$ times ($m \in \mathbb{N}$), terminating if all clauses are satisfied:

(a) Choose an arbitr. clause that is not satisfied

(b) Choose uniformly at random one of the literals in the clause and switch the value of its variable.

③ If a valid truth assign. has been found, return it. the formula

④ Else, return that F is unsatisfiable.

5-satisfying assign. for $x_1, \dots, x_n$

x_i = the # of var-s (in the current assign.) that are the same as $s_{(n)}$ after the step i.

$\Pr(x_{i+1} = 1 \mid x_i = 0) = 1$.

$1 \le j \le n-1$:

$\Pr(x_{i+1} = j+1 \mid x_i = j) \ge \frac{1}{2}$,

$\Pr(x_{i+1} = j-1 \mid x_i = j) \le \frac{1}{2}$.

$x_0, x_1, x_2, \dots$ not necess. a M. chain. M. chain $Y_0, Y_1, \dots$: $Y_0 = x_0$,

$\Pr(Y_{i+1} = 1 \mid Y_i = 0) = 1$; $\Pr(Y_{i+1} = j+1 \mid Y_i = j) = \frac{1}{2}$;

$\Pr(Y_{i+1} = j-1 \mid Y_i = j) = \frac{1}{2}$.

$\mathbb{E}$ time to reach n in the process $X \le \mathbb{E}$ time to reach n in the process Y (from any starting point).

Y: r. walk on (undir.) graph $G = (\{0, 1, \dots, n\}, E)$

E: 

Z_j := the # steps to reach n from j, $h_j := \mathbb{E} Z_j$, for $1 \le j \le n-1$:

$\mathbb{E} Z_j = \mathbb{E}\left(\frac{1}{2}(1 + Z_{j-1}) + \frac{1}{2}(1 + Z_{j+1})\right)$, by (in.) of expectation

$\rightsquigarrow h_j = \frac{1}{2}(h_{j-1} + h_{j+1}) + 1$.

$h_n = 0$, $1 \le j \le n-1$

$h_j = \frac{1}{2}(h_{j-1} + h_{j+1}) + 1$,

$h_0 = h_1 + 1$.

Thus we have the system

Lemma For $0 \le j \le n-1$, $h_j = h_{j+1} + 2j+1$. Hence $h_0 = h_1 + 1 = h_2 + 1 + 3 = \dots = 0 + \sum_{i=0}^{n} (2i+1) = n^2$.

Proof. By induction. ☒

Corollary If a 2-SAT formula with n vars has a satisfying assign. and if the 2-SAT algorithm runs until it finds a satisf. assign. $\Rightarrow$ # steps until the algor. finds $\le n^2$.

Theorem the 2-SAT algor. always returns a correct answer if F is unsatisfiable. If F is satisfiable $\Rightarrow$ with $\Pr \ge 1 - 2^{-n}$ the alg. returns a satisf. assign. Else it incorrectly returns that F is unsatisfiable.

Proof. No satisf. assign. – clear. Let F be satisf. Executions of the alg.: n segments

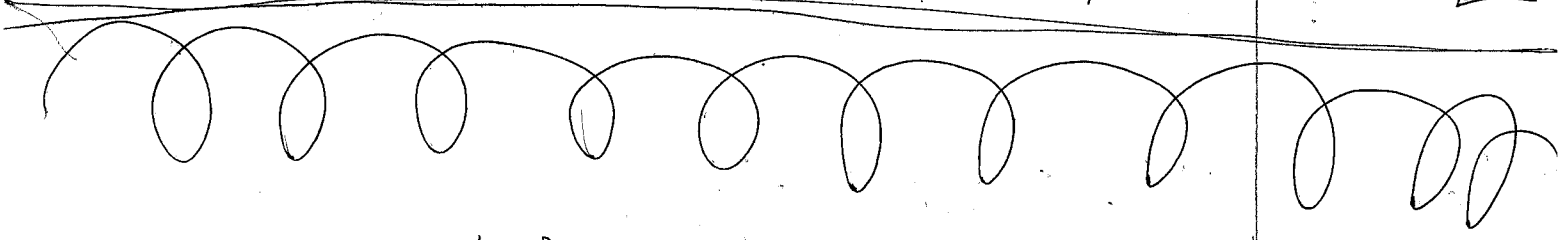
By the Corollary:

$\Pr($ no satisf. assign. found in the i-th segment | no satisf. assign. found in the segments $1 \dots i-1$

$< \frac{1}{2}$, because if $Z :=$ # steps from the start of the i-th segment until the alg. finds a satisf. assign., then $\mathbb{E}Z \le n^2$ by the Corollary, and so by Markov's ineq.

$$\Pr(Z > 2n^2) \le \frac{\mathbb{E}Z}{2n^2} \le \frac{n^2}{2n^2} = \frac{1}{2}.$$

$\rightsquigarrow \Pr(\text{the alg. fails to find a satisf. assign. after } n \text{ segments}) \le \left(\frac{1}{2}\right)^n$.



Thank you!