Sept. 24, 2015@ELC秋学校(岐阜)

Exploring the Limits of Computation

# 乱択の技法

3. 今後の課題: BPP=P?

来嶋秀治

九州大学 大学院システム情報科学研究院

# BPP (Bounded-Error Probabilistic Polynomial-Time)

言語$L$がBPP ⇔
多項式時間で停止する
確率的 Turing Machineが存在し，以下の条件を満たす．
$x \in L$なら確率2/3以上で受理．
$x \notin L$なら確率1/3以下で受理．

| | ACCEPT | REJECT |
|---|---|---|
| 真の解: YES | $\geq 2/3$ | $\leq 1/3$ |
| 真の回: NO | $\leq 1/3$ | $\geq 2/3$ |

# BPL (Bounded-Error Probabilistic L)

言語$L$がBPL ⇔

入力サイズ$n$の対数領域を使用する

確率的 Turing Machineが存在し，以下の条件を満たす．

$x \in L$なら確率2/3以上で受理．

$x \notin L$なら確率1/3以下で受理．

| | ACCEPT | REJECT |
|---|---|---|
| 真の解: YES | $\geq 2/3$ | $\leq 1/3$ |
| 真の回: NO | $\leq 1/3$ | $\geq 2/3$ |


# 3.1. ストリーム中の頻出アイテム検知

緒方 正虎，山内 由紀子，来嶋 秀治，山下 雅史

九州大学

θ: “頻出度”パラメータ

Σ: アイテム集合(有限)

# ストリームデータ中の頻出アイテム検知

## 問題: 頻出アイテム検知

Input: $\theta \in (0,1)$ $\mathbf{x} = (x_1,\ldots,x_N) \in \Sigma^N$ (順々に)

Find: all $s \in \Sigma$ s.t. $f(s) \geq \theta \cdot N$

但し f(s) は s が **x** 中で出現した回数

事前には、
N (or log Nの近似値)も
わからない。

例1. 1日のPOSデータ(@果物屋)

$\Sigma$ ={  }

**x** = 

例. 1日のアクセスIPアドレス

$\Sigma \subseteq$ {0.0.0.0,…,255.255.255.255}

**x** = 123.45.67.89, 111.11.1.1., 123.45,67,89, 122.122.12.12…

would like to find items appearing w/ frequency more than θ=1% of N.

# 頻出アイテム検知の領域複雑度

定理 [Karp, Shenker, Papadimitriou '03]

頻出アイテム検知を厳密に行うには,

$\Omega(|\Sigma| \log (N/|\Sigma|))$ bits が必要.

(N >> $|\Sigma|$ >> $1/\theta$ とする)

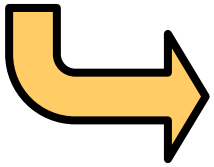
定理 [Karp, Shenker, Papadimitriou '03]

頻出アイテム検知に対する

$O((1/\theta) \log N)$ bits の偽陽性(近似)アルゴリズムが存在.

(N >> $|\Sigma|$ >> $1/\theta$ とする)

決定的



- $o(\log N)$ bitsアルゴリズム？
  - e.g. $O(\log \log N)$ bits?

# 単純化: o(log N) bitsで要素数を数えられるか?

**問題: 要素数え上げ**

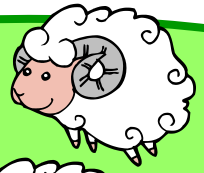
Input: $x = (a,\ldots,a) \in \Sigma^N$ (順々に)

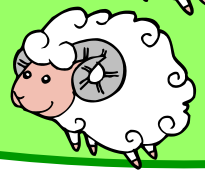
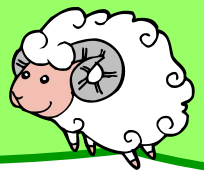
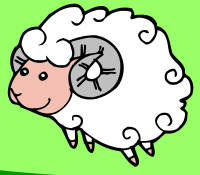
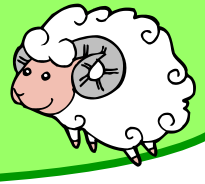
Find: N

事前には、
N (or log Nの近似値)も
わからない。

**アルゴリズム: 数え上げ**

0. **Set** n:= 0.
1. **Read** an input. **If** no more input, **goto** 3.
2. n++, **Goto** 1.
3. **Output** n (as N = n).


$\Sigma$ = {  }

X =  ,  ,  ,  , … …

# 単純化: o(log N) bitsで要素数を数えられるか？

問題: 要素数え上げ

Input: $x = (a,\ldots,a) \in \Sigma^N$ (順々に)

Find: N

事前には、
N (or log Nの近似値)も
わからない。

アルゴリズム: 数え上げ

0. **Set** n:= 0.
1. **Read** an input. **If** no more input, **goto** 3.
2. n++, **Goto** 1.
3. **Output** n (as N = n).

O(log N) bits

o(log N) bits近似アルゴリズム？

- e.g. O(log log N) bits?

# 単純化: o(log N) bitsで要素数を数えられるか？

問題: 要素数え上げ

Input: $x = (a,\ldots,a) \in \Sigma^N$ (順々に)

Find: N

事前には、
N (or log Nの近似値)も
わからない。

Remark

- NはO(log N) bitsで表現可能.
  - ✓ N = 1,351,127,649,213
- Nの近似はO(log log N) bitsで表現可能
  - ✓ $N \approx 1.351 \times 10^{12}$

つまり、

指数部(=log N)がo(log N) bits領域で近似計算できるか？ということ。

表現はO(log log N) bitsで可能

# 単純化: o(log N) bitsで要素数を数えられるか？

問題: 要素数え上げ

Input: $\mathbf{x} = (a,\ldots,a) \in \Sigma^N$ (順々に)

Find: N

事前には、
N (or log Nの近似値)も
わからない。

アルゴリズム: 数え上げ

0. **Set** n:=0.
1. **Read** an input. **If** no more input, **goto** 3.
2. n++, **Goto** 1.
3. **Output** n (as N = n).

$\Theta(\log N)$ bits

定理. [Flajolet '85, Ogata et al. '11]
決定的アルゴリズムでは$\Omega(\log N)$ bit 必要！

# 確率的数え上げ

**問題: 要素数え上げ**

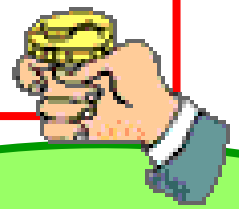
Input: $x = (a,\ldots,a) \in \Sigma^N$ (順々に)

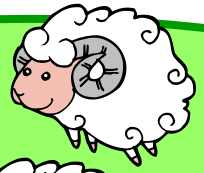
Find: N

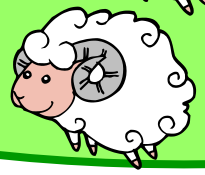
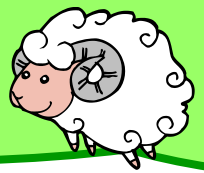
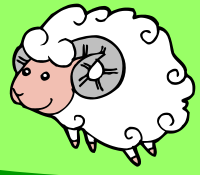
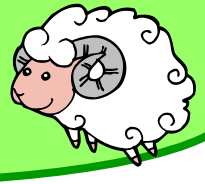
事前には、
N (or log Nの近似値)も
わからない。

**アルゴリズム: 確率的数え上げ**

0. **Set** k:=0.
1. **Read** an input. **If** no more input, **goto** 3.
2. k++, w.p. $1/2^k$. **Goto** 1.
3. **Output** k (as $N \approx 2^k$).



$\Sigma$ = {  }

X =  ,  ,  ,  , … …

# 確率的数え上げ

**問題: 要素数え上げ**

Input: $x = (a,\ldots,a) \in \Sigma^N$ (順々に)

Find: N

事前には、
N (or log Nの近似値)も
わからない

⇒ key point
"w.p. $1/2^k$ " using
O(log K) bits on PTM.

**アルゴリズム: 確率的数え上げ**

0. **Set** k:=0.
1. **Read** an input. **If** no more input, **goto** 3.
2. k++, w.p. $1/2^k$. **Goto** 1.
3. **Output** k (as $N \approx 2^k$).

O(log log N) bits

**Thm.** [Morris '78, Flajolet '85]
$E[2^k] \approx N+1$

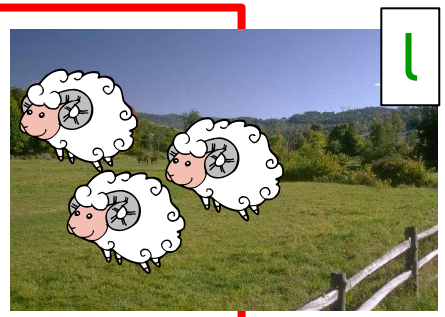
because
$N \approx 1+2+4+8+16+\ldots+2^k$

# 確率的数え上げ(改良型)

問題: 要素

(直感的に) “l” (< $2^b$) は仮数部分を表す.

i.e., 各ひつじを牧場“l”に(確率$1/2^k$で)捕まえておく.




アルゴリズム: 確率的数え上げ(改良型)

0. **Set** k:=0, l:=0.
1. **Read** an input. **If** no more input, **goto** 4.
2. l++, w.p. $1/2^k$.
3. If l==$2^b$, k++, and **set** l := l’ w.p. $\binom{2^b}{l'} \cdot 2^{-2^b}$. **Goto** 1.
4. Output

各ひつじは, k反復目の時, 確率$1/2^k$で(牧場“l”に)残っている.
なぜなら“exponent”=jの時捕まった確率$1/2^j$で捕まったひつじは“exponent”=k (k > j)の時, 確率$1/2^{k-j}$で牧場に残っている.

⇒ Pr[ひつじ in l] = $1/2^j * 1/2^{k-j} = 1/2^k$.

# 確率的数え上げ(改良型)

問題: 要素数え上げ

Input: $\mathbf{x} = (a,\ldots,a) \in \Sigma^N$ (順々に)

Find: N

事前には、
N (or log Nの近似値)も
わからない。

アルゴリズム: 確率的数え上げ(改良型)

0. **Set** k:=0, l:=0.
1. **Read** an input. **If** no more input, **goto** 4.
2. l++, w.p. $1/2^k$.
3. If l==$2^b$, k++, and **set** l := l' w.p. $\binom{2^b}{l'} \cdot 2^{-2^b}$. **Goto** 1.
4. **Output** l and k (as $N \approx l * 2^k$ ).

O(log log N) bits

Thm. [Ogata, Yamauchi, K., Yamashita '11]

$E[l*2^k] \approx N$.

"改良型" を使うと、頻出アイテム検知も可能. (詳細略)

# ストリームデータ中の頻出アイテム検知

問題: 頻出アイテム検知

Input: $\theta \in (0,1)$ $\mathbf{x} = (x_1, \ldots, x_N) \in \Sigma^N$ (順々に)

Find: all $s \in \Sigma$ s.t. $f(s) \geq \theta \cdot N$

但し $f(s)$ は $s$ が $\mathbf{x}$ 中で出現した回数

事前には、
N (or log Nの近似値)も
わからない。

o(log N) bitsアルゴリズム？

- e.g. O(log log N) bits?

# ストリームデータ中の頻出アイテム検知

問題: 頻出アイテム検知

Input: $\theta \in (0,1)$ $\mathbf{x} = (x_1, \ldots, x_N) \in \Sigma^N$ (順々に)

Find: all $s \in \Sigma$ s.t. $f(s) \geq \theta \cdot N$

但し $f(s)$ は $s$ が $\mathbf{x}$ 中で出現した回数

事前には、
N (or log Nの近似値)も
わからない。

Thm. [Ogata, Yamauchi, K., Yamashita '11]

O(log log N) bits 領域の乱択近似アルゴリズムが存在.

詳細略

o(log N) bitsアルゴリズム？

- e.g. O(log log N) bits?

# 3.2. Probabilistic Counting

# 確率的数え上げ

**問題: 要素数え上げ**

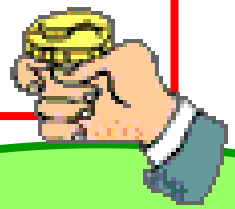
Input: $x = (a,\ldots,a) \in \Sigma^N$ (順々に)

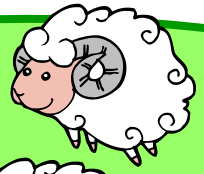
Find: N

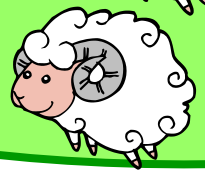
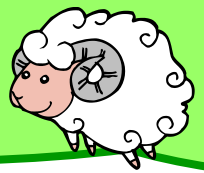
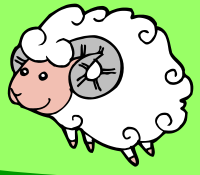
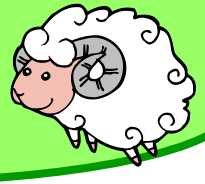
事前には、
N (or log Nの近似値)も
わからない。

**アルゴリズム: 確率的数え上げ**

0. **Set** k:=0.
1. **Read** an input. **If** no more input, **goto** 3.
2. k++, w.p. $1/2^k$. **Goto** 1.
3. **Output** k (as $N \approx 2^k$).



$\Sigma$ = {  }

X =  ,  ,  ,  , … …

# Probabilistic Counting by Morris

## Prob. Count. by Morris.

0. Set "exponent" $h := 0$.
1. Read an input if exists, otherwise goto 3.
2. Increment $h$ by one w/ porbability $\frac{1}{2^h}$. Goto 1.
3. Output $h$ (as $\log n$).

R. Morris, Counting large numbers of events in small registers, Communications of the ACM, **21**(1978), 840–842.

# Probabilistic Counting by Karpinski & Verbeek

## Prob. Count. by Karpinski & Verbeek

0. Set "exponent" $h \coloneqq 0$. Set $\mathrm{TMP} \coloneqq 0$.
1. Read an input if exists, otherwise goto 6.
2. Generate a random bit $r \in \{0,1\}$.
3. If $r = 1$ then $\mathrm{TMP}++$,
4. else $\mathrm{TMP} \coloneqq 0$.
5. If $\mathrm{TMP} > h$, then $h \coloneqq \mathrm{TMP}$ (infact, just $h++$).
6. Return $h$ (as $\log n$).

M. Karpinski and R. Verbeek. On the Monte Carlo Space Constructible Functions and Separation Results for Probabilistic Complexity Classes. Information and Computation, 75:178-189, 1987.

# Probabilistic Counting by Uehara

## Prob. Count. by Uehara

0. Set "exponent" $h \coloneqq 0$. Set $\mathrm{TMP} \coloneqq 0$.
1. Read an input if exists, otherwise goto 8.
2. Set $\mathrm{TMP} \coloneqq 0$. Set $r \coloneqq 1$.
3. While$(r = 1)\{$
4. Generate a random bit $r \in \{0,1\}$
5. $\mathrm{TMP}++$ unless $r = 0$.
6. $\}$
7. If $\mathrm{TMP} > h$, then $h \coloneqq \mathrm{TMP}$.
8. Return $h$ (as $\log n$)

上原隆平. 確率Turing Machineにおける低いレベルの領域強構成可能性について. 情報基礎理論ワークショップ, 1993.

# $\log\log n$を$O(\log\log\log n)$bit計算領域で近似できるか？

(たぶん)未解決問題

$\log\log n \in \mathrm{BPSPACE}(\log\log\log n)$?

期待値の意味では可能.

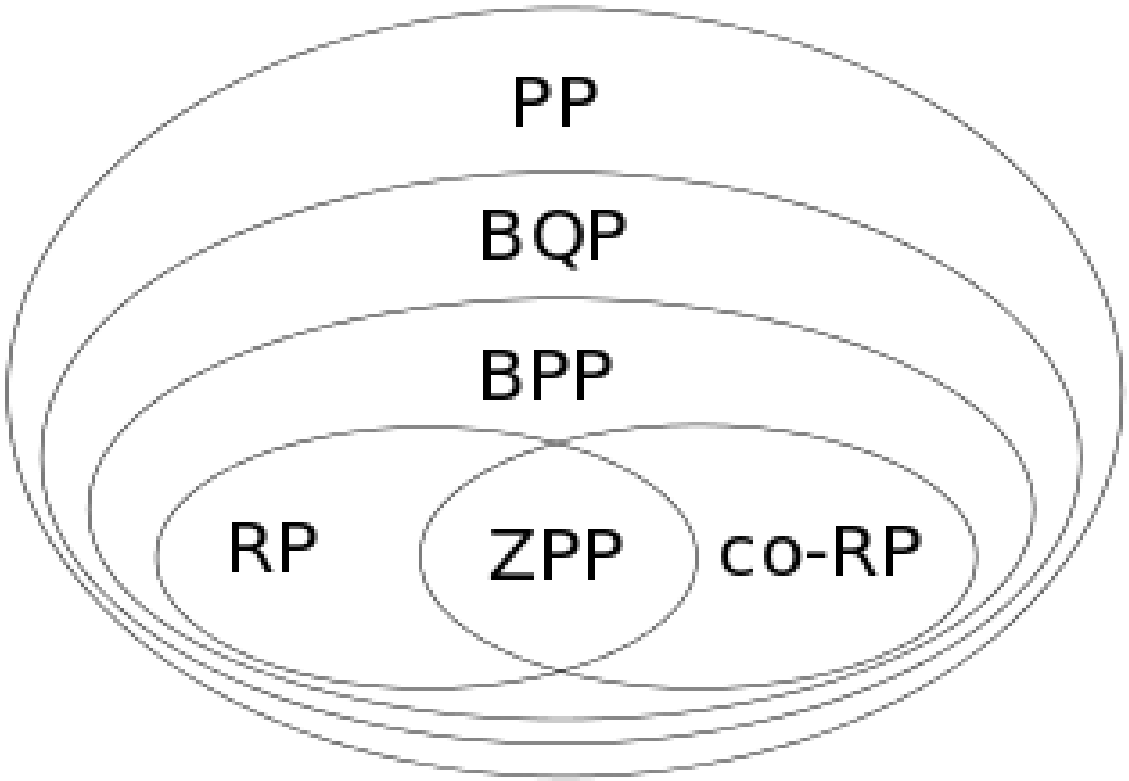
マニアックな話 = 細かい話

# 3.3. 確率的Turing Machine

決定性TM + {0,1}乱数

# BPP vs ZPP

多項式時間停止 vs 期待多項式時間




https://en.wikipedia.org/wiki/BPP_%28complexity%29

# RP (Randomized Polynomial-Time)

言語$L$がRP ⇔
多項式時間で停止する
確率的 Turing Machineが存在し，以下の条件を満たす．
$x \in L$なら確率1/2以上で受理．
$x \notin L$なら非受理．

| | ACCEPT | REJECT |
|---|---|---|
| 真の解: YES | $\geq 1/2$ | $\leq 1/2$ |
| 真の回: NO | 0 | 1 |

# ZPP (Zero-Error Probabilistic Polynomial-Time)

言語$L$がZPP ⇔

期待値として多項式時間で停止する

確率的 Turing Machineが存在し，以下の条件を満たす．

$x \in L$なら確率1で受理．

$x \notin L$なら確率1で非受理．

| | ACCEPT | REJECT |
|---|---|---|
| 真の解: YES | 1 | 0 |
| 真の回: NO | 0 | 1 |

命題

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

# 確率的TMは有限時間停止性に弱い

細かい話

命題

確率的TMでは有限時間停止の最適じゃんけんは実装できない.

最適じゃんけん

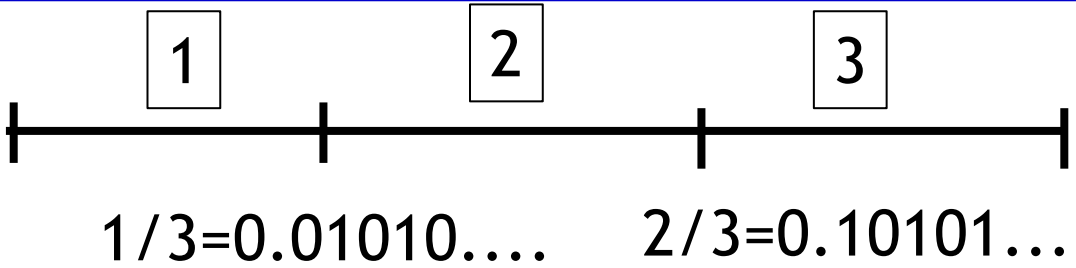
- {グー，チョキ，パー}の一様分布.
- つまり，確率1/3は有限時間停止では実装できない.

# {1,2,3}上の一様分布の実装1

generate random real $\lambda \in [0,1)$ を丁寧に書き下すと.

## 素直な実装

0. Set $i = 1$.
1. Generate $r_i \in \{0,1\}$.
2. If $0.r_1 r_2 \cdots r_i 11 \cdots < \frac{1}{3}$ then output 1.
3. If $0.r_1 r_2 \cdots r_i 00 \cdots > \frac{1}{3}$ and $0.r_1 r_2 \cdots r_i 11 \cdots < \frac{2}{3}$ then output 2.
4. If $0.r_1 r_2 \cdots r_i 11 \cdots < \frac{2}{3}$ then output 3.
5. Else goto 1.




たとえば乱数列が

0.010101

と続くと1か2か定まらない. ⇒ 有限時間停止しない. . .

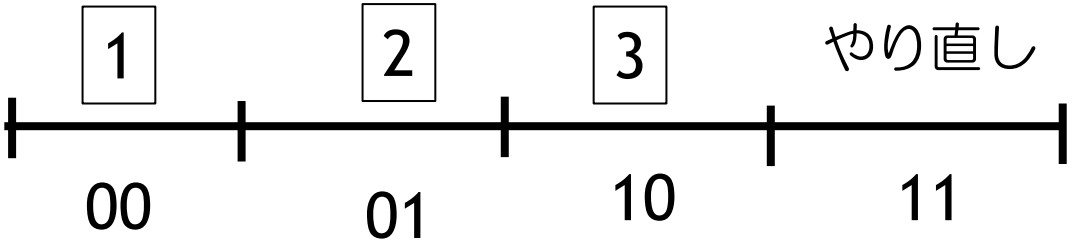
# 有限時間停止しないことの証明

- ✓ アルゴリズムAは有限時間停止するとして，
  $\{0,1\}$乱数を高々$n$個生成するものとする．
- ✓ この「高々」は「丁度」に置き換えられる．
  - ➢ 例えば，Aで $k < n$ 個の生成で出力される値は，
    そのあとも(無駄に)$n$個になるまで乱数を生成して，
    最初の $k$ 個だけ使う．
- ✓ この時，実現される確率はすべて $\frac{1}{2^n}$ の整数倍
- ✓ しかし，$\frac{1}{3} = \frac{a}{2^n}$となる$a$は存在しない．矛盾．

# {1,2,3}上の一様分布の実装2

**実装 2**

1. Generate $r_1 \in \{0,1\}$, $r_2 \in \{0,1\}$.
2. If $r_1 r_2 = 00$ then output 1.
3. If $r_1 r_2 = 01$ then output 2.
4. If $r_1 r_2 = 10$ then output 3.
5. Else goto 1.




生成する乱数の期待個数.

$$\sum_{k=1}^{\infty} 2k \left(\frac{3}{4}\right)^k = \frac{8}{3}$$

(幾何分布の期待値より)

実装1と実装2は実は同じ.

# 有理数確率の離散確率変数の生成

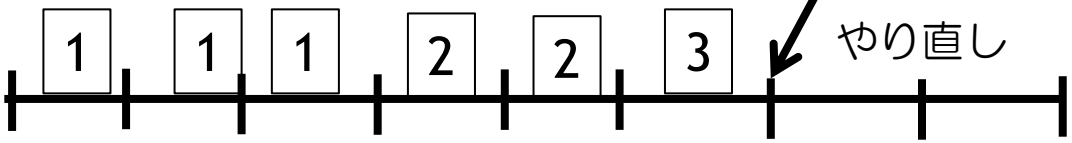
命題

確率変数$X \in \{1,2,\dots,n\}$は

$$\Pr[X = k] = \frac{b_k}{a_k}$$

とする．$A = \gcd(a_1,\dots,a_n)$とすると，
期待値として$2\lg A$個のランダムビットを生成して，
期待時間 $O(\lg A)$で所望の$X$が得られる．

細かい点として，
ここの値とまず比較する
のが$O(\lg A)$時間のポイント．




- ✓ $[0,1)$区間を$2^{\lceil \lg A \rceil}$分割して，$k$を適切に割り当てる．
  - ➢ $\frac{b_k}{a_k} = \frac{\frac{A}{a_k} b_k}{A}$ に注意．
- ✓ $A \le 2^{\lceil \lg A \rceil} \le 2A$ より，確率1/2以上で割り当てられる．
- ✓ あとは幾何分布の期待値と二分探索．

# 無理数の確率でもうまくいく

命題

確率変数$X \in \{1, \ldots, n\}$の累積分布確率を

$$F(k) = \sum_{i=1}^{k} \Pr[X = k]$$

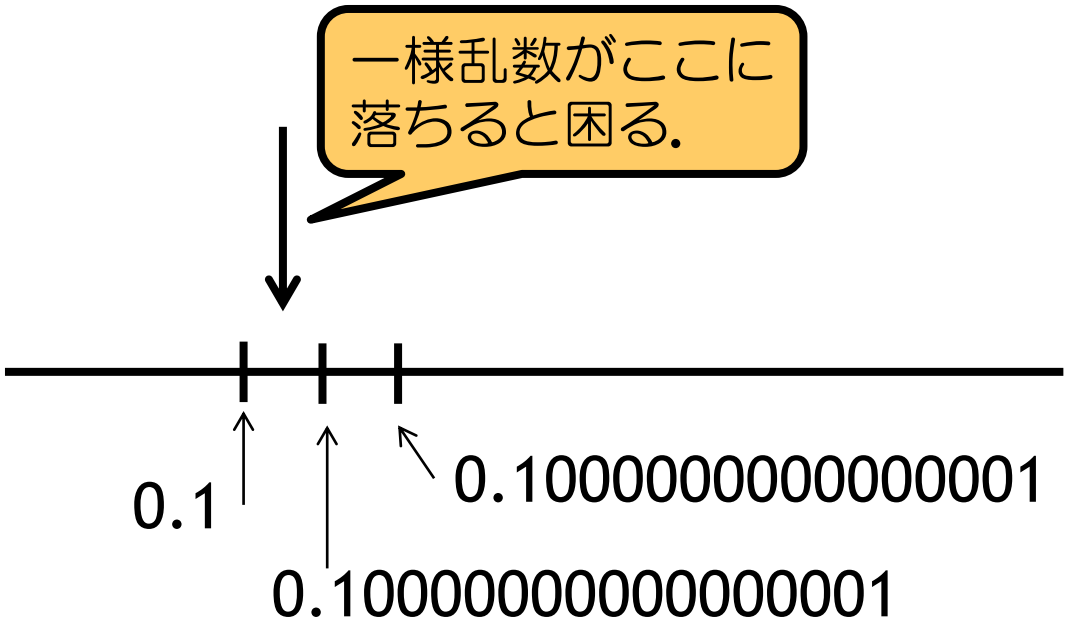
とする．任意の$k$に対して，関数$F(k)$の2進$j$桁目の値が$g(j) = O\left((2-\varepsilon)^j\right)$で計算できるとき，

期待値として高々 $2\lg n$ 個のランダムビットを生成して，

期待時間 $O\left([\log n] + \frac{2}{2-\varepsilon}\right)$で所望の$X$が得られる．

# 何が問題なの？

generate random real $\lambda \in [0,1)$ をして,
$\{1, \ldots, n\}$に関する単純な二分探索だと,
区別をつけるのに時間がかかる場合がある.



# 証明の概略

$(\lceil \log n \rceil + 1)$-bitの乱数を生成すると確率$\frac{1}{2}$以上で確定する.

$2n$個(以上)に分類されるのに，仕切りは$n$個.

- 二分探索$O(\lg n)$
- このとき，$(\lceil \log n \rceil + 1)$-bitまで計算すればよい.

$(\lceil \log n \rceil + k)$-bitの乱数を生成すると確率$1 - \left(\frac{1}{2}\right)^k$で確定する.

すなわち期待計算時間は

$$O\left(\lceil \log n \rceil + \sum_{j=1}^{\infty} \frac{1}{2^j} g(j)\right) = O\left(\lceil \log n \rceil + \sum_{j=1}^{\infty} \left(\frac{2-\varepsilon}{2}\right)^j\right)$$

$$= O\left(\lceil \log n \rceil + \frac{2}{2-\varepsilon}\right)$$

# 3.4. 対角線論法と測度

$[0,1]_{\mathbb{Q}}$: 区間[0,1]の有理数集合

$[0,1]_{\mathbb{R}}$: 区間[0,1]の実数集合

**命題**

$$\Pr\left[X \in [0,1]_{\mathbb{Q}}\right] = \frac{\left|[0,1]_{\mathbb{Q}}\right|}{\left|[0,1]_{\mathbb{R}}\right|} = 0$$

証明（アイデア）

任意の$x \in [0,1]_{\mathbb{Q}}$に対して

$$\Pr[X = x] = 0$$

有理数は可算無限なので，完全加法性より

$$\Pr\left[X \in [0,1]_{\mathbb{Q}}\right] = \sum_{x \in [0,1]_{\mathbb{Q}}} \Pr[X = x] = \sum_{x \in [0,1]_{\mathbb{Q}}} 0 = 0$$

対角線論法$\left|[0,1]_{\mathbb{Q}}\right| \neq \left|[0,1]_{\mathbb{R}}\right|$から

$$\frac{\left|[0,1]_{\mathbb{Q}}\right|}{\left|[0,1]_{\mathbb{R}}\right|} = 0$$

は導けるか？

注意: 濃度と測度はちがう。

濃度: $\left|[0,1]_{\mathbb{R}}\right| = \left|[0,1]_{\mathbb{R}} \times [0,1]_{\mathbb{R}}\right|$

測度: $[0,1]_{\mathbb{R}} \times [0,1]_{\mathbb{R}}$上の直線の面積(確率，測度）は0

=>測度の方が概念として “弱い”

計算量限界に使いたい！

# *The end*

*Thank you for the attention.*