CS687

Evaluation

TAKE HOME	Homework	15 + 15
	MidSem	30
	EndSem	40

P = BPP?

- BPP: Probabilistic Polynomial algorithms with Bounded error
- If yes, then simpler randomized algorithms can be made with proof of correctness instead of having to worry about deterministic algorithms.

Historical Context

- Randomness used in religious texts, old board games (dice-based), etc.
- Genesis of Probability
  - Luca Paciolo does some analysis
  - Pierre Fermat, Blaise Pascal — Huygens
  - "Problem of Points":
    - Coin is tossed w/ 100 coins buy-in
    - A wins if 10 Heads otherwise B wins if 10 Tails before 10 H
    - Prize is 200 coins.
    - Atmost 19 tosses required
    - Game gets interrupted at k (≤ 19) games then how should the prize money be distributed? m H, $k-m$ T

Luca Pacioli

- Win ratio

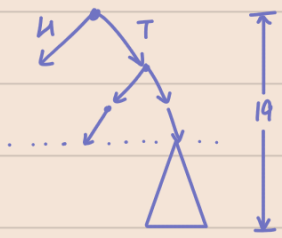
A : $\frac{m}{k} * 200$, B : $\frac{k-m}{k} * 200$

Unfair for $k \ll 19$ but okay for $k \sim 19$

Fermat proposed it, Pascal too but he also gave a method to count it : Pascal's Triangle

- Winning Chances

To count all games, technically games of depth ≤ 19 should be considered.



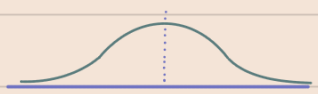
But, even if an "already-won" game is extended to depth 19, the winner remains the same.

∴ there is a unique winner in a game of depth 19.

- D. Bernoulli, 1713, "Art of Conjecture"
  - ↳ Weak Law of Large Numbers

One of the first landmark results in the field of probability.

- Central Limit Theorem
  - ↳ First proved by Laplace



Later proved by the likes of Gauss, Einstein, Turing, Lindelof, Chebyshev, Markov

- Borel and Cantelli prove the Strong Law of Large Numbers (c. 1909)
- Kolmogorov realizes that Radon-Nikodym Theorem is essentially Conditional Probability and drastically changes the scene in 1933

- Kolmogorov's theory widely accepted by 1935 except by Richard von Mises.
- Ville provides a counterexample to von Mises' theory in 1939. ↳ Martingale
- Instead, Church provides another definition of "Admissible Sequences" in 1940 but it too failed in the same counter-example.
- Kolmogorov provides another axiomatization in 1960's answering von Mises' concerns.

Now had access to Computability Theory and Information Theory, instead of earlier relying on Measure Theory.

Kolmogorov's Definition

x is random if the length of the smallest program outputting x has length $\geq |x|$

Naïve Set Theory

A set is an unordered collection of objects

- Notions of
  - $x \in A$ $\quad x \notin A$
  - ϕ : empty set
  - $A \cup B$ is a set s.t.
    - $x \in (A \cup B)$ iff $x \in A$ or $x \in B$
    - $x \in (A \cap B)$ iff $x \in A$ or $x \in B$
    - $x \in (A \setminus B)$ iff $x \in A$ or $x \notin B$

- $A \times B := \{(a, b) \mid a \in A, b \in B\}$

ordered

Difficult to define as numbers are defined using sets and we do not have numbers → No notion of position in an ordered pair

So, instead of $(a, b) := a \in A, b \in B$

Let us use

$(a, b) := \{\{a\}, \{a, \{b\}\}\}$

Uses depth as an implicit measure of count.

$\{a\} \equiv \bigcap_{s \in (a,b)} s$

$\{\{b\}\} \equiv \{a, \{b\}\} \setminus \{a\}$

$\{\{a\}, \{b\}\}$ does not preserve order

$\{\{a\}, \{a, b\}\}$ does not work with (a, a)

$\{\{a\}, \{\{b\}\}\}$ does not work because $(\{a\}, b) = (\{b\}, a)$

Now, we can define $A \times B$ as above

- A relation R from A to B is a subset of $A \times B$
- A function f from A to B is a relation R such that if $(a, b_1), (a, b_2) \in R$, then $b_1 = b_2$ & for every $a \in A$, there is some $b \in B$ such that $(a, b) \in R$
- Numbers can be defined as:
  $0 : \{\phi\}$, $1 : \{\phi, \{\phi\}\}$, $2 : \{\phi, \{\phi\}, \{\phi, \{\phi\}\}\}$ and so on.

Cardinality

- For a finite set A, cardinality of A is the number of elements in it.
- What about infinite sets?

Q) Given arbitrary sets A & B, can we say when the cardinality of A is less than or equal to the cardinality of B, i.e., when $|A| < |B|$?

For finite sets, $|A| \leq |B|$ iff

$\exists\ f: A \xrightarrow{1\text{-}1} B$. f is one-one if $\forall$ $x_1, x_2 \in A$ if $f(x_1) = f(x_2)$, then $x_1 = x_2$.

Define: For any 2 sets A & B, the relation $\leq$ by $|A| \leq |B|$ if $\exists\ f: A \xrightarrow{1\text{-}1} B$.

↳ Abstract notion ∵ no numbers are involved yet.

* A set A is infinite if $\exists\ f: \mathbb{N} \xrightarrow{1\text{-}1} A$

Ex: $2\mathbb{N}$, primes, $\mathbb{Q}$ ($\because \mathbb{N} \subseteq \mathbb{Q}$), $\mathbb{R}$ ($\because \mathbb{N} \subseteq \mathbb{R}$) are infinite

$\leq$: Reflexive $\forall$ A $\exists$ $id: A \xrightarrow{1\text{-}1} A$

Transitive if $|A| \leq |B|$, $|B| \leq |C|$, then $|A| \leq |C|$

$\exists\ f_1: A \xrightarrow{1\text{-}1} B$, $\exists\ f_2: B \xrightarrow{1\text{-}1} C$

$f_1 \circ f_2 : A \xrightarrow{1\text{-}1} C$

$\forall\ x_1, x_2 \in A$, $f_2(f_1(x_1)) = f_2(f_1(x_2))$

$\Rightarrow f_1(x_1) = f_1(x_2)$ (f_2 is 1-1)

$\Rightarrow x_1 = x_2$ (f_1 is 1-1)

Antisymmetry If $|A| \leq |B|$ & $|B| \leq |A|$, then $|A| = |B|$, i.e., if $\exists\ f: A \xrightarrow{1\text{-}1} B$ & $\exists\ g: A \xrightarrow{1\text{-}1} B$, then does $\exists\ h: A \xrightarrow[\text{onto}]{1\text{-}1} B$?

↳ Known as the Cantor-Schröder-Bernstein Theorem.

Proof: $a, g^{-1}(a), f^{-1}(g^{-1}(a)), \ldots$ ($a \in A$) may get stuck.

Disjoint s.t. $A = A_\infty \uplus A_A \uplus A_B$ ("Disjoint Union")

(i) A_∞ : Ancestry continues indefinitely
(ii) A_A : Ancestry terminates in A
(iii) A_B : Ancestry terminates in B

For,

A_∞ : both f & g^{-1} are 1-1 & onto from A_∞ to $f(A_\infty)$

A_A : f is onto from A_A to $f(A_A)$

A_B : g^{-1} is onto from A_B to $g^{-1}(A_B)$

So, h can be defined as:

$$h : A \xrightarrow{\text{1-1 onto}} B$$

$$h(a) = \begin{cases} f(a) & ; a \in A_A \\ g^{-1}(a) & ; a \in A_B \\ f(a) \text{ or } g^{-1}(a) & ; \text{otherwise} \end{cases}$$

But only one can be picked for A_∞ & needs to be stuck with

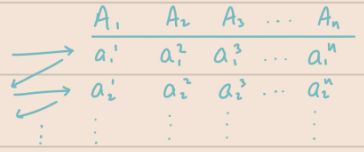
Non-constructive proof because h is not provided directly ∵ Membership test to check the set can be non-computable

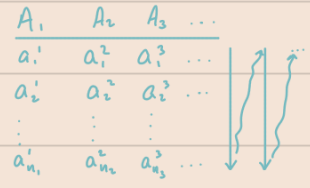
$\mathbb{N}$, $\mathbb{Q}$, $\mathbb{R}$

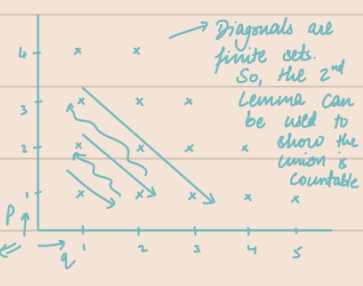
Theorem (Cantor) $|\mathbb{N}| = |\mathbb{Q}|$

Technique: Dovetailing → Used for showing that a set has the same cardinality as $\mathbb{N}$

- If A has a function $f : \mathbb{N} \xrightarrow{\text{onto}} A$ then is called countable.
- Lemma: Finite union of countable sets is countable.
- Lemma: Countable union of finite sets is countable.
- Lemma: Countable union of countable sets is countable.







· Thus, the result follows from the dovetailing argument.

Theorem (Cantor)
For any set S, $\nexists\ f : S \xrightarrow{\text{onto}} \mathcal{P}(S)$

Corollary: $|\mathcal{P}(\mathbb{N})| \geq |\mathbb{N}|$

Technique (Diagonalisation) → Used for showing that a set has cardinality larger than $\mathbb{N}$

Idea: 2 player game

1st player: $f : S \to \mathcal{P}(S)$
let $S = \mathbb{N}$ for the sake of demonstration

2nd player: f is not onto, i.e., $\exists\ T \in \mathcal{P}(\mathbb{N})$ but not in the range of f

Using 1's f, construct the set
$i \in C$ iff $i \notin f(i)$

If $f(x) = C$
↳ Now, if $x \in C = f(x)$
then $x \notin f(x)$ $\Rightarrow \in$
Also, if $x \notin f(x) = C$
then $x \in C$ $\Rightarrow \notin$

Thus, C is not in the range of f, though $C \in 2^{\mathbb{N}}$.
Thus, f is not onto.

Computability Basics

- A Turing Machine: is a 7-tuple $(Q, \Sigma, \Gamma, \delta, B, q_a, q_r)$
  - Q: Finite set of states
  - Σ: Finite input alphabet
  - Γ: Tape alphabet
  - δ: Transition function
  - B: Blank, $\{B\} \subseteq \Gamma \setminus \Sigma$
  - q_a: Accept state, $q_a \in Q$
  - q_r: Reject state, $q_r \in Q$
  - $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$

- Church-Turing Thesis:
  Any reasonable model of computation (read programming language) cannot express any more programs than the Turing Machine model.
- A Turing machine is a model of universal computation.
  We use the Church-Turing thesis, and adopt a C-like language as our model of computation.
- A language $L \subseteq \{0,1\}^*$ is a set of finite-length strings.
- $\{0,1\}^* = \{\lambda, 0, 1, 00, 01, \ldots\}$
  - ↳ Countably Infinite Set ($|\mathbb{N}|$)
  - ↳ Number of languages is uncountably infinite ($|\mathbb{R}|$)
- $L \subseteq \{0,1\}^*$ is acceptable if there is a Turing Machine M such that

$\forall\ x \in \{0,1\}^*$

$x \in L \Rightarrow M(x)$ accepts & halts

$x \notin L \Rightarrow M(x)$ does not accept & halt (may go on indefinitely)

- $L \subseteq \{0,1\}^*$ is Turing decidable if L, L^c are Turing acceptable.

Ex: $L \subseteq \{0,1\}^*$ is Turing decidable if $\exists\ M$ s.t. $\forall\ x \in \{0,1\}^*$,

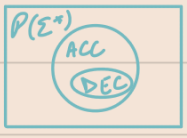
$x \in L \Rightarrow M$ accepts & halts

$x \notin L \Rightarrow M$ rejects & halts

Turing Machines → C program → Finite strings → Countably infinite. But, # of languages are uncountably infinite.

* There are languages which are not Turing acceptable.
* There are acceptable languages which are not decidable, eg. Halting Problem (H).

Corollary: H^c is not acceptable.



- **Definition**: A one-way infinite 2-tape Turing-Machine computes $f: \Sigma^* \to \Sigma^*$ if the machine, when presented with x in the input tape in the initial configuration, halts and produces $f(x)$ BBBB*... on the output tape, when $f(x)$ is defined ($f(x)\downarrow$) and runs forever if $f(x)\uparrow$.

- f is said to be partial computable if $\exists$ a Turing Machine M which computes f in the preceding sense.
- f is said to be total computable if $dom(f) = \Sigma^*$ { The machine always halts }

- Computing from $f: \mathbb{N} \to \mathbb{N}$:
  Encode $\lambda, 0, 1, 00, 01, 10, 11, \ldots$
  (Sort by length, then lexicographically)
  $n \to n+1$ in Binary, Drop leading 1
  $\leftarrow 0, 1, 2, 3, 4, 5, 6, \ldots$

- S_n: Standard Encoding of $n \in \mathbb{N}$
- $f: \mathbb{N} \to \mathbb{N}$ is partial computable if $\exists M$ $\forall n \in \mathbb{N}$ s.t. on input S_n, it outputs $S_{f(n)}$.

- Bit-doubling function:
  $bd(\lambda) = \lambda$
  $bd(w0) = bd(w)00$
  $bd(w1) = bd(w)11$
- The pairing function:
  $\langle , \rangle : \Sigma^* \times \Sigma^* \to \Sigma^*$
  is defined by
  $\langle x, y \rangle = bd(x)\,01\,bd(y)\,01$

Lemma: $\langle , \rangle$ is a prefix code (or a prefix-free code), i.e., if w is an output of the pairing function, then no proper prefix of w is an

(prefix-free encodings are good for circuits (from the POV of Electrical Engineers))

output of $\langle , \rangle$.

- In computability theory, multi-variate functions can be replaced by uni-variate functions (using pairing functions or otherwise) without much complication.

  So, our functions shall have a single input (our program can parse it into different arguments if need be).

* Partial computable function $\cong$ Acceptable language

Total computable function $\cong$ Decidable language

- A language $L \subseteq \{0,1\}^*$ is computably enumerable if it is either finite, or if $\exists$ a total computable bijection $f: \mathbb{N} \xrightarrow[\text{onto}]{1\text{-}1} L$

Theorem: A language is C.E. iff it is Turing Acceptable.

Proof: ($\Rightarrow$)

If L is finite, then it is CE by definition. Also, $\because L$ is finite, it is decidable $\Rightarrow L$ is acceptable $\therefore$ The theorem holds trivially for finite languages.

Suppose L is CE and infinite. Let $f: \mathbb{N} \xrightarrow[\text{onto}]{1\text{-}1} L$ be a total computable bijection.

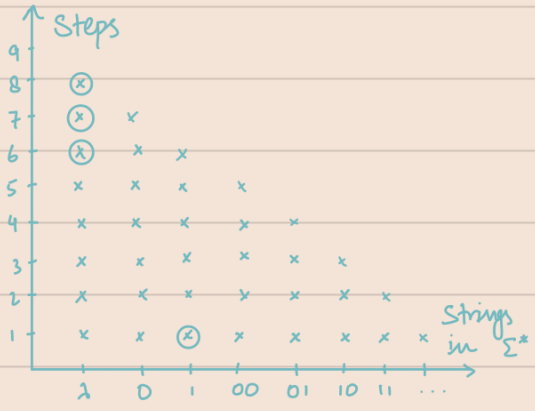
Algo:

1. Input x
2. i = 1
   2.1 if f(i) = x, accept
   2.2 else i ← i+1

(⇐)

Suppose L is acceptable by M. Also, let L be infinite. Now, we use dovetailing.

1. Input i
2. Until i distinct strings are accepted, dovetail over $\Sigma^* \times$ Steps
3. Output the i^{th} accepted string



Not exactly the same as order of enumeration, uses both length & # of steps taken to accept it. It does not depend on L, rather M

* An infinite language L is CE in monotone increasing order iff it is decidable.
- Corollary: An infinite CE language has an infinite decidable subset.
- Corollary: Every acceptable language (that is not decidable) is inherently unsortable.

If it could be sorted, it would be decidable.

Theorem. (Kleene's Normal Form Theorem) There is a 3-argument partial

computable function C and a 1-argument partial computable function U such that any 1-argument part. comp. function f_e can be expressed as

C simulates M_e, the TM corresponding to e, on the input n. z is the first time when C halts. $U[\mu z: C(e,n,z)]$ is the output of M_e when the computation has halted, effectively $f_e(n)$ if it is $\downarrow$.

$$f_e(n) = \overbrace{U[\mu z : C(e,n,z)=0]}^{\text{Output tape during halt state}}$$

e → index; n → argument; μz : least z s.t. first time/step when computation halts

Theorem. There is a partial computable function that is not total computable.

Proof.

Let us define the following partial computable function $f: \mathbb{N} \to \mathbb{N}$ as:

	1	2	3	4
f_1	0	0	0	
f_2	0	1	1	
f_3	0	1	0	
⋮				

$$f(n) = U(\mu z\,[C(n,n,z)=0]) + 1$$
$$\equiv \begin{cases} f_n(n)+1, & f_n(n)\downarrow \\ \text{undefined}, & \text{otherwise} \end{cases}$$

Clearly then, by definition, $f(n)$ is partially computable.

Now, take an enumeration of partially computable functions, $(f_e)_{e\in\mathbb{N}}$.

We shall show that not every func. in the enumeration is total.

i.e., $\exists$ an f in $(f_e)_{e\in\mathbb{N}}$ which is partially computable but not totally computable.

If $f \in (f_e)_{e\in\mathbb{N}}$ then $\exists\ e_0 \in \mathbb{N}$ such that $f_{e_0} = f$. But, on feeding the input e_0 to f, we get

$$f(e_0) = f_{e_0}(e_0) + 1 > f_{e_0}(e_0) = f(e_0) \quad \Rightarrow\Leftarrow$$

Thus, f is not present in any enumeration of totally computable functions. So, we have a partially computable function that is not totally computable. □

Plain Kolmogorov Complexity

- What is a finite length random binary string?

s_1 : 00...00 → 10k 0's

s_2 : 10...11 → outcome of 10k independent unbiased coin flips

(1) s_2 is "unpredictable"
Prediction/Betting algorithm ←
Given a prefix, cannot guess the next bit with probability more than ½.

Short description ≡ Compression ←
s_2 cannot be compressed ⇒ random
Shall use this notion

(2) There is no "visible pattern" in s_2

(3) Talking about probability does not make sense, both strings have the same probability.
Difficult to make this work for finite strings, but okay for infinite-length strings.
But, we can use the idea of a "typical set" eg. the string should be ½ 0's & ½ 1's is a "typical set" containing s_2 & not s_1. Elements of "typical set" are called "random".

* For infinite-length strings, all 3 approaches work.

- What is a compressor?
  - Any valid program which outputs a given string $x \in \{0,1\}^*$ is a compressed version of x

- If $|p| < |x|$, then x is compressible.
- If every program outputting x, has length $\geq |x|$, then x is called incompressible ($\equiv$ random).

* Both partially computable functions & totally computable functions are enumerable/countable, but TCF's are not CE (difficult ∵ it is the same as the halting problem.)

- So, there is a computable enumeration of $\varphi_1, \varphi_2, \ldots$ of PCF's.

  $\exists$ a UTM $\mathcal{U}$ s.t. for any $\langle s_n, x \rangle$ does $\varphi_n(x)$, i.e., $\mathcal{U}(\langle s_n, x \rangle) \equiv \varphi_n(x)$ (when defined, have the same value, otherwise both are undefined)

Definition: Let x, y, p be strings. Any partial computable function φ such that

$$\varphi(\langle y, p \rangle) = x = p(y)$$

(UTM; input to p; p → program)

is a description of x. Then, complexity of x given y is

C_φ (Complexity subject to the choice of UTM φ. ∵ there can be multiple UTM's.)

$$C_\varphi(x \mid y) = \min \{ p : \varphi(\langle y, p \rangle) = x \}$$

($C_\varphi(x \mid y)$: complexity subject to input y)

b

$$C_\varphi(x) \triangleq C_\varphi(x \mid \lambda)$$

($C_\varphi(x)$: complexity of x; $C_\varphi(x \mid \lambda)$: complexity subject to the empty string)

Example: $C_\varphi(w \mid w) \leq O(1)$

Consider: p
1. INPUT x
2. OUTPUT x

$$|p| = \underbrace{2 \cdot k}_{\text{Assume the language is something like asm}} = O(1)$$

(Fixed ops, Fixed vars (registers))

Example: $C_\varphi(w) \triangleq C_\varphi(w \mid \lambda) \leq |w| + O(1)$

Consider: q {
1. INPUT x
2. OUTPUT w

$|q| = 2 \cdot K + \underbrace{|w|}_{\text{w is hardcoded into our program q}} \leq |w| + O(1)$

Example: $w_1 = 0^{10,000}$

q_1 {
1. INPUT x
2. FOR i=1 to 10,000
   2.1. PRINT 0

$C_\varphi(w_1) \leq |q_1| \leq \log(10,000) + O(1)$
$= \log |w_1| + O(1)$

q_2 {
1. INPUT x
2. $n = 1$
3. FOR i=1 to 4
   3.1. $n = n * 10$
4. FOR i=1 to n
   4.1. PRINT 0

$C_\varphi(w) \leq |q_2| \leq \log 4 + O(1)$
$= \log \log 10,000 + O(1)$
$= \log \log |w| + O(1)$

Theorem. (Invariance)

There is a universal partially computable function $\psi : \Sigma^* \to \Sigma^*$ s.t. $\forall$ PCF's $\varphi : \Sigma^* \to \Sigma^*$ $\exists$ c_φ s.t. $\forall x, y \in \Sigma^*$

$C_\psi(x \mid y) \leq C_\varphi(x \mid y) + c_\varphi$ → just a description of the index of φ

Can fix ψ w/o having to worry about some UTM's cheating (by hardcoding, etc.)

Proof.

Let $\psi : \Sigma^* \to \Sigma^*$ be the UPCF such that on input $\langle s_n, y, p \rangle$

computes $\varphi_n(\langle y, p\rangle)$.

$\because$ φ in the statement is a PCF,

$\exists\ n: \varphi = \varphi_n$

$\psi(\langle s_n, y, p\rangle) = \varphi_n(\langle y, p\rangle) = p(y) = x$

i.e., $\langle s_n, y, p\rangle$ is a description of x for ψ.

$$|\langle s_n, y, p\rangle| = |bd(s_n)\,01\,bd(y)\,01\,p|$$
$$= 2*|s_n| + 2 + |\langle y, p\rangle|$$
$$= c_\varphi + C_\varphi(x|y)$$
$$\Rightarrow C_\psi(x|y) \leq c_\varphi + C_\varphi(x|y)$$

- For every string $x \in \{0,1\}^*$, $C(x)$ is well-defined.

Theorem: $\exists\ c \quad \forall\ x \quad C(x) \leq |x| + c$

Proof. Let $x \in \{0,1\}^*$ be given.

Consider the program p: 1. PRINT x

$\because$ x is hardcoded in p,

$|p| \leq |x| + c$

Hence, $C(x) \leq |x| + c$ □

Corollary: $\forall\ x \in \{0,1\}^*$ $C(x)$ is well-defined and finite. Similarly, $C(x|y) \leq |x| + c$

Theorem: $C: \{0,1\}^* \longrightarrow \mathbb{N}$ is uncomputable

Intuition 1: Why not bruteforce all strings based on $C(x) \leq |x| + c$? Can use dovetailing but Halting Problem continues to be an issue.

Suppose M_i has put x on the output tape and halted. But, there is an M_j, $j<i$, that has not halted ⇒ We can never know that M_j will halt ⇒ Never know if M_j outputs x ⇒ Never compute $C(x)$

Intuition 2: Berry's Paradox

5 words ← 123: One hundred and twenty-three

Consider the smallest number not expressible using ten words

Paradox arises because there is no fixed convention of "expressing" numbers

∴ the above sentence expresses the number in nine sentences

There is no such number!

Proof. (Berry's Paradox, Chaitin '69)

Assume C is computable.

Consider the first string x such that $C(x) > n$.

→ $\exists\, x$ because if $\nexists\, x$, then $|\{0,1\}^*| \leq 2^{n+1}-1$

Not just uncomputable, but even unpredictable.

Consider the program:

1. FOR $w = \lambda, 0, 1, 00, \ldots$
   1.1. IF $C(w) > n$, OUTPUT w

This program always halts (C is computable) and $\forall\, n$, it outputs the first string such that $C(w) > n$

Also, $|p| \approx \log n + c$

So, the output w:

$$C(w) \leq \log n + c$$
$$\Rightarrow n < \log n + c \quad \forall\, n$$

But, this is false for all large enough n.

$\therefore$ C is not computable $\square$

Lemma. (Landmark Lemma)

$$\exists\, c \quad \forall\, x \in \{0,1\}^* \quad \forall\, h \in \mathbb{N}$$
$$|C(x+h) - C(x)| \leq 2|h| + c$$

Proof. $C(x+h) \leq C(x) + |h|$

Consider the following program to output $x+h$:

If it is easy to output x, then it is not too hard to output nearby strings either: just describe x, then add h to it.

1. INPUT x
2. $y = p_x(\lambda)$ — Let p_x be the shortest program for x
3. OUTPUT $y + h$

$\Rightarrow C(x+h) \leq C(x) + |h|$

Similarly, we can show that

$$C(x) \leq C(x+h) + |h|$$

★ Most strings are incompressible.

— "Abundance of Randomness"

Lemma. The number of strings of length n with complexity $\leq n-c$ is at most $2^{n-c+1} - 1$

Proof. If $C(x) \leq n-c$, then there exists a program p s.t. $|p| \leq n-c$ and $\varphi(\langle \lambda, p \rangle) = p(\lambda) = x$

The number of such programs is $\leq \sum_{i=0}^{n-c} 2^i = 2^{n-c+1} - 1$ □

- Similarly, for any fixed $y \in \Sigma^*$, $\forall n$ the number of strings of length n with complexity $C(x|y) \leq n-c$ is at most $2^{n-c+1} - 1$.
- Only $1/2^c$ fraction of Σ^n can be reduced by c bits.
- Any lossless compressor $f: \Sigma^* \to \Sigma^*$ must be 1-1.

Problems with Plain K

1) Lack of Subadditivity

$\forall\ x, y \in \Sigma^*$

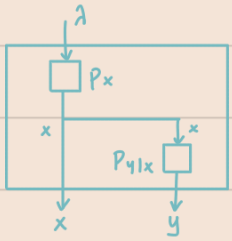
$$C(xy) \not\leq C(x) + C(y|x) + O(1)$$

Shannon Entropy

$$H(X, Y) = H(X) + H(Y|X)$$

Why?

Intuitively, let p_x be a program for x, $p_{y|x}$ for y on input x.



If $p_x \cdot p_{y|x}$ is a parseable program, then we can get $x = p_x(\lambda)$, $y = p_{y|x}(x)$.

$$C(x, y) \leq |p_x| + |p_{y|x}|$$
$$= C(x) + C(y|x) + O(1)$$

Need to find a way to encode their concatenation — something like

$\text{bd}(p_x)\,01\,p_{y|x} \Rightarrow C(x, y) \leq (2)\,C(x) + C(y|x) + O(1)$

But, $\forall\ X, Y$ $\quad H(X|Y) \leq H(X)$

$\forall\ x, y \in \Sigma^*$ $\quad C(x|y) \leq C(x) + O(1)$

2) C is non-monotone over prefixes.

Self-delimiting Kolmogorov Complexity

— Chaitin '75

Prefix Code

A set of strings $P \subseteq \Sigma^*$ is a prefix code (or equivalently prefix-free code) if $v \in P$ and $w \sqsubset v$ (w is a proper prefix of v), then $w \notin P$.

· Can be finite, can be infinite (leaves of a binary tree; 1*0)

Lemma. (Kraft Inequality)
If P is a prefix-free code,
$$\sum_{x \in P} \frac{1}{2^{|x|}} \leq 1$$
($\sum_{x \in \Sigma^*} \frac{1}{2^{|x|}} = \sum_n 2^n \cdot \frac{1}{2^n} = \sum_n 1 = \infty$)

Proof. Experiment – Flip an unbiased coin independently & repeatedly until we hit a string in P.
What is the probability that the experiment terminates?
$$= \sum_{x \in P} \frac{1}{2^{|x|}}$$
(because it is a prefix-free code)
Valid experiment, valid sample space, valid event
$\Rightarrow \mathbb{P}(\text{event}) \leq 1 \Rightarrow \sum_{x \in P} \frac{1}{2^{|x|}} \leq 1$

· If we restrict programs to be elements of some prefix set P, then $\forall\, p, q \in P$ pq can be unambiguously parsed to $p \cdot q$

* 3 basic theorems:
1. Complexity is uncomputable
2. Subadditivity
3. Abundance of randomness

Invariance theorem is not included here, since it is a precursor to these theorems (it is a part of the definition)

· Let $\varphi_1, \varphi_2, \ldots$ be a comp. enumeration

of all unary partially comp. functions mapping $\{0,1\}^*$ to $\{0,1\}^*$. Let $T_1, T_2, \ldots$ be the corresponding TM's.

We define a new C.E. of P.C.U.F. $\varphi_1', \varphi_2', \ldots$ s.t. if $dom(\varphi_i)$ is a prefix-free set, then $\varphi_i' = \varphi_i$, otherwise $dom(\varphi_i')$ is some prefix-free set.

Not an easy question, since it is atleast as difficult as answering the Halting Problem.

Consider T_i which computes φ_i. We define T_i'. If $dom(T_i)$ is prefix-free, then $dom(T_i') = dom(T_i)$ & $T_i' = T_i$. Otherwise, suppose the input to T_i is $b_0 b_1 \ldots$

The algorithm does not answer the question whether a set is prefix-free or not, it instead just tries to construct a prefix-free set.

1. See whether there is an extension z of λ s.t. $T_i(z)\downarrow$
2. If yes:
   - 2.1. If $z = \lambda$, then accept λ and halt
   - 2.2. Else, read b_0
   - 2.3. If $\exists$ an extension z of b_0 s.t. $T_i(z)\downarrow$, and $z = b_0$, then accept and halt
   - 2.4. Continue similarly

- If $dom(T_i)$ is a prefix-free set, then for every b s.t. $T_i(b)\downarrow$, $T_i'(b)$ will be equal to $T_i(b)$.
- Otherwise, $dom(T_i')$ is a prefix-free set, but $T_i' \neq T_i$.

Theorem. $\exists$ univ. part. comp. fn. ψ with

prefix-free domain s.t. for any part. comp. unary fn. φ with prefix-free domain, $\exists c : \forall x, y \in \{0,1\}^*$,

$$C_\psi(x|y) \leq C_\varphi(x|y) + c$$

Do not have to read $|x|+1$ bits. can know where it ends in $|x|$ bits — it limits itself

Self-limiting Kolmogorov Complexity

$$\boxed{\begin{aligned} K(x|y) &= C_\psi(x|y) \\ K(x) &= C_\psi(x|\lambda) \end{aligned}}$$

1) K is uncomputable as $K = C_\psi$, and C_ψ is uncomputable.
2) K is subadditive.

$\forall x, y \in \{0,1\}^*$,

$$K(xy) \leq K(x) + K(y|x) + O(1)$$

If p_x is a program from $\mathcal{P}$ for x, & p_y is a program from $\mathcal{P}$ for y given x, then $p_x p_y$ is an unambiguously parseable program for x, y.

Moreover,

$$\mathcal{P}' = \{ p_u p_v \mid p_u \mapsto u,\ p_v \mapsto v,\ p_u, p_v \in \mathcal{P} \}$$

is a prefix-free set.

$\forall p_x \mapsto x,\ p_y \mapsto y,\ p_x, p_y \in \mathcal{P}$.

$$K(x,y) \leq |p_x| + |p_y| + O(1)$$

Considering the shortest programs for x, y, we get,

$$K(x,y) \leq K(x) + K(y|x) + O(1)$$

"Symmetry of Information"

We'll have an analogue of Shannon's Entropy → A theory independent of probability

- Is $K(x,y) \geq K(x) + K(y|x) + O(1)$?
- Is $K(x) - K(x|y) \approx K(y) - K(y|x)$?

We show: $K(x) - K(x \mid y, K(y)) \approx K(y) - K(y \mid x, K(x))$

and $K(x, y) \geq K(x) + K(y \mid x, K(x)) + O(1)$

Definition: Let $x \in \{0,1\}^*$. The algorithmic (sub)probability of x is:

$$im(x) = \sum_{p \in \tilde{P},\ u(p)=x} \frac{1}{2^{|p|}}$$

Feeding p to the UTM U produces the output x

$$\sum_{x \in \{0,1\}^*} im(x) = \sum_{x \in \{0,1\}^*} \sum_{\substack{p \in \tilde{P} \\ u(p)=x}} \frac{1}{2^{|p|}} \leq \sum_{p \in \tilde{P}} \frac{1}{2^{|p|}} \leq 1$$

[Kraft-inequality for $\tilde{P}$]

One of the terms will be $1/2^{K(x)}$

$$\Rightarrow \forall\, x \in \{0,1\}^* \quad im(x) = \sum_{\substack{p \in \tilde{P} \\ u(p)=x}} \frac{1}{2^{|p|}} \geq \frac{1}{2^{K(x)}}$$

$$\Rightarrow K(x) \geq -\log im(x)$$

- Levin showed that $\exists\, c_{coding}$ $\forall\, x \in \{0,1\}^*$

$$K(x) \leq -\log im(x) + c_{coding}$$

$$\Rightarrow K(x) \approx -\log im(x)$$

Theorem (Levin's Coding Theorem)

$$\exists\, c_{coding} \quad \forall\, x \in \{0,1\}^* \quad K(x) \leq -\log im(x) + c_{coding}$$

Definition: An indexed enumerator is a Turing Machine M taking an input $z \in \{0,1\}^*$ such that $M(z)$ is a computably enumerable language

Output of M is a CE set of CE languages ~ Set of Sets

Definition: A request set is a set of pairs of (strings, numbers) (x, n) such that $\sum_{(x,n)} \frac{1}{2^n} \leq 1$

Definition: A conditional requester R is an indexed enumerator such that $\forall\, x \in \{0,1\}^*$, $h(R(x))$ is a request set.

Ex: $R(\lambda) = \{(0100, 1000), (\lambda, 4), (01, 2), \ldots\}$
$R(0) = \{(\lambda, 1), (101, 3), \ldots\}$

Theorem [Levin]
For each conditional requester R, $\exists\, c_R$ s.t. $\forall\, z \in \{0,1\}^*$ $\forall\, (x,n) \in L(R(z))$
$K(x \mid z) \leq n + c_R$

(Index of the sequence of requests given to R; demand: assign some code to x of length n)

Levin's Coding Theorem (Converse to Kraft's Inequality)
$\exists\, R$, conditional request set s.t. $\forall\, z \in \{0,1\}^*$, $R(z)$ enumerates a c.e. set of the form $\{(x,n) : x \in \{0,1\}^*, n \in \mathbb{N}\}$ s.t. $\sum_{(x,n)} \frac{1}{2^n} \leq 1$

Proof. Take the leftmost valid string w of length n, s.t. no proper prefix or extension of w has been assigned.
If (y, m) is a demand for $L(R(z))$, then there is a length m string in the binary tree that is available.
Let w be the k^{th} string

assigned, $|w| = n$. This corresponds to some $[k/2^n, k+1/2^n)$. The assigned set of codes C correspond to a finite, disjoint union of intervals.

Suppose the length of intervals assigned so far is $1/2^{n_1} + 1/2^{n_2} + \ldots + 1/2^{n_j}$

Ex: $C^c = [\lambda]$ → Subtree rooted at λ

- $(0010, 4)$
  $C^c = [0001] \cup [001] \cup [01] \cup [1]$
- $(1, 5)$
  $C^c = [00011] \cup [001] \cup [01] \cup [1]$
- $(110, 2)$
  $C^c = [00011] \cup [001] \cup [1]$
- $(\lambda, 1)$
  $C^c = [00011] \cup [001]$

At every stage in the algorithm, C^c can be written as a finite disjoint union of trees, of strictly decreasing lengths.

Assume that all strings of length n are unavailable — i.e. $\forall\ w \in \{0,1\}^*$, some prefix of w or an extension of w is already assigned. By the above claim, the length of every interval in C^c is at least $(n+1)$.

The length of the intervals corresponding to C^c can be

written as → $0 < k_1 < k_2 < \dots < k_m$

$$1/2^{n+1} + 1/2^{n+1+k_1} + 1/2^{n+1+k_2} + \dots + 1/2^{n+1+k_m}$$

$$< 1/2^{n+1} + 1/2^{n+2} + \dots = 1/2^n$$

(Strict because the sum is finite)

Hence, the adversary has $< 1/2^n$ remaining in the budget.

- It is a bit non-trivial to show:
$$K(x) \leq -\log m(x) + O(1)$$
But first, observe that,
$$\sum_x m(x) = \sum_x \sum_{\substack{p \in P \\ U(p)=x}} 1/2^{|p|} \leq \sum_{p \in P} 1/2^{|p|} \leq 1$$

- Attempt:
Request Set: $\hat{L} = \{(x, -\log m(x)) : x \in \{0,1\}^*\}$
KI: $\sum_x 2^{+\log m(x)} = \sum_x m(x) \leq 1$
But $\hat{L}$ is not c.e. $\therefore$ $m(x)$ is not computable.

- m is computably approximable from below
(Dovetail over programs until one of them outputs x & halts)
$\Leftrightarrow -\log m$ is computably approximable from above.

- Attempt:
$$L = \{(x, n) : -\log m(x) < n+1\}$$
L is c.e. because if $-\log m(x) < n+1$, then a TM can detect it at some stage in the upper approximation of $-\log m(x)$.
Also,
$$\sum_{n > -\log m(x)} 1/2^n = 1/2^{1-\log m(x)} + 1/2^{2-\log m(x)} + \dots$$
$$= 1/2^{-\log m(x)}$$

$\therefore \forall x \exists$ prefix-free code p (by

coding algorithm) s.t.

$$|e_p(x)| \leq -\log m(x) + 1$$

$$\Rightarrow K(x) \leq -\log m(x) + 1 \quad \square$$

- Levin's Coding Theorem: $K(x) \approx -\log m(x)$

Subadditivity

$$\forall x, y \in \{0,1\}^* \quad K(x,y) \leq K(x) + K(y|x) + O(1)$$

But, what about

$$K(x,y) \geq K(x) + K(y|x) + O(1)$$

We show:

$$K(x,y) \geq K(x) + K(y|x) - K(K(x)) + O(1)$$

Lemma:

$$\exists\, c_{proj} \text{ s.t. } \forall\, x \in \{0,1\}^*, \quad im(x) \geq \frac{1}{2^{c_{proj}}} \sum_{\substack{p \in P \\ U(p) = (x,-)}} \frac{1}{2^{|p|}}$$

Proof:

$$im(x) = \sum_{\substack{p \in P \\ U(p) = x}} \frac{1}{2^{|p|}}$$

Now, observe that if p outputs (x, y), then

π_1 picks the first argument in a tuple, i.e., it projects its first input.

$$\pi_1(U(p)) = \pi_1(x,y) = x$$

The length of the program:

$$|\pi_1| + |p| = c_{proj} + |p|$$

Now,

$$m(x) = \sum_{\substack{p \in P \\ U(p) = x}} 1/2^{|p|} \geq \sum_{\substack{p \in P \\ U(p) = (x,y) \\ \pi_1(U(p)) = x}} 1/2^{|p| + |\pi_1|}$$

$$\geq 1/2^{c_{proj}} \sum_{p \in P,\ U(p) = (x,-)} 1/2^{|p|} \quad \square$$

Theorem:

$\exists\ c$ s.t. $\forall\ x, y \in \Sigma^*$,

$$K(y|x, K(x)) \leq K(x, y) - K(x) + c$$

Note that $K(y|x, K(x)) \leq K(y|x)$. So, we are showing a weaker inequality. If we'd show the inequality w/ $K(y|x)$, it'd be tighter, and we'd be done.

Proof:

We form a request set to code $(y|x, K(x))$. We design the machine which on input $z_{a,n} = (a, n)$ enumerates the request set:

$$L(R(z_{a,n})) = \{ \langle b, |p| - n + c_{proj} + c_{code} \rangle : U(p) = (a, b) \}$$

$L(R(z_{a,n}))$ is c.e.

When $z = (x, K(x))$,

$$L' = L(R(z_{x,K(x)})) = \{ \langle b, |p| - K(x) + c_{proj} + c_{code} \rangle : U(p) = (x, b) \}$$

L' satisfies the Kraft inequality:

$$\sum_{b \in \Sigma^*} \frac{1}{2^{|p| - K(x) + c_{proj} + c_{code}}} = 2^{K(x) - c_{proj} - c_{code}} \sum_{b \in \Sigma^*} \frac{1}{2^{|p|}}$$

$$\leq 2^{K(x) - c_{proj} - c_{code}} \cdot 2^{c_{proj}} \cdot im(x)$$

$$= 2^{K(x) - c_{code}} \cdot im(x) = 1$$

$$K(x) = -\log im(x) + c_{code}$$

$\Rightarrow L'$ is a conditional request set

Hence, for every $b \in \{0,1\}^*$,

$K(b|z) \leq |p| - K(x) + c_{proj} + c_{code}$ will be true for all p.

$U(p) = (x, b)$. When p is the smallest such program, $|p| = K(x, b)$

$\therefore\ K(b|z) \leq K(x, b) - K(x) + O(1)$

i.e., $K(b|x, K(x)) \leq K(x, b) - K(x) + O(1)$

$K(b|x, K(x))$ same as $K(b|x) - K(K(x))$ ∵ we can find $K(x)$ if we are provided a program to compute it, specifically a program of length $K(K(x))$

□

Applications of Finite Strings

· K-incompressible strings have roughly half zeroes and half ones.
∴ An effective version of the weak law of large numbers. ∴

$$\frac{{}^nC_{n/2}}{2^n} \to 0 \qquad \frac{\sum {}^nC_{n/2 \pm \epsilon n}}{2^n} \to 1$$

(ϵ is constant; for any ϵ, number of strings where $|\#0\text{'s} - n/2| \leq \epsilon n$)

(just a counting argument, ∵ Intersection of two large sets is also large)

· Most strings are K-incompressible.
Most strings are nearly balanced.
~ Most K-incompressible strings have roughly ½ 0's & ½ 1's.
It is in fact the case that ALL K-incompressible strings have roughly ½ 0's & ½ 1's ∴

(Only problem is that K-incompressibility is not computable)

(If a string is K-incompressible, then it also passes any practical test/notion of randomness)

Theorem: Every k-incompressible string has approximately ½ 0's.

WLLN: $\forall\ \epsilon, \delta > 0$ $\forall$ sufficiently large n,

$$\mathbb{P}_{x \in \{0,1\}^n}\left(\left| \frac{\sum_{i=0}^{n-1} x_i}{n} - \frac{1}{2} \right| \leq \epsilon \right) \geq 1 - \delta$$

SLLN: $$\mathbb{P}_{x \in \{0,1\}^\infty}\left[\lim_{n \to \infty} \sum_{i=0}^{n-1} \frac{x_i}{n} = \frac{1}{2} \right] = 1$$

· Peter Gacs 1987: Annals of probability

Def: A function $d : \Sigma^n \to \mathbb{R}_{\geq 0}$ is called

a deficiency of randomness function if $\forall k$

$$\mathbb{P}\left[d(x) > k\right] < 1/2^k$$

x randomly sampled from Σ^n $\longleftarrow$ $x \sim \Sigma^n$

- d α How NOT-random a string is
- This, in some sense, is a test for deficiency of randomness.
- Proof: The argument goes as follows:
  $\exists$ def. test d s.t.
  1. Every k-incompressible string has low d-value
  2. Every string deviating from WLLN has higher d-value.

Def: A real valued function $f: \Sigma^* \to \mathbb{R}$ is called lower semi-computable if $\exists$ a computable function $\hat{f}: \Sigma^* \times \mathbb{N} \to \mathbb{Q}$ s.t.

1. Monotonicity in 2^{nd} variable, i.e., $\forall n \ \forall x \in \Sigma^*$,
   $$\hat{f}(x, n) < \hat{f}(x, n+1) < f(x)$$
2. Convergence:
   $$\lim_{n \to \infty} \hat{f}(x, n) = f(x)$$

Def: A lower semi-computable deficiency of randomness is called a Martin-Löf test.

* We call a Martin-Löf test d_0 universal if for any MLT d, $\exists\, c \ \forall x \in \{0,1\}^n$
$d(x) < d_0(x) + c$

Theorem (Gács)

The function $d : \Sigma^* \to \mathbb{R}$, defined by:

$$d_0(x) = |x| - K(x \mid |x|) \quad \{\geq |x| - K(x)\}$$

is a universal MLT.

- Consequentially, $d(x) < d_0(x) + c$

$$< |x| - K(x \mid |x|) + c$$

$$\Leftrightarrow K(x \mid |x|) < |x| - d(x) + c$$

- With the universality theorem, all we need is an ML-test for d.
- Consider any $x \in \{0,1\}^n$,

  Let $\theta_x = |\{i : x_i = 1\}| / n$

  Then, we define the likelihood function

$$L_x : \Sigma^n \to [0,1]$$

$$L_x(y) = \theta_x^{|\{i : y_i = 1\}|} (1 - \theta_x)^{|\{i : y_i = 0\}|}$$

- $x = (01)^{(\frac{n}{2})} \rightsquigarrow L_x(x) = 1/2^n$
- $x = 0^n \rightsquigarrow L_x(x) = 1$
- $x = 1^n \rightsquigarrow L_x(x) = 1$

* L_x can be interpreted as a probability space over Σ^n — coin tosses with a coin of head-bias θ_x

$$\Leftrightarrow \sum_{y \in \Sigma^n} L_x(y) = 1 \quad \forall \quad x \in \Sigma^n$$

Kullback-Leibler Divergence

Let $(p_1, p_2, \ldots, p_n)$ and $(q_1, \ldots, q_n)$ be two probability distributions.

The Kullback-Leibler divergence of $\vec{p}$ from $\vec{q}$, denoted $D(p \| q)$ is:

$$D(p \,||\, q) := \sum p_i \log\left(\frac{p_i}{q_i}\right)$$
$$= \left(-\sum p_i \log q_i\right) - \left(-\sum p_i \log p_i\right)$$

* Huffman Encoding:

$(p_1, \ldots, p_n)$ ← Actual distribution

$$-L \sum p_i \log p_i \cong \ell(\text{COMPRESSED})$$

TEXT → COMPRESSED

If we instead think the distribution is $(q_1, \ldots, q_n)$, and encode according to $\vec{q}$, then the expected length will be

$$-L \sum p_i \log q_i > -L \sum p_i \log p_i$$

Text is still sampled from the dist. $\vec{p}$

$$\Rightarrow -\sum p_i \log q_i > -\sum p_i \log p_i$$
$$\Rightarrow D(p \,||\, q) > 0 \iff \vec{p} \neq \vec{q}$$

- $\therefore D(p \,||\, q) \geq 0$ and $D(p \,||\, q) > 0$ iff $\vec{p} = \vec{q}$
- D is not a metric however as $D(p \,||\, q) \neq D(q \,||\, p) \quad \forall \quad \vec{p}, \vec{q}$

- Every k-incompressible string has nearly $1/2$ 0's. What about for $x \in \Sigma^n$?

$$d(x) = \left| \frac{N(0|x)}{n} - \frac{1}{2} \right| ? \quad \left(\frac{N(0|x)}{n} - \frac{1}{2} \right)^2 ?$$

↳ $N(0|x) = |\{i : x_i = 0\}|$

- Given $x \in \Sigma^n$, we define the empirical cointoss probability (defined by x) as $\forall \; y \in \Sigma^n$

$$P_x(y) = P_x^{N(1|y)} (1 - P_x)^{N(0|y)}$$

where $P_x = \dfrac{N(1|y)}{n}$

- If x has $1/2$ 0's, then $p_x(x) = 1/2^n$
  If x has 0 0's, then $p_x(x) = 1$
- Note that $p_x(\cdot)$ defines a probability space over Σ^n.

- Define

$$\begin{aligned} d(x) &= (\log p_x(x) + n) - \log(n+1) \\ &= (\log p_x(x) - \log 1/2^n) - \log(n+1) \\ &= \log \frac{p_x(x)}{1/2^n} - \log(n+1) \end{aligned}$$

$\log \frac{p_x(x)}{1/2^n} \approx D(p_x \| U)$; $\log(n+1)$: Normalization Constant

1) d is computable, hence lower semicomputable.

2) Need to show $\forall\ k \in \mathbb{N}$

$$\mathbb{P}_{x \sim \Sigma^n}[\, d(x) > k \,] < 1/2^k$$

↓

We instead show $\sum_{x \sim \Sigma^n} U(x)\, 2^{d(x)} \leq 1$

↳ $\mathbb{E}[2^{d(x)}] \leq 1$

$\Rightarrow \mathbb{P}[\, 2^{d(x)} > 2^k \,] < \frac{\mathbb{E}[2^{d(x)}]}{2^k}$

$\Rightarrow \mathbb{P}[\, d(x) > k \,] < \frac{1}{2^k}$ ↑ Markov's Inequality

and then the result follows as:

$$\begin{aligned} 1 &\geq \sum U(x)\, 2^{d(x)} \\ &= \sum_{d(x)>k} U(x)\, 2^{d(x)} + \sum_{d(x)\leq k} U(x)\, 2^{d(x)} \\ &> \sum_{d(x)>k} U(x)\, 2^k + 0 = 2^k \sum_{d(x)>k} U(x) = 2^k\, \mathbb{P}[d(x)>k] \end{aligned}$$

$$\Rightarrow \mathbb{P}[\, d(x) > k \,] \leq 1/2^k \qquad \square$$

Now,

$$\sum_{x \in \Sigma^n} U(x)\, 2^{d(x)} = \sum_{x \in \Sigma^n} 1/2^n \, \frac{p_x(x)}{1/2^n} \, \frac{1}{n+1}$$

$$= \frac{1}{n+1} \sum_{x \in \Sigma^n} p_x(x) = \frac{1}{n+1} \sum_{x \in \Sigma^n} \left(\frac{N(1|x)}{n}\right)^{N(1|x)} \left(\frac{N(0|x)}{n}\right)^{N(0|x)}$$

$$= \frac{1}{n+1} \sum_{x \in \Sigma^n} \left(\frac{N(1|x)}{n}\right)^{N(1|x)} \left(1 - \frac{N(1|x)}{n}\right)^{n-N(1|x)}$$

$$= \frac{1}{n+1} \sum_{N(1|x)} \sum_{x \in \Sigma^n} p_x^{N(1|x)} (1-p_x)^{n-N(1|x)}$$

$$= \frac{1}{n+1} \sum_{N(1|x)} \binom{n}{N(1|x)} \left(\frac{N(1|x)}{n}\right)^{N(1|x)} \left(1 - \frac{N(1|x)}{n}\right)^{n-N(1|x)}$$

$$\leq \frac{1}{n+1} \sum_{N(1|x)} \sum_{k=0}^{n} \binom{n}{k} \left(\frac{N(1|x)}{n}\right)^{k} \left(1 - \frac{N(1|x)}{n}\right)^{n-k}$$

$$\leq \frac{1}{n+1} \sum_{N(1|x)=0}^{n} 1 = \frac{1}{n+1} \cdot (n+1) = 1$$

□

Randomness of Infinite Binary Sequences

Mathematical Robustness

- When can you call a string random?

Incompressibility
All prefixes are K-incompressible

⟷

Constructive Measure Theory (Martin-Löf)
("statistical")
SLLN is a necessary condition for randomness

↕

Martingale / Betting / Unpredictability [Schnorr]

Measure Theory

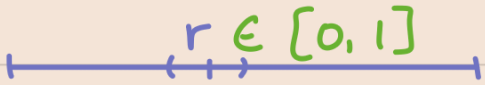
- We will define only measure zero sets & measure 1 sets.
- $q \geq 0 \wedge q = 0 \Leftrightarrow q \geq 0 \wedge \forall \epsilon > 0.\ q < \epsilon$

A set $S \subseteq [0,1]$ has Lebesgue measure 0 if $\forall \epsilon > 0$ $\exists$ open intervals $O_{1,\epsilon}, O_{2,\epsilon}, \ldots$ such that

$$\bigcup_{n=1}^{\infty} O_{n,\epsilon} \supseteq S \quad \text{and} \quad \sum_{n=1}^{\infty} \ell(O_{n,\epsilon}) < \epsilon$$

1) Any singleton set $S \subseteq [0,1]$ has Lebesgue measure 0.

Let $S = \{r\}$.



Given an $\epsilon > 0$,

consider the set $(r - \epsilon/4, r + \epsilon/4)$.

Hence, $\mu(S) = 0$

2) Any finite set $S = \{r_1, \ldots, r_n\}$ has $\mu(S) = 0$

Given an ϵ, consider the sequence

$O_{1,\epsilon} = (r_1 - \epsilon/4n, r_1 + \epsilon/4n)$

$O_{2,\epsilon} = (r_2 - \epsilon/4n, r_2 + \epsilon/4n)$

$\ldots$

$$\sum_i \ell(O_{i,\epsilon}) = \sum_i \epsilon/2n = \epsilon/2 < \epsilon$$

$$\Rightarrow \mu(S) = 0$$

3) Any countably infinite set $S = \{r_i \mid i \in \mathbb{N}\}$ has $\mu(S) = 0$

Given an ϵ, consider

$$O_{i,\epsilon} = \left(r_i - \frac{\epsilon}{4^i}, r_i + \frac{\epsilon}{4^i}\right)$$

$$\sum_i \ell(O_{i,\epsilon}) = \sum_i \frac{2\epsilon}{4^i} = \frac{2\epsilon/4}{1 - 1/4} = \frac{2\epsilon}{3} < \epsilon$$

$$\Rightarrow \mu(S) = 0$$

Corollary: $\mathbb{Q} \cap [0,1]$ has measure 0

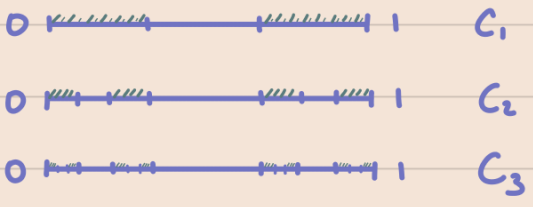
4) Some uncountable sets have measure 1.

Eg: $\mu([0,1]) = 1$ since $\mu([0,1]^c) = \mu(\phi) = 0$

5) There are uncountable sets with measure 0.

Eg: Cantor Set

$$C = \bigcap_{i=1}^{\infty} C_i$$



$\mu(C) < \ldots < \mu(C_n) < \ldots < \mu(C_2) < \mu(C_1)$

$\mu(C_1) = 2/3, \quad \mu(C_2) = 4/9, \quad \ldots$

$\mu(C_n) = 2^n/3^n \longrightarrow 0$ as $n \to \infty$

$\therefore \mu(C) = \lim_{n\to\infty} \mu(C_n) = 0$

- A set $S \subseteq [0,1]$ has $\mu(S) = 0$ if $\forall\, m \in \mathbb{N}$ $\exists$ $\langle O_{n,m} \rangle_{n=1}^{n=\infty}$ of open intervals such that $\bigcup_{n=1}^{\infty} O_{n,m} \supseteq S$, and $\sum_{n=1}^{\infty} l(O_{n,m}) < \frac{1}{2^m}$.

  $\epsilon \in \mathbb{R} \cap [0,1]$ ← ~
  ↳ Continuous variable → Problematic in CS → Replace ϵ wt $1/2^m < \epsilon$
  → We have an equivalent definition that is CS-friendly.

- $\mu(S) = 1$ iff $\mu(S^c) = 0$

- A set $S \subseteq \{0,1\}^{\infty}$ has constructive measure 0 if $\exists$ a Turing Machine M such that $\forall\, m \in \mathbb{N}$, $\forall\, n \in \mathbb{N}$

  $M(m,n) = \{q_1, q_2\}$ $[O_{n,m} = (q_1, q_2)]$

such that $q_1 < q_2$, $q_1, q_2 \in \mathbb{Q}$ and

$$\bigcup_{n=1}^{\infty} O_{n,m} \supseteq S \quad \text{and} \quad \sum_{n=1}^{\infty} l(O_{n,m}) < \frac{1}{2^m}$$

- S has constructive measure 1 if S^c has constructive measure 0.
* $S = \{\omega\}$ does not have constructive measure 0 even though it is a (classical) measure 0 set.
  ($\because$ ω (Chaitin's Constant) is not upper approximable)
* There are uncountably many (classically) measure 0 sets ($\because$ there are uncountably many singleton subsets of $[0,1]$) but, there are only countably many constructive measure 0 sets ($\because$ there are countably many TM's).

Q. Can we say if $S_1, S_2, \ldots$ are constructive measure 0 sets, then $\bigcup_{i=1}^{\infty} S_i$ is a constructive measure 0 set?

No. Because, we need the enumeration of S_i to be computable.

Q. Can we say if $S_1, S_2, \ldots$ is a computable enumeration of constructive measure 0 sets, then $\bigcup_{i=1}^{\infty} S_i$ is a constructive measure 0 set?

Yes.

Proof Sketch:

Turing Machine T s.t. $T(i) = S_i$

$\forall i$ $M(i, n, m)$ will output $\mathcal{U}^i_{n,m}$. Construct a single M for $\left[\bigcup_{i=1}^{\infty} S_i\right]$. $M(i, n, m) = \mathcal{U}^i_{n, m+i}$

Rational Open Interval ⟵

Not possible mathematically because of the following argument:
- Let largest set be S.
- Then $S \neq [0,1]$ ∵ $\mu([0,1]) = 1$
- $[0,1] \setminus S \neq \phi$
- Let $x \in [0,1] \setminus S$
- $\mu(\{x\}) = 0$, $\mu(S) = 0$
- $\Rightarrow \mu(\{x\} \cup S) = 0$
- $\Rightarrow S$ is not the largest

$$\sum_{i=1}^{\infty} \sum_{n=1}^{\infty} \ell(U^i_{n,m+i}) = \sum_{i=1}^{\infty} 1/2^{m+i} = 1/2^m$$

Also, $\forall\, i \quad \bigcup_{i=1}^{\infty} U^i_{n,m} \supseteq S_i$

Hence, $\bigcup_{i=1}^{\infty} \bigcup_{n=1}^{\infty} U^i_{n,m} \supseteq \left[\bigcup_{i=1}^{\infty} S_i\right]$

Theorem [Martin-Löf]
There is a largest constructive measure 0 set.

Proof Sketch:
The set of all constructive measure 0 sets is c.e.
By the previous fact, their union is a constructive measure 0 set, & it must be the largest since we have enumerated all TMs.

Corollary: There is a smallest constructive measure 1 set, R.
Take the complement of the largest constructive measure 0 set w.r.t. $[0,1]$, it will be a measure 1 set.

Definition: Every element of R is called Martin-Löf random.

Rationale:
Example Theorem in Probability, SLLN (Strong Law of Large Numbers):
$X_i : \Omega \to \{0,1\}$
Let $X_0, X_1, \ldots$ be a sequence of

IID's (Independent & Identically Distributed) variables valued 0-1 randomly.

Then,

$$\lim_{n \to \infty} \frac{\sum_{i=0}^{n-1} X_i(x)}{n} = \mathbb{E}[X]$$

Not the same as saying it holds for all sequences $\{x_i\}$ $\therefore$ $(T, T, T, \ldots)$ exists, but it has probability 0.

with probability 1.

Intuitively, these sequences constitute the smallest set, whose elements we can call "random".

Similarly, R can be thought of as a set which passes all "constructive measure 1" tests. It can then be thought of as an "intersection" of "constructive measure 1 sets", and thus, its elements are "random".

Martingales

Martingale Strategy

Game: If outcome of the toss matches your bet, you get double the money, otherwise you lose the bet.

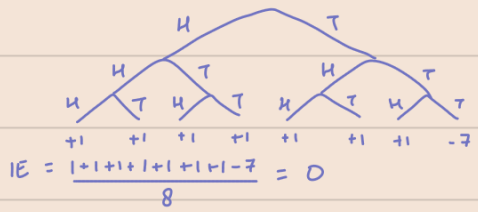
Strategy:

- Bet Re 1 on toss being H
  - Outcome = H → Walk out w/ +1
  - Outcome = T
  - Bet Re 2 on H
    - Outcome = H → Walk out w/ +1
    - Outcome = T
    - Bet Re 4 on H ...

Claim: Can always walk out with +1.

· Does the strategy work?

No, winnings on expectation remain 0



∴ A martingale is a stochastic process $X_0, X_1, X_2, \ldots$ such that

$\forall\ i \quad \mathbb{E}[X_i \mid X_0, X_1, \ldots, X_{i-1}] = X_{i-1}$

— Ville, 1939

Martingales (Binary Setting)

A function $m: \{0,1\}^* \to [0, \infty)$ is called a martingale if

(1) $m(\lambda) \leq 1$

(2) for every $w \in \{0,1\}^*$,

$$m(w) = \frac{m(w0) + m(w1)}{2}$$

Note: This is a martingale under uniform probability on $\{0,1\}^\infty$

· A martingale models a "fair" betting game.

Eg: A martingale that wins unbounded money on $\{0^\infty\}$.

$m(\lambda) = 1$. $m(w0) = 2m(w)$,

$m(w1) = 0 \longrightarrow m(w) = \frac{m(w0)+m(w1)}{2}$

$\Rightarrow m(0^n) = 2^n$

Eg: A martingale that wins unbounded

money on all infinite binary sequences x such that,

$$\lim_{n\to\infty} \sum_{i=1}^{i=n} x_i / n = 3/4$$

$m(\lambda) = 1, \quad m(w0) = 1/4 \cdot 2\, m(w)$

$m(w1) = 3/4 \cdot 2\, m(w)$

$m(w) = (m(w0) + m(w1))/2$

Suppose x is a sequence such that $\lim_{n\to\infty} \sum_{i=1}^{i=n} x_i / n$ exists and is equal to $3/4$. Let $\epsilon > 0$ be fixed. Then, for all large enough n,

$$\left| \frac{\sum_{i=1}^{i=n} x_i}{n} - \frac{3}{4} \right| < \epsilon$$

$$\Rightarrow \frac{\sum_{i=1}^{i=n} x_i}{n} > \frac{3}{4} - \epsilon \Rightarrow m(x[1\ldots n]) > \left(\frac{3}{2}\right)^{(3/4-\epsilon)n} \left(\frac{1}{2}\right)^{(1/4+\epsilon)n}$$

Success of a Martingale

We say that a martingale $m : \{0,1\}^* \to [0, \infty)$ wins on a $x \in \{0,1\}^\infty$ if

~least upper bound ← $\limsup_{n\to\infty} m(x[1\ldots n]) = \infty$

- Every set S of reals has a supremum, which may not be in S,

  eg: $\sup \{x \in \mathbb{R} \mid x \leq 2\} = 2$,

  $\sup \{x \in \mathbb{R} \mid x < 2\} = 2$ → Different than maximum of a set

- A sequence of reals $\langle r_n \rangle_{n=1}^{\infty}$ has a limit r if $\forall\, \epsilon > 0$, for all large enough n, $|r_n - r| < \epsilon$

↕

$\langle r_n \rangle_{n=1}^{\infty}$ does not have a limit r if $\exists\ \epsilon > 0$ there are infinitely many n such that, $|r_n - r| \geq \epsilon$

- Every sequence has a limsup and a liminf defined as:

$$\limsup_{n\to\infty} r_n = \lim_{n\to\infty} \left(\sup_{k>n} r_n \right)$$

$$\liminf_{n\to\infty} r_n = \lim_{n\to\infty} \left(\inf_{k>n} r_n \right)$$

* Any monotone decreasing sequence of reals has a limit; the limit may be $-\infty$, but it exists.

- $\sup \phi = -\infty$ $\qquad$ $\inf \phi = +\infty$

- A martingale $m : \{0,1\}^* \to [0, \infty)$ is called constructive (lower semicomputable) if $\exists$ a total computable function $\hat{m} : \{0,1\}^* \times \mathbb{N} \to \mathbb{Q}^+$ such that

(1) [monotonicity]

$\forall\ n\ \forall\ w$

$$\hat{m}(w, n) \leq \hat{m}(w, n+1) \leq m(w)$$

(2) [convergence]

$\forall\ n$ $\qquad$ $\lim_{n\to\infty} \hat{m}(w, n) = m(w)$

- A martingale $m : \{0,1\}^* \to [0, \infty)$ succeeds on $x \in \{0,1\}^\infty$ if $\limsup_{n\to\infty} m(x[0 \ldots n-1]) = \infty$

$\Leftrightarrow\ \forall\ N\ \exists\ n\ \ m(x[0 \ldots n-1]) = \infty$

Theorem [Ville, 1939]
A measurable set S has Lebesgue measure 0 if and only if some martingale succeeds on every element of S.

Theorem [Schnorr, 1972]
A set $S \subseteq \{0,1\}^*$ has constructive measure 0 iff $\exists$ lsc martingale $m : \{0,1\}^* \to [0, \infty)$ such that m succeeds on all elements in S.

Proof Sketch:

Suppose $S \subseteq \{0,1\}^*$ is such that some lsc martingale $m : \{0,1\}^* \to [0,\infty)$ succeeds on all $x \in S$.

$\Leftrightarrow \forall x \in S \; \forall N \; \exists n : m(x[0 \ldots n-1]) > N$

$M(m,n) = u_{m,n}$ open intervals s.t.

(i) $\forall m \quad \bigcup_{n=1}^{\infty} u_{m,n} \supseteq S$

(ii) Fix $m \in \mathbb{N} \quad \sum_{n=1}^{\infty} \ell(u_{m,n}) < 1/2^m$

$u_{N,n} = \{ x \in \{0,1\}^\infty \mid m(x[0 \ldots n-1]) > 2^N \}$

- $u_{N,n}$ are open sets due to lower semicontinuity of m
- These sets are constructive because m is lsc.

$\sum \ell(u_{N,n}) < 1/2^N$ by Kolmogorov inequality.

Conversely, let S be a constructive measure 0 set.

$u_{m,n}$

$\forall m \quad \bigcup_{n=1}^{\infty} u_{m,n} \supseteq S$

$$\forall\, m \qquad \sum_{n=1}^{\infty} l(u_{m,n}) < 1/2^m$$

Idea: ∵ Construct a martingale that bets that its input will be in $u_{m,n}$. ∴

Ideas in the proof

- The Kolmogorov Inequality

Lemma: Let $m: \{0,1\}^* \to [0,\infty)$.
Then for any $n \in \mathbb{N}$,

$$\sum_{w \in \{0,1\}^n} m(w) \cdot \frac{1}{2^{|w|}} \leq 1 \qquad [= m(\lambda)]$$

Proof:
By induction on n. Clearly,

$$m(\lambda) \cdot \frac{1}{2^{|\lambda|}} = m(\lambda) \cdot \frac{1}{2^0} \leq 1$$

Suppose the claim holds for n,
i.e., $\sum_{w \in \{0,1\}^n} m(w) \cdot \frac{1}{2^n} \leq 1$

Then,

$$\sum_{v \in \{0,1\}^{n+1}} m(v) \cdot \frac{1}{2^{n+1}}$$

$$= \sum_{w \in \{0,1\}^n} [m(w0) + m(w1)] \cdot \frac{1}{2^{n+1}}$$

$$= \sum_{w \in \{0,1\}^n} \frac{[m(w0) + m(w1)]}{2} \cdot \frac{1}{2^n}$$

$$= \sum_{w \in \{0,1\}^n} m(w) \cdot \frac{1}{2^n} \leq 1$$

□

- **Lemma:** Let $X \subseteq \{0,1\}^{\leq n}$ be a prefix-free set of strings. Then,
$$\sum_{w \in X} m(w) \cdot \frac{1}{2^{|w|}} \leq 1$$

- **Theorem [Kolmogorov Inequality]** Let $X \subseteq \{0,1\}^*$ be a prefix-free set. Then $\sum_{w \in X} m(w) \cdot \frac{1}{2^{|w|}} \leq 1$

Proof:

Let X_n be $X \cap \{0,1\}^{\leq n}$.

Then,
$$\sum_{w \in X} m(w) \cdot \frac{1}{2^{|w|}} \geq \sum_{w \in X_n} m(w) \cdot \frac{1}{2^{|w|}}$$

Infact, [Monotone Convergence from below, $X_1 \subseteq X_2 \subseteq X_3 \ldots \subseteq X$, $\lim_{n \to \infty} X_n = X$]
$$1 \geq \lim_{n \to \infty} \sum_{w \in X_n} m(w) \cdot \frac{1}{2^{|w|}}$$
$$= \sum_{w \in X} m(w) \cdot \frac{1}{2^{|w|}}$$

- **Corollary [Kolmogorov Inequality]** Let $m : \{0,1\}^* \to [0, \infty)$ be a martingale. Then, if $X \subseteq \{0,1\}^*$ is a prefix-free set, for any $N \in \mathbb{R}_+$,
$$P[w \in X \mid m(w) > N] < 1/N.$$

Proof:
$$\sum_{w \in X} m(w) \cdot 1/2^{|w|} \leq 1$$

Hence,
$$1 \geq \sum_{\substack{w \in X \\ m(w) > N}} m(w) \cdot \frac{1}{2^{|w|}} + \sum_{\substack{w \in X \\ m(w) \leq N}} m(w) \cdot \frac{1}{2^N}$$
$$\geq N \cdot \sum_{\substack{w \in X \\ m(w) > N}} \frac{1}{2^{|w|}} + 0$$

$= N \cdot \mathbb{P}[w \in X \mid m(w) > N]$

$\Leftrightarrow \mathbb{P}[w \in X \mid m(w) > N] < 1/N$ □

Ideas in the converse direction

- **Lemma:** If $m_1, m_2 : \{0,1\}^* \to [0, \infty)$ are two lsc martingales, $\exists\ m : \{0,1\}^* \to [0, \infty)$ lsc martingale that succeeds on all sequences on which either m_1 or m_2 succeeds.

  Proof:

  $\forall\ w \in \{0,1\}^*$,

  $m(w) = 0.5\,(m_1(w) + m_2(w))$ □

- **Theorem [Schnorr, 1972]**

  If $m_i : \{0,1\}^* \to [0, \infty)$, $i \in \mathbb{N}$ are a computable enumeration of lsc martingales, then $\exists\ m : \{0,1\}^* \to [0, \infty)$ that succeeds on all sequences that any m_i succeeds on.

  Proof:

  $\forall\ w \in \{0,1\}^*$, $m(w) = \sum_{i=1}^{\infty} \frac{1}{2^i} m_i(w)$

  Then, $m(\lambda) = \sum_{i=1}^{\infty} \frac{1}{2^i} m_i(\lambda) \leq \sum_{i=1}^{\infty} \frac{1}{2^i} = 1$

  By linearity, $m(w) = \frac{m(w0) + m(w1)}{2}$ for any $w \in \{0,1\}^*$

  Suppose $m_i(x[0 \ldots n-1]) > N$.

  Then, $m(x[0 \ldots n-1]) > N/2^i$

  Hence, if,

$$\limsup_{n\to\infty} m_i(x[0\ldots n-1]) = \infty,$$

then

$$\limsup_{n\to\infty} m(x[0\ldots n-1]) = \infty$$

□

- By the same theorem, ∃ a universal martingale.
  ⇔ This martingale can be thought of as an amalgamation of all the martingales
  ⇒ It wins on some strings
  ⇒ There is some set of strings on which a martingales can win
  ⇒ Its complement contains strings random
  (This set is the same as Martin-Löf's R)

Properties of Infinite Random Sequences

Van Lambalgen's Theorem

1) If $A = A_0 A_1 \ldots$ & $B = B_0 B_1 \ldots$ are infinite binary sequences, then
$A \oplus B := A_0 B_0 A_1 B_1 \ldots$

Can consult the tape any no. of times

2) Oracle Turing Machine
Let $Y \in \Sigma^\infty$ be an infinite binary sequence. M^Y, an oracle TM, decides a language L if $M^Y(x) = L(x)$ for every $x \in \Sigma^*$.

M is a TM w/ an extra tape it can consult with Y written on it

$\therefore L$ is decidable in/w.r.t. Y $\therefore \equiv L \leq_T Y$

Ex: H^C is decidable in H; $H_C \leq_T H$

- $A \in \Sigma^\infty$ is random relative to $B \in \Sigma^\infty$ if $\exists c \; \exists N \; \forall n > N \quad K^B(A[0 \ldots n-1]) \geq n + c$
- If X is ML random, then $\bar{X}$ is not random relative to X.
- Given X, an ML random, the set of sequences random relative to X will have constructive measure 1.

Theorem: A is ML random and B is ML random relative to B iff $A \oplus B$ is ML random.

Corollary: A is random relative to B iff B is random relative to A.

1st direction: If B is incompressible and A is incompressible given B, then $A \oplus B$ is incompressible.

To show: $2n + c \leq K(A \oplus B[0 \ldots 2n])$

Given: $n + c_1 \leq K(B[0 \ldots n-1])$

$n + c_2 \leq K^B(A[0 \ldots n-1])$

Is the following true?

$K(B[0 \ldots n-1]) + K^B(A[0 \ldots n-1]) < K(A \oplus B[0 \ldots 2n])$

Let $X \upharpoonright n := X[0 \ldots n-1]$

Or this?

$K(B \upharpoonright n) + K(A \upharpoonright n \mid B \upharpoonright n) < K(A \oplus B \upharpoonright 2n+1)$ □

The standard symmetry of information fails to prove this!

$\hookrightarrow K(B\restriction n) + K(A\restriction n \mid B\restriction n, \underline{K(B\restriction n)}) < K(A \oplus B \restriction 2n+1)$

Given the Lemma, the prev. theorem holds ∵ $B[0\ldots n-1]$ is incompressible

Lemma: Let x be a finite string with $K(x) > |x| - c$, and let y be a finite string. Then,

$$|x| + K(y \mid x) \leq K(x, y) + O(1)$$

Proof: Let P_i be a program which outputs a pair of strings. Consider the following program:

1. INPUT x
2. LET $p_i \longmapsto (a, b)$
3. IF $a = x$, THEN OUTPUT$(b, |p_i| - |x| + c)$

$$\sum_{b\in\Sigma^*} \sum_{\substack{\pi_1(U(p_i))=x \\ \pi_2(U(p_i))=b}} 1/2^{|p_i|-|x|+c} = 2^{|x|-c} \sum_{b\in\Sigma^*} \sum_{\substack{\pi_1(U(p_i))=x \\ \pi_2(U(p_i))=b}} 1/2^{|p_i|}$$

$$< 2^{|x|-c+c'} \sum_{b\in\Sigma^*} 1/2^{K(x,b)}$$

As in Levin's coding theorem

$$\leq 2^{|x|-c+c'}/2^{K(x)} \leq 1$$

∵ x is incompressible

□

2nd direction: If B is not MLR or A is not MLR relative to B, then $A \oplus B$ is not MLR

Idea: $K(x, y) \leq K(x) + K(y \mid x)$

↪ Subadditivity

Weakly compressible ~ What if B is compressible but

Highly incompressible ~ A is incompressible relative to B?

$$K(B\restriction n) \leq n - \log\log n$$

$$K(A\restriction n \mid B\restriction n) \geq n + \log n$$

$$K(A \oplus B \restriction 2n) \overset{?}{\approx} 2n + \log n - \log\log n \Rightarrow\Leftarrow$$

Let's try a different approach.

Proof: [Use very succinct prefix codes]

Case I: (B is incompressible)

Assume $K(B[0\ldots n-1]) \leq n-c$.

Let σ be a shortest program from the prefix-free set of programs P which outputs $B[0\ldots n-1]$

$\Rightarrow |\sigma| \leq n-c$

[If we use the shortest program for $A[0\ldots n-1]$, let's say τ, and use $\sigma\tau$ for $A \oplus B[0\ldots 2n-1]$, $|\tau\sigma|$ maybe $\geq n-c+n+\log n \geq 2n$

$\Rightarrow A \oplus B$ is MLR $\Rightarrow\Leftarrow$]

Lesson: Better to print A directly rather than using a program to print it.

$|A|=n$
$K(A)>n$
$\Downarrow$
$|A|<K(A)$

For the same, we construct a new prefix free set of programs

Consider the set

$$Q_n = \{\tau\rho \mid \tau \in P,\ |\rho| = n\}$$

Then, Q_n is a prefix-free set.

Also, $\sigma\cdot(A[0\ldots n-1]) \in Q_n$

$$|\sigma\cdot(A[0\ldots n-1])| = |\sigma| + n$$
$$= n-c+n = 2n-c$$

and $A \oplus B[0\ldots n-1]$ is producible from $\sigma\cdot(A[0\ldots n-1])$

Case II: (A is compressible relative to B)

Assume

$$K(A\upharpoonright n \mid B\upharpoonright m) \leq n-c$$

Because extra information can always be ignored
$\Rightarrow K(A\upharpoonright n \mid B\upharpoonright m)$
$\leq K(A\upharpoonright n \mid B\upharpoonright n)$
when $m \geq n$

We can assume $m \geq n$.

Let the code for $A[0\ldots n-1]$ given $B[0\ldots m-1]$ be τ,

$|\tau| \leq n-c$

Define

$Q_{m,n} = \{ \nu p \mid \nu \in \mathcal{P},\ |p| = 2m-n \}$

We show that $A \oplus B[0 \dots 2m-1]$ is compressible.

$Q_{m,n}$ is a prefix-free set.

$\tau \cdot (B[0 \dots m-1])(A[n \dots m-1])$ is a code for $A \oplus B[0 \dots 2m-1]$.

$\because B[0 \dots m-1]$ is the minimal length of the oracle needed to compute $A[0 \dots n-1]$, τ can say where $B[0 \dots m-1]$ ends and $A[n \dots m-1]$ starts. $\square$

- A disjunctive binary sequence X is one where every finite string w occurs in X.
- Claim: Every MLR is disjunctive.

Proof Sketch: If some pattern does not occur in $X \in \{0,1\}^\infty$, then there is an ML test $\langle G_m \rangle_{m \in \mathbb{N}}$ (G_m are uniformly c.e. open and $\mu(G_m) < 2^{-m}$) s.t. $X \in \bigcap_m G_m$

Note: $\mu(\bigcap_m G_m) = \lim_{m \to \infty} 2^{-m} = 0$

Eg: Pick 01

We want an ML test capturing all sequences in which 01 is absent.

Consider, for $i \in \mathbb{N}$, $S_i : \{0,1\}^\infty \to \{0,1\}$ defined by

$$S_i(X) = \begin{cases} 1, & \text{if } X_i X_{i+1} \neq 01 \\ 0, & \text{otherwise} \end{cases}$$

We want $\mathbb{P}[\bigcap_i^{\infty} (S_i = 1)]$

Note: S_i & S_{i+1} are not (always) mutually independent

Note: $S_1, S_3, S_5, \ldots$ are mutually independent.

$\Rightarrow \mathbb{P}[\bigcap_{i=1}^{\infty} (S_i = 1)] \leq \mathbb{P}[\bigcap_{i=1}^{\infty} S_{2i-1}]$

$= \prod_{i=1}^{\infty} \mathbb{P}[S_{2i-1} = 1]$

$= \lim_{n \to \infty} (3/4)^n = 0$

$\Rightarrow$ Similarly, it can be shown for any string other than 01.

Also, the probability of a string not having a particular pattern, say 01, being 0 implies that the set of strings not having the particular pattern have constructive measure 0.

$\Rightarrow$ The set of all possible patterns is countably infinite and computably enumerable.

$\Rightarrow$ The sets of strings not having specific patterns are countably infinite and c.e, each with constructive measure 0.

$\Rightarrow$ Their union has constructive measure 0.

$\Rightarrow$ The complementary set, the set of strings not missing any pattern, has constructive measure 1.

$\Rightarrow$ Infinite Monkey Theorem $\square$

· Van-Lambalgen Theorem ~ MLR good
Kučera-Gács Theorem ~ MLR bad → ∃ some fools

Kučera-Gács Theorem

$\forall\ Y \in \{0,1\}^\infty\ \exists\ R \in MLR$ s.t. $Y \leq_T R$

Proof [Merkle, Mihalovič]

Lemma [Space Lemma]

Given a rational $\delta > 1$ and a positive integer k, we can compute a length $l(\delta, k)$ such that for any l.s.c. martingale $m : \{0,1\}^* \to [0, \infty)$ and any string $\sigma \in \{0,1\}^*$,

$$|\{\tau \in \{0,1\}^{l(\delta,k)} \mid m(\sigma\tau) \leq \delta \cdot m(\sigma)\}| \geq k$$

Proof:

By Kolmogorov inequality, we have,

LHS: Probability
Inequality: Markov ←

$$\frac{|\{\tau \in \{0,1\}^{l(\delta,k)} \mid m(\sigma\tau) > \delta \cdot m(\sigma)\}|}{2^{l(\delta,k)}} \leq \frac{1}{\delta}$$

Set $l(\delta, k) = \left\lceil \log \frac{k}{1 - 1/\delta} \right\rceil$

($k \uparrow$ or $\delta \downarrow \Rightarrow l(\delta, k) \uparrow$)

Then,

$$|\{\tau \in \{0,1\}^{l(\delta,k)} \mid m(\sigma\tau) \leq \delta \cdot m(\sigma)\}| \geq (1 - 1/\delta) \cdot 2^{l(\delta,k)} \geq (1 - 1/\delta) \frac{k}{1 - 1/\delta} = k$$

□

[Continuing]

$R \in MLR \Leftrightarrow$ Universal l.s.c. martingale

M does not win on R
$\Leftrightarrow$ $\limsup_{n\to\infty} M(R[0\ldots n) < \infty$
$\Rightarrow$ $\liminf_{n\to\infty} M(R[0\ldots n-1]) < \infty$

Now, let M be a universal l.s.c. martingale.

Let $r_0 > r_1 > r_2 > \ldots$ be a sequence of rationals all greater than 1 so that $\prod_i r_i$ converges.

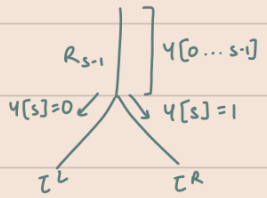
We construct R in stages $s = 1, 2, \ldots$ At stage s, we have a prefix R_s already determined. Now we have to determine a $\tau \in \{0,1\}^{\ell(r_s, 2)}$.

By the Space Lemma, there are atleast 2 paths of length $|R_{s-1}| + \ell(r_s, 2)$ s.t. $M(R_{s-1}\,\tau) \leq r_s \cdot M(R_{s-1})$.

Let the leftmost path be τ^L and rightmost path be τ^R.

Inductively, assume that $Y[0\ldots s-1]$ are Turing computable from R_{s-1}.

Our Task: Encode $Y[s]$ into the s^{th} stage extension of R_{s-1}.



If $Y[s] = 0$, then $R_s = R_{s-1}\,\tau^L$
If $Y[s] = 1$, then $R_s = R_{s-1}\,\tau^R$

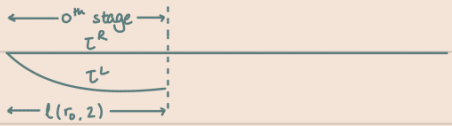
(i) R is MLR

At every stage s,
$M(R\upharpoonright_s) \leq r_s\, M(R\upharpoonright_{s-1}) \leq r_s\, r_{s-1}\, M(R\upharpoonright_{s-2})$
$\leq \ldots \leq \prod_0^s r_s \leq \prod_0^\infty r_s < \infty$

$\Rightarrow \liminf_{n\to\infty} M(R\upharpoonright_n) < \infty$
$\Rightarrow R$ is MLR

(ii) R computes Y

Given R, $\exists$ an oracle TM M that accepts Y.



If τ^R was chosen at stage 0, then $Y[0] = 1$, otherwise $Y[0] = 0$.

But, we only know τ, and not whether it is τ^L or τ^R. ($\because$ we know $l(r_0, 2)$)

Obs: $m(\sigma) > r_0 \cdot m(\lambda)$ is Turing acceptable, but $m(\sigma) \leq r_0 \cdot m(\lambda)$ is not $\because$ m is l.s.c.

Obs: All paths to the left of τ^L satisfy $m(\sigma) > r_0\, m(\lambda)$; All paths to the right of τ^R satisfy $m(\sigma) > r_0 \cdot m(\lambda)$.

Idea: Simulate m on all the paths left of τ as well as to the right of τ.

One by one, paths get rejected $\because$ m is l.s.c. If all paths to the left of τ get rejected, then $\tau = \tau^L$. otherwise, if all paths to the right of τ get rejected first, then $\tau = \tau^R$.

Similarly, the later bits of Y can be extracted from R.

Poincaré Recurrence

- $T: \Omega \to \Omega$ is called measure-preserving if for every Borel set $A \subseteq \Omega$,
$$P(T^{-1}A) = P(A)$$

∴ If $(\Omega, \mathcal{F}, P)$ is a probability space and $T: \Omega \to \Omega$ is measure-preserving, then for almost every $x \in \Omega$, for every neighbourhood $N_\epsilon(x)$, $\exists\, k \in \mathbb{N}$ s.t.
$$P(T^{-k} N_\epsilon(x) \cap N_\epsilon(x)) > 0 \;\therefore$$

AP in Integer Sets

No matter how you partition integers into 2 sets, one of the partitions will have an arbitrary long arithmetic progression.

One of the two sets satisfy:
$\forall\, k \;\; \exists\, a_0, d$ such that
$$a_0,\ a_0 + d,\ \ldots,\ a_0 + (k-1)d \in \text{that set}$$

- δ density of a set A is defined as:
$$\limsup_{n \to \infty} \frac{|A \cap \{1, \ldots, n\}|}{n} > \delta$$

Note that primes are 0-dense.
Do primes have arb. long APs?

- Erdős: If $\sum_{n \in S} \frac{1}{n} \uparrow$ (diverges), then S has arb. long AP's?

- $\sum_{p:\text{prime}} 1/p \uparrow$ Primes have arb. long AP's too.

... a lot of work ...

$$P[T^{-a_0} N_\epsilon(x) \cap T^{-(a_0+d)} N_\epsilon(x) \cap \ldots \cap T^{-(a_0+(k-1)d)} N_\epsilon(x)] > 0$$

Ville's Theorem

THEOREM

A "selection rule" is a partial function

$$f: \Sigma^\infty \times \mathbb{N} \dashrightarrow \{\text{Yes, No}\}$$

Let E be a countably infinite collection of selection functions. Then, $\exists$ a sequence $\alpha \in \Sigma^\infty$ such that the following hold:

1) $\lim_{n\to\infty} \frac{\sum_{i=0}^{n-1} \alpha_i}{n} = \frac{1}{2}$

(½ 0's & ½ 1's in total)

2) $\forall f \in E$, such that f says yes infinitely many times, $\lim_{n\to\infty} \frac{\sum_{i=0}^{n-1} Y_i}{n} = \frac{1}{2}$, where Y_i is the i^{th} bit selected by f in α.

(So that we have an infinite string Y_f for each f under consideration)

↳ i.e. $Y_f = \lim_{n\to\infty} f(\alpha[0 \ldots n-1])$

3) $\forall n \in \mathbb{N}$, $\frac{\sum_{i=0}^{n-1} \alpha_i}{n} \leq \frac{1}{2}$

(The limit is reached from the below:)

Ex: $f(x_0 \ldots x_{n-1}) = \begin{cases} \text{yes}, & \text{if } n \text{ is even} \\ \text{no}, & \text{otherwise} \end{cases}$

Output sequence will be:

$$\langle x[i] : f(x[0 \ldots i-1]) = \text{"yes"} \rangle_{i \in \mathbb{N}}$$

* Let $f \in E$ s.t. f always says "yes". Then, (2) subsumes (1).

Proof of Theorem (when E is finite): Consider $(\alpha \restriction n)$.

Define $C(n) = \{f \in E \mid f(\alpha \restriction n) = \text{yes}\}$

↳ There are $2^{|E|}$ possibilities of $C(n)$ for any n.

Let $\alpha[n] = 0$ if $C(n)$ has appeared an even number of times among $C(0), C(1), \ldots, C(n-1)$, and 1 otherwise.

→ $C(n)$ has appeared odd no. of times previously, i.e., $C(0), C(1), \ldots, C(n-1)$.

Now, we show (3)

↳ Position the indices of $\alpha[0 \ldots n-1]$ based on which of the $2^{|E|}$ sets appear at that index.

It can now be argued that each 1 has a corresponding 0 which is also unique to it.

For every $C \in 2^E$, there can be an imbalance of at most 1 (in favor of #0)
Also $|\{C \in 2^E\}| = 2^{|E|}$
⇒ Diff. in #0 & #1 in α is upper bounded by $2^{|E|}$, a constant.

$\Rightarrow$ # of 1's $\leq$ # of 0's

Specifically $0 \leq \#0 - \#1 \leq 2^{|E|}$

For (1),

This argument can be specialized for any f and its corresponding γ_f.
$|\{C_f \in 2^E \mid f \in C_f\}| = 2^{|E|-1}$

$$\#0 + \#1 = n, \quad -2^{|E|} \leq \#1 - \#0 \leq 0,$$

$$\Rightarrow \frac{n - 2^{|E|}}{2} \leq \#1 = \sum_0^n \alpha_i \leq \frac{n}{2}$$

$$\Rightarrow 1/2 - 2^{|E|}/2n \leq \sum_0^n \alpha_i / n \leq 1/2$$

$$\Rightarrow \lim_{n \to \infty} \frac{\sum \alpha_i}{n} = \frac{1}{2}$$

Lempel-Ziv Compression and Complexity

LZ76 – Lempel Ziv Complexity
LZ77 – LZ77 Compressor
ZL78 – LZ78 Compressor
Shields '99

Algorithm [ZL78]

▷ Parsing:

Parse from left to right, looking for unique phrases. Every phrase occurs at most once.

Ex: λ 0 00 000 0000 ...

↳ k^{th} phrase's length ~ k

λ 0 1 00 01 10 11 000 ...

↳ k^{th} phrase's length ~ $\lfloor \log k \rfloor$

▷ Representation of phrases:

Suppose the current phrase (i^{th}) is $a[j \dots j+k-1]$. Then, $a[j \dots j+k-2]$ has occurred as a previous phrase in $a[0 \dots j-1]$.

Denote its index in phrasing as $\pi(i)$

($a[j \dots j+k-2]$'s)

Representation of phrase (i)

$= \pi(i) \cdot a[j+k-1]$

(represented in binary using exactly $\lceil \log i \rceil$ bits)

Ex: ZL78(0 1 00 01 10 11 000)

= 10 11 100 101 110 111 1000

13 input vs 20 output

ZL78(0 00 000 0000 00000)

= 10 10 110 1000 1010

- Omit λ during compression.
- Representation of k^{th} phrase uses $\lceil \log k \rceil + 1$ bits.

▷ Elegance: Online (incremental), deterministic, nearly linear time.

▷ Optimality: Over the class of information lossless finite state compressors, ZL78 compresses optimally.

↳ Output is asymptotically shorter than the output of any finite state compressor.

Complexity Measures

$c(x)$ (ZL78) := largest number of distinct phrases whose concatenation is x.

$c_{LZ}(x)$ (LZ76) := smallest number of distinct phrases in the substring parsing revision.

* Lower bound in terms of a deterministic notion of complexity ~ finite-state compressibility!

$$\frac{K(x)}{|x|} \leq \boxed{\text{Some-thing Comp-utable}} \leq \frac{\text{o/p length}}{\text{i/p length}} \leq |o/p|/|i/p| \text{ of the best ILFSC for } x$$

$$\frac{c(x)\ \log\ c(x)}{n\ \log |\Sigma|} + \text{error}$$

Also,

$$c_{LZ}(x) - 1 \leq c(x) \leq \frac{n \log |\Sigma|}{(1-\epsilon_n) \log n}$$

· "Surprisal" of x : $-\log p(x)$

- LZ76 parsing:
  phrase: < Overlap start, overlap end, next character >

  T: 0 01 01011

  There is 01 in the past string, but since overlap is allowed, the substring 0101 also exists

  001 01011 → substring
  parsed → overlap

  0 01 0101 1
  ↕ ↕ ↕ ↕
  0 1 0 1 1

  Essentially, overlap with the current phrase is allowed.
- Number of such phrases in this parsing is $C_{LZ}(s)$.

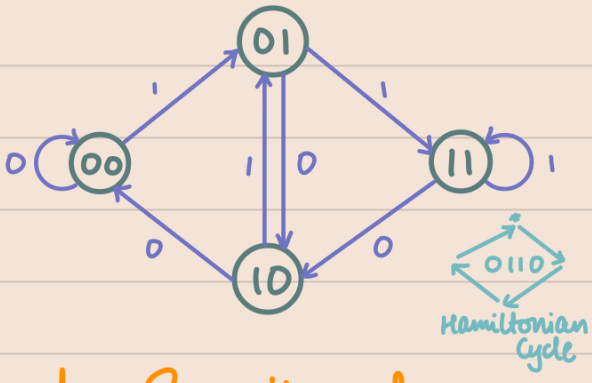
Q. How large is $C_{LZ}(x)$?
(Essentially, $C_{LZ}(x) \leq |x| / \log |x|$)

- Suppose an n-length string parses into N phrases.
  N is large when each phrase is small.
  Ex: λ, 0, 1, 00, 01, 10, 11, ...
  2^i phrases of length i
  $\Rightarrow n = \sum_{i=1}^{\square} i \cdot 2^i + \square$
  $\Rightarrow \square = N \quad \Rightarrow N < \frac{n}{\log n} + \dots$

de-Bruijn Sequence

- $B(k, n)$: A k^n length string in which each n-length string of Σ ($|\Sigma| = k$, a k-length alphabet) appears once, modulo cyclic shifts.
- $B(n) = B(2, n)$

- Ex: 0110 ∈ B(2)
- It was shown that de-Bruijn sequences have high C_{LZ} complexity
- Eulerian cycles in (k+1)-dimension automata correspond to Hamiltonian cycles in the k-dimension automata.



Hamiltonian cycle ≡ de-Bruijn Sequence