CS687 Endsem exam

Rohan (241110057)

1. Show that if there is a constant c such that for the sequence $X \in \sum^{\infty}$, we have: $C(X[0 \dots (n-1)] \mid n) < c$, then the sequence X is computable.

Solution:

Definitions and Background

Kolmogorov Complexity $C(x \mid y)$**:** This denotes the length of the shortest program (with respect to a fixed universal Turing machine U) that produces x when given y as input.

Computable Sequence: A binary sequence $X \in \{0,1\}^{\infty}$ is computable if there exists a total computable function $f : \mathbb{N} \to \{0,1\}$ such that $f(n) = X_n$, the n-th bit of X.

Overview of the Strategy

We are told that for every $n \in \mathbb{N}$, there exists a program of length less than c that, when given n, outputs the prefix $X[0 \dots n-1]$. There are only finitely many programs of length less than c. Hence, by the pigeonhole principle, some program must produce correct outputs for infinitely many n. We will use this to construct a total computable function that outputs each bit of X, thereby showing X is computable.

Formal Proof

Let P_c be the finite set of all binary programs of length less than c. Since there are at most $2^c - 1$ such programs, $|P_c| < \infty$.

By assumption, for every $n \in \mathbb{N}$, there exists some $p_n \in P_c$ such that:

$$U(p_n, n) = X[0 \dots n-1]$$

Because P_c is finite, by the pigeonhole principle, there exists a single program $p \in P_c$ that computes $X[0 \dots n-1]$ for infinitely many n. That is:

$$\exists p \in P_c \quad \text{such that } U(p, n) = X[0 \dots n-1] \text{ for infinitely many } n$$

We now use this fact to compute each bit X_m of the sequence.

Computing the Sequence X

To define a total computable function f such that $f(m) = X_m$, proceed as follows:

- On input m, simulate the universal machine U on input (p, n) for all $n > m$, dovetailing over increasing n.
- As soon as $U(p, n)$ halts and outputs a string s_n of length n, extract the m-th bit $s_n[m]$ and define $f(m) := s_n[m]$.
- Since p works for infinitely many n, this process will always halt for any m, and produce the correct bit X_m.

Thus, the function f is total and computable, and outputs each bit of X. Therefore, X is a computable sequence.

Conclusion

We have shown that if $C(X[0 \ldots n-1] \mid n) < c$ for all n, then there exists a fixed program of length less than c that outputs $X[0 \ldots n-1]$ for infinitely many n. By dovetailing over outputs of this program, we can recover each bit of X, thereby constructing a total computable function that computes X.

$$\boxed{\text{Hence, such a sequence } X \text{ is computable.}}$$

2. Show that if for any infinite binary sequence X,

$$\sum_{n \in N} 2^{n - K(X[0...(n-1)])} < \infty,$$

then

$$\lim_{n \to \infty} K(X[0 \ldots (n-1)]) = \infty.$$

Solution:

Definitions and Context

Prefix Kolmogorov Complexity $K(x)$: The prefix-free Kolmogorov complexity of a string x, denoted $K(x)$, is the length of the shortest self-delimiting program (for a fixed universal prefix machine) that outputs x.

Goal: Show that if the above weighted sum involving prefix complexities converges, then the prefix complexities of initial segments of X must diverge to infinity.

Strategy: Proof by Contrapositive

We will prove the contrapositive:

$$\text{If } \lim_{n \to \infty} K(X[0 \ldots n-1]) \neq \infty, \text{ then the sum diverges.}$$

That is, suppose there exists a constant c and an infinite set $S \subseteq \mathbb{N}$ such that:

$$\forall n \in S, \quad K(X[0 \ldots n-1]) \leq c.$$

We will show this implies that:

$$\sum_{n=1}^{\infty} 2^{n - K(X[0...n-1])} = \infty,$$

which contradicts the assumption of convergence.

Main Argument

Let $S = \{n_1, n_2, \ldots\} \subseteq \mathbb{N}$ be an infinite set such that:

$$\forall n_i \in S, \quad K(X[0 \ldots n_i - 1]) \leq c.$$

Then for each such n_i,

$$2^{n_i - K(X[0\ldots n_i-1])} \geq 2^{n_i - c}.$$

Therefore:

$$\sum_{n=1}^{\infty} 2^{n-K(X[0\ldots n-1])} \geq \sum_{n_i \in S} 2^{n_i - c} = 2^{-c} \sum_{n_i \in S} 2^{n_i}.$$

Because S is infinite and unbounded, the terms 2^{n_i} grow exponentially. Hence:

$$\sum_{n_i \in S} 2^{n_i} = \infty \Rightarrow \sum_{n=1}^{\infty} 2^{n-K(X[0\ldots n-1])} = \infty.$$

This contradicts the assumption that the sum is finite.

Conclusion

We have shown that if the complexity $K(X[0\ldots n-1])$ does not diverge (i.e., remains bounded for infinitely many n), then the sum diverges. Therefore, the original assumption of convergence implies:

$$\boxed{\lim_{n\to\infty} K(X[0\ldots n-1]) = \infty.}$$

This result reflects the intuition that compressibility of infinitely many long prefixes leads to heavy contributions in the sum, causing divergence. Thus, convergence requires increasing incompressibility of prefixes — i.e., high complexity.

3. Suppose $d : \Sigma^* \to [0, \infty)$ is a lower semicomputable martingale. Let X be a sequence on which d succeeds. Then show that X has infinitely many prefixes with low self-delimiting Kolmogorov complexity.

Solution:

Definitions and Background

Martingale: A function $d : \Sigma^* \to [0, \infty)$ is a martingale (with respect to the uniform measure) if it satisfies:

$$d(w) = \frac{1}{2}d(w0) + \frac{1}{2}d(w1), \quad \text{for all } w \in \Sigma^*.$$

Success: A martingale d is said to *succeed* on a sequence $X \in \{0,1\}^\infty$ if:

$$\limsup_{n\to\infty} d(X[0 \dots n-1]) = \infty.$$

Lower Semicomputability: A function d is lower semicomputable if there exists a total computable function $\phi(w, s) \in \mathbb{Q}$ such that:

$$\phi(w, 0) \leq \phi(w, 1) \leq \cdots \nearrow d(w).$$

Prefix-Free Kolmogorov Complexity $K(w)$: The length of the shortest self-delimiting program (on a fixed universal prefix-free machine) that outputs w.

Goal

We aim to show that if d is a lower semicomputable martingale that succeeds on X, then:

$$\text{There exist infinitely many } w \sqsubset X \text{ such that } K(w) \leq |w| - \log d(w) + O(1).$$

Main Argument

Since d succeeds on X, we can extract an infinite subsequence $\{n_i\}$ such that:

$$d(X[0 \dots n_i - 1]) \geq 2^i.$$

Define $w_i := X[0 \dots n_i - 1]$. We aim to show that each w_i has complexity significantly smaller than its length.

Step 1: Define a Semimeasure. We define a function $\mu : \Sigma^* \to [0, 1]$ as:

$$\mu(w) := d(w) \cdot 2^{-|w|}.$$

Since d is a martingale, μ is a semimeasure:

$$\sum_{w \in \Sigma^n} \mu(w) = \sum_{w \in \Sigma^n} d(w) \cdot 2^{-n} = 1.$$

Moreover, μ is lower semicomputable because d is.

Step 2: Apply the Coding Theorem. By the coding theorem for prefix-free complexity, for every string w, we have:

$$K(w) \leq -\log \mu(w) + O(1).$$

Substituting the definition of $\mu(w)$:

$$K(w) \leq -\log \left(d(w) \cdot 2^{-|w|}\right) + O(1) = |w| - \log d(w) + O(1).$$

Step 3: Apply to the Prefixes w_i. For each i, since $d(w_i) \geq 2^i$, we have:

$$K(w_i) \leq |w_i| - \log d(w_i) + O(1) \leq n_i - i + O(1).$$

This shows that w_i is significantly compressible compared to its length.

Conclusion

We have shown that if a lower semicomputable martingale d succeeds on a sequence X, then there exists an infinite sequence of prefixes $w_i \sqsubset X$ for which:

$$K(w_i) \leq |w_i| - \log d(w_i) + O(1),$$

i.e., the prefixes are highly compressible. This confirms that the sequence X cannot be prefix-random at all lengths, as martingale success implies the existence of infinitely many prefixes with low Kolmogorov complexity.

Hence, such a sequence X has infinitely many compressible prefixes w with $K(w) \leq \|w\| - \log d(w) + O(1)$

4. Show that the following inequality holds for all strings x, y and z of length n.

$$2C(x, y, z) \leq C(x, y) + C(y, z) + C(x, z) + O(\log n).$$

Solution:

Overview

We are given three strings $x, y, z \in \{0, 1\}^n$ and asked to bound twice the plain Kolmogorov complexity of the triple (x, y, z) in terms of the complexities of the three pairwise combinations.

Intuitively, this inequality reflects the subadditivity of information: knowing the three overlapping pairs provides enough information to reconstruct the entire triple, up to logarithmic overhead.

Tools from Kolmogorov Complexity

We use the following standard results:

1. **Chain Rule (Plain):**

$$C(a, b, c) \leq C(a, b) + C(c \mid a, b) + O(\log n)$$

$$C(a, b, c) \leq C(b, c) + C(a \mid b, c) + O(\log n)$$

2. **Combining Conditional Descriptions:** Given programs computing z from x, y and x from y, z, we can combine them into a program computing x, z from y, with an overhead of $O(\log n)$:

$$C(z \mid x, y) + C(x \mid y, z) \leq C(x, z) + O(\log n)$$

Main Argument

Apply the chain rule twice:

$$C(x, y, z) \leq C(x, y) + C(z \mid x, y) + O(\log n)$$

$$C(x, y, z) \leq C(y, z) + C(x \mid y, z) + O(\log n)$$

Adding both inequalities gives:

$$\begin{aligned} 2C(x, y, z) &\leq C(x, y) + C(z \mid x, y) + C(y, z) + C(x \mid y, z) + O(\log n) \\ &= C(x, y) + C(y, z) + [C(z \mid x, y) + C(x \mid y, z)] + O(\log n) \end{aligned}$$

Using the combination bound for the conditional terms:

$$C(z \mid x, y) + C(x \mid y, z) \leq C(x, z) + O(\log n)$$

Programs for $z \mid x, y$ and $x \mid y, z$ can be combined with a fixed interpreter to compute x, z, contributing $O(\log n)$ overhead. Specifically:

- The program for $z \mid x, y$ outputs z given x and y.
- The program for $x \mid y, z$ outputs x given y and z.
- By running both programs and verifying consistency with y, we reconstruct x, z, with $O(\log n)$ bits for synchronization.

Substituting into the previous inequality:

$$2C(x, y, z) \leq C(x, y) + C(y, z) + C(x, z) + O(\log n)$$

Conclusion

We have shown that for any three strings $x, y, z \in \{0, 1\}^n$, the following inequality holds:

$$\boxed{2C(x, y, z) \leq C(x, y) + C(y, z) + C(x, z) + O(\log n)}$$

This inequality captures a deep property of Kolmogorov complexity: the ability to compress a tuple using overlapping pairs, up to a small additive overhead. It formalizes the idea that there is significant shared information among the three pairs, allowing for efficient joint representation of the triple.