Kolmogorov Derandomization

Samuel Epstein
samepst@jptheorygroup.org

October 25, 2023

Abstract

Using Kolmogorov derandomization, we provide an upper bound on the compression size of numerous solutions. In general, if solutions to a combinatorial problem exist with high probability and the probability is simple, then there exists a simple solution to the problem. Otherwise the problem instance has high mutual information with the halting problem.

Contents

Chapter 1

Introduction

In mathematics, the probabilistic method is a constructive method of proving the existence of a certain type of mathematical object. This method, pioneered by Paul Erdös, involves choosing objects from a certain class randomly, and showing objects of a certain type occur with non-zero probability. Thus objects of a certain type are guaranteed to exist. For more information about the probabilistic method, we refer readers to [AS04]. Recent results have shown that there is a strong connection between probabilistic method and the compression sizes of encodings of mathematical objects, i.e. their Kolmogorov complexity, $\mathbf{K}$:

> *If the probabilistic method can be used to prove the existence of an object, then bounds on its Kolmogorov complexity can be proven as well.*

If there is a simple probability such that objects of a certain mathematical type occur with large probability, then there exists an object of that type that is simple, i.e. has low Kolmogorov complexity. More formally, if object x has P-probability of at least p of randomally occuring, then

$$\mathbf{K}(x) <^{\log} \mathbf{K}(P) - \log p + \epsilon.$$

The ϵ term is the amount of information that an encoding of the entire mathematical construct has with the halting sequence, which can obviously considered to be a negligible amount, except for exotic cases.

This inequality occurs through the application of the EL Theorem [Lev16, Eps19]. Producing bounds of the Kolmogorov complexity of an object through probabilistic means is called Kolmogorov derandomization.

I'd recommend derandomization as an area of research for masters students or researchers who are interested in moving into algorithmic information theory. This is because the majority of the technical effort resides in the domain to which derandomization is applied.

1.1 Three Eggs from the Chicken

Future work involves finding instances of the probabilistic method and applying derandomization to them. In particular, the Lovász Local Lemma, [EL], has been particularly compatible with derandomization. We present the first proved consequence of LLL and show how it is compatible with three versions of derandomization, one that involves Kolmogorov complexity, one that involves resource bounded Kolmogorov complexity, and one involving games.

A *hypergraph* is a pair $J = (V, E)$ of vertices V and edges $E \subseteq \mathcal{P}(V)$. Thus each edge can connect ≥ 2 vertices. A hypergraph is *k-regular* of the size $|e| = k$ for all edges $e \in E$. A 2-regular hypergraph is just a simple graph. A valid *C-coloring* of a hypergraph (V, E) is a mapping

$f : V \to \{1, \dots, C\}$ where every edge $e \in E$ is not *monochromatic* $|\{f(v) : v \in e\}| > 1$. The following classic result is proven using LLL.

Theorem. (Probabilistic Method) *Let $G = (V, E)$ be a k-regular hypergraph. If for each edge f, there are at most $2^{k-1}/e - 1$ edges $h \in E$ such that $h \cap f \neq \emptyset$, then there exists a valid 2-coloring of G.*

We can now use derandomization, to produce bounds on the Kolmogorov complexity of the simpliest such 2-coloring of G.

Theorem A. (Kolmogorov Derandomization) *Let $J = (V, E)$ be a k-regular hypergraph with $|E| = m$. If, for each edge f, there are at most $2^{k-1}/e - 1$ edges $h \in E$ such that $h \cap f \neq \emptyset$, then there exists a valid 2-coloring x of J with*

$$\mathbf{K}(x) <^{\log} \mathbf{K}(n) + 4me/2^k + \mathbf{I}(J; \mathcal{H}).$$

The term $\mathbf{I}(J; \mathcal{H}) = \mathbf{K}(J) - \mathbf{K}(J|\mathcal{H})$ is the amount of mutual information that J has with the halting sequence $\mathcal{H}$. We can now use resource derandomization to achieve bounds for the smallest time-bounded Kolmogorov complexity $\mathbf{K}^t(x) = \min\{p : U(p) = x \text{ in } t(\|x\|) \text{ steps}\}$ of a 2-coloring of J.

Assumption. ***Crypto*** *is the assumption that there exists a language in* $\mathbf{DTIME}(2^{O(n)})$ *that does not have size $2^{o(n)}$ circuits with Σ_2^p gates.*

Theorem B. (Resource Bounded Derandomization) *Assume **Crypto**. Let $J_n = (V, E)$ be a $k(n)$-regular hypergraph where $|V| = n$ and $|E| = m(n)$, uniformly polynomial time computable in n. Furthermore, for each edge f in J_n there are at most $2^{k(n)-1}/e - 1$ edges $h \in E$ such that $h \cap f \neq \emptyset$. Then there is a polynomial p, and a valid 2-coloring x of J_n with*

$$\mathbf{K}^p(x) < 4m(n)e/2^{k(n)} + O(\log n).$$

We define the following game involving hypergraphs. The player has access to a list of vertices and the goal of the player is to produce a valid 2-coloring of the hypergraph. We assume that for each edge f of the graph, there are at most $2^{k-1}/e - 1$ edges h such that $f \cap h \neq \emptyset$.

The game proceeds as follows. For the first round, environment gives the number of vertices to the player. The player has n vertices, each with starting color 1. At each subsequent turn, the environment sends to the player the edges which are monochromatic. The player can change the color of up to k vertices and sends these changes to the environment. The game ends when the player has a valid 2-coloring of the graph.

Theorem C. (Game Derandomization) *For $k \geq 6$, there exists a player $\mathbf{p}$ that can beat the environment $\mathbf{q}$ in $(1+\epsilon)n/k$ turns, with Kolmogorov complextiy $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q}; \mathcal{H}) - \log \epsilon$, where $\epsilon \in (0, 1)$.*

Chapter 2

Conventions

As noted in the introduction, $\mathbf{K}(x|y)$ is the conditional prefix free Kolmogorov complexity. $\mathbf{m}(x)$ is the algorithmic probability. The function $\mathbf{m}$ is universal, in that for any computable probability P over $\{0,1\}^*$, $O(1)\mathbf{m}(x) > 2^{-\mathbf{K}(P)}P(x)$. Thus for set $D \subseteq \{0,1\}^*$, computable probability P, $O(1)\mathbf{m}(D) > 2^{-\mathbf{K}(P)}P(D)$. $\mathbf{I}(x;\mathcal{H}) = \mathbf{K}(x) - \mathbf{K}(x|\mathcal{H})$ is the amount of information that the halting sequence $\mathcal{H} \in \{0,1\}^\infty$ has about x. For some function $t : \mathbb{N} \to \mathbb{N}$, the t-time bounded Kolmogorov complexity is $\mathbf{K}^t(x) = \min\{\|p\| : U(p) = x \text{ in time } t(\|x\|)\}$. A probability is *elementary*, if it has finite support and rational values. The deficiency of randomness of x relative to a elementary probability measure Q is $\mathbf{d}(x|Q) = -\log Q(x) - \mathbf{K}(x|Q)$. We recall for a set $D \subseteq \{0,1\}^*$, $\mathbf{m}(D) = \sum_{x\in D}\mathbf{m}(x)$. For the nonnegative real function f, we use $<^+ f$, $>^+ f$, and $=^+ f$ to denote $< f+O(1)$, $> f-O(1)$, and $= f \pm O(1)$. We also use $<^{\log} f$ and $>^{\log} f$ to denote $< f+O(\log(f+1))$ and $> f - O(\log(f+1))$, respectively. The following lemma is conservation of mutual information information with the halting sequence over deterministic processing.

Lemma 1 ([Eps22]) *For partial computable $f : \{0,1\}^* \to \{0,1\}^*$, $\mathbf{I}(f(a);\mathcal{H}) <^+ \mathbf{I}(a;\mathcal{H}) + \mathbf{K}(f)$.*

The following result is the EL Theorem [Lev16, Eps19]. It was originally formulated as a statement about learning. However since that time, there has been several unexpected applications. In this paper, the EL Theorem is used for derandomization.

Theorem 1 (EL Theorem [Lev16, Eps19]) *For finite $D \subset \{0,1\}^*$, $-\log\max_{x\in D}\mathbf{m}(x) <^{\log} -\log\mathbf{m}(D) + \mathbf{I}(D;\mathcal{H})$.*

Lemma 2 (Symmetric Lovász Local Lemma) *Let $E_1,\dots,E_n$ be a collection of events such that $\forall i : \Pr[E_i] \leq p$. Suppose further that each event is dependent on at most d other events, and that $ep(d+1) \leq 1$. Then, $\Pr\left[\bigcap_i \overline{E}_i\right] > \left(1 - \frac{1}{d+1}\right)^n$.*

Chapter 3

15 Instances of Kolmogorov Derandomization

3.1 Continuous Measures

A continuous semi-measure Q is a function $Q : \{0,1\}^* \to \mathbb{R}_{\geq 0}$, such that $Q(\emptyset) = 1$ and for all $x \in \{0,1\}^*$, $Q(x) \geq Q(x0) + Q(x1)$. For prefix free set D, $Q(D) = \sum_{x \in D} Q(x)$. Let $\mathbf{M}$ be a largest, up to a multiplicative factor, lower semi-computable continuous semi-measure. That is, for all lower computable continuous semi-measures Q there is a constant $c \in \mathbb{N}$ where for all $x \in \{0,1\}^*$, $c\mathbf{M}(x) > Q(x)$. Thus for any lower computable continuous semi-measure W and prefix-free set $S \subset \{0,1\}^*$, $-\log \mathbf{M}(S) <^+ \mathbf{K}(W) - \log W(S)$, where $\mathbf{K}(W)$ is the size of the smallest program that lower computes W. The monotone complexity of a finite prefix-free set G of finite strings is $\mathbf{Km}(G) \overset{\mathrm{def}}{=} \min\{\|p\| : U(p) \in x \sqsupseteq y \in G\}$. Note that this differs from the usual definition of $\mathbf{Km}$, in that our definition requires U to halt.

Theorem 2 **([Eps22])** *For finite prefix-free set $G \subset \{0,1\}^*$, we have $\mathbf{Km}(G) <^{\log} -\log \mathbf{M}(G) + \mathbf{I}(G;\mathcal{H})$.*

3.2 Examples

In this section 22 examples of derandomization are given. Some use the Lovász Local Lemma, which is particularly suited for derandomization. There are 4 instances of games.

3.2.1 k-Sat

For a set of n Boolean variables $x_1, \ldots, x_n$, a *CNF* formula ϕ is a conjunction $C_1 \cap \cdots \cap C_m$ of clauses. Each clause C_j is a disjunction of k literals, where each literal is a variable x_i or its negation $\overline{x_i}$. Clauses C_j and C_l are said to intersect if there is some x_i such that both clauses contain either

$$\left(x_1 \vee x_3 \vee \overline{x}_5\right) \wedge \left(\overline{x}_1 \vee x_2 \vee \overline{x_3}\right) \wedge \left(\overline{x_2} \vee \overline{x}_4 \vee \overline{x}_5\right)$$

Figure 3.1: An example 3-Sat instance. Each clause contains 3 literals consisting of variables x_i or their negations $\overline{x}_i$. An example satisfying assignment is $x_1 = \text{True}, x_2 = \text{True}, x_3 = \text{False}, x_4 = \text{False}, x_5 = \text{True}$.

x_i or $\overline{x_i}$. A satisfying assignment is a setting of each x_i to true or false that makes ϕ evaluate to true. An example of K-SAT can be seen in Figure 6.2.

Theorem 3 *Let ϕ be a* K-SAT *instance of n variables and m clauses, with $k \geq 3$. If each clause intersects at most $(2^k/e) - 1$ other clauses, then there exists a satisfying assignment ψ of ϕ of complexity $\mathbf{K}(\psi) <^{\log} \mathbf{K}(n) + 2em/2^k + \mathbf{I}(\phi;\mathcal{H})$.*

Proof. The sample space is the set of all 2^n assigments, and for each clause C_J, E_j is the bad event "C_j is not satisfied". Let $p = 2^{-k}$ and $d = (2^k/e) - 1$. Thus $\forall j$, $\Pr[E_j] \leq p$ as each clause has size k and each E_j is dependent on at most d other events by the intersection property. Thus since $ep(d+1)$, by the Lovász Local Lemma 6, we have that,

$$\Pr\left[\bigcap_j \overline{E_j}\right] > \left(1 - \frac{1}{d+1}\right)^m = \left(1 - \frac{e}{2^k}\right)^m. \tag{3.1}$$

Let $D \subset \{0,1\}^n$ be the set of all assignments that satisfy ϕ. $\mathbf{K}(D|\phi) = O(1)$. Let P be the uniform measure over sequences of size n. By Equation 7.2, assuming $k \geq 3$,

$$-\log P(D) < -m\log(1 - e/2^k) < 2em/2^k.$$

Thus by Theorem 1 and Lemma 6, for $k \geq 3$, there exists an assignment $\psi \in D$ that satisfies ϕ with complexity

$$\mathbf{K}(\psi) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H}) <^{\log} \mathbf{K}(n) + 2em/2^k + \mathbf{I}(\phi;\mathcal{H}).$$

□

3.2.2 HYPERGRAPH-COLORING

In this section we show how to compress colorings of k-uniform hypergraph. A *hypergraph* is a pair $J = (V,E)$ of vertices V and edges $E \subseteq \mathcal{P}(V)$. Thus each edge can connect ≥ 2 vertices. A hypergraph is *k-uniform* of the size $|e| = k$ for all edges $e \in E$. A 2-uniform hypergraph is just a simple graph. A valid *C-coloring* of a hypergraph (V,E) is a mapping $f : V \to \{1,\ldots,C\}$ where every edge $e \in E$ is not *monochromatic* $|\{f(v) : v \in e\}| > 1$. The goal of HYPERGRAPH-COLORING with parameter k, is given a k uniform hypergraph, produce a coloring using the smallest amount of colors. Theorem 4 uses the union bound whereas Theorem 5 uses the Lovász Local Lemma.

Theorem 4 *Every k-uniform hypergraph $J = (V,E)$, $|E| = n$, $|V| = m$ has a $\lceil \sqrt[k-1]{2m} \rceil$ coloring g where $\mathbf{K}(g) <^{\log} \mathbf{K}(k,n,m) + \mathbf{I}(J;\mathcal{H})$.*

Proof. We randomly color every vertex $v \in V$ using $C = \lceil \sqrt[k-1]{2m} \rceil$ colors. Let A_e be the bad event that edge e is monochromatic. This event has probability:

$$\Pr[A_e] = C \cdot (1/C)^k = (1/C)^{k-1} < 1/2m,$$

because there are C possible colors and each vertex has a $1/C$ chance of getting a particular color. We can get a union-bound over all m edges to find the bad probability.

$$\Pr\left[\bigcup_{e\in E} A_e\right] < \sum_{e\in E} \Pr[A_e] < m \cdot (1/2m) = 1/2. \tag{3.2}$$

We let $D \subset \{0,1\}^{n\lceil \log C\rceil}$ be the set of all encodings of C colorings (so no edge in monochromatic. $\mathbf{K}(D|J) = O(1)$. Let $P : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ be a probability measure over $\{0,1\}^*$, uniformly distributed over all $x \in \{0,1\}^{n\lceil \log C\rceil}$ that encode a C color assignment. $P(D) > .5$. By Theorem 1 and Lemma 1, there is a graph coloring $g \in D$ where

$$\mathbf{K}(g) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H}) <^{\log} \mathbf{K}(k,m,n) + \mathbf{I}(J;\mathcal{H}).$$

□

The second result on k-hypergraph coloring uses Lovász Local Lemma.

Theorem 5 *Let $J = (V,E)$, $|V| = n$, $|E| = m$ be a hypergraph and $k = \min_{f\in E}|f|$, with $k \geq 3$. Assume for each edge f, there are at most $2^{k-1}/e$ edges $h \in E$ such that $h \cap f \neq \emptyset$. Then for $k \geq 4$, there is a 2-coloring g of G such that $\mathbf{K}(g) <^{\log} \mathbf{K}(n)4me/2^k + \mathbf{I}(J;\mathcal{H})$.*

Proof. The case is degenerate for $k = 1$. Assume $k \geq 3$. We will use the Lovász Local Lemma to get a lower bound on the probability that a random assignment of colors is a 2 coloring. We assume each vertex is colored black or white with equal probability. For each edge $f \in E$, we define E_f to be the bad event "f is monochromatic". A valid 2-coloring exists iff $\Pr\left[\bigcap_f \overline{E}_f\right] > 0$.

Let $p = 1/2^{k-1}$ and $d = (2^{k-1}/e) - 1$. For each f, $\Pr[E_f] \leq p$ by the fact that f contains at least k vertices. Furthermore since f intersects at most d edges besides itself, E_f is dependent on at most d of the other events. Therefore since $ep(d+1) = 1$ we can apply the Lovász Local Lemma 6,

$$\Pr\left[\bigcap_f \overline{E}_f\right] > \left(1 - \frac{1}{1+d}\right)^m = \left(1 - \frac{e}{2^{k-1}}\right)^m. \tag{3.3}$$

Let $D = \{0,1\}^n$ be the set of all encoded 2 colorings of J. $\mathbf{K}(D|J) = O(1)$. Let $P(x) = [\|x\| = n]2^{-n}$ is the uniform distribution over sequences of length n. By Equation 3.3, assuming $k \geq 4$

$$-\log P(D) < -m\log(1 - 2e/2^k) < 4me/2^k.$$

By Theorem 1 and Lemma 6, there exist a 2-coloring g of J such that for $k \geq 4$,

$$\mathbf{K}(g) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H}) <^{\log} \mathbf{K}(n) + 4me/2^k + \mathbf{I}(J;\mathcal{H}).$$

□

3.2.3 Vertex-Disjoint-Cycles

Proposition 1 (Mutual Independence Principle) *Suppose that $Z_1, \dots Z_m$ is an underlying sequence of independent events and suppose that each event A_i is completely determined by some subset $S_i \subset \{Z_1, \dots, Z_m\}$. If $S_i \cap S_j = \emptyset$ for $j = j_1, \dots, j_k$ then A_i is mutually independent of $\{A_{j_1}, \dots, A_{j_k}\}$.*

This section deals with partitioning graphs into subgraphs such that each subgraph contains an independent cycle. An example partition can be seen in Figure 3.2.

Theorem 6 *There is a partition ℓ of vertices of a k-regular graph $G = (V,E)$, with vertices $|V| = n$, into $c = \lfloor \frac{k}{3\ln k}\rfloor$ components each containing a cycle that is vertex disjoint from the other cycles with complexity $\mathbf{K}(\ell) <^{\log} \mathbf{K}(n,k) + 2n/k^2 + \mathbf{I}(G;\mathcal{H})$.*

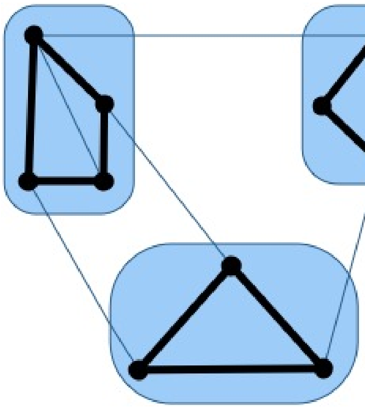


Figure 3.2: An example graph partitioned into 3 groups such that each group contains a cycle.

Proof. We partition the vertices of G into $c = \lfloor k/3\ln k \rfloor$ components by assigning each vertex to a component chosen independently and uniformly at random. With positive probability, we show that every component contains a cycle. It is sufficient to prove that every vertex has an edge leading to another vertex in the same component. This implies that starting at any vertex there exists a path of arbitrary length that does not leave the component of the vertex, so a sufficiently long path must include a cycle. A bad event $A_v = \{\text{vertex } v \text{ has no neighbor in the same component}\}$. Thus

$$\begin{aligned}\Pr[A_v] &= \prod_{(u,v)\in E} \Pr[u \text{ and } v \text{ are in different components}]\\ &= \left(1-\frac{1}{c}\right)^k < e^{-k/c} \le e^{-3\ln k} = k^{-3}.\end{aligned}$$

x A_v is determined by the component choices of itself and of its out neighbors $N^{\text{out}}(v)$ and these choices are independent. Thus by the Mutual Independence Principle, (Proposition 3) the dependency set of A_v consist of those u that share a neighor with v, i.e., those u for which $(\{v\} \cup N(v)) \cap (\{u\} \cup N(u)) \neq 0$. Thus the size of this dependency is at most $d = (k+1)^2$.

Take $d = (k+1)^2$ and $p = k^{-3}$, so $ep(d+1) = e(1+(k+1)^2)/k^3 \le 1$, holds for $k \ge 5$. One can trivially find a partition of a k-regular graph when $k < 5$ because $c = 1$. Thus, noting that $k \ge 5$,

$$\Pr\left[\bigcap_{v\in G} \overline{A}_v\right] > \left(1-\frac{1}{d+1}\right)^n = \left(1-\frac{1}{(k+1)^2+1}\right)^n > \left(1-\frac{1}{k^2}\right)^n. \tag{3.4}$$

$$-\log P(D) < -n\log(1-1/k^2) < 2n/k^2.$$

By Theorem 1 and Lemma 1, for large enough k, there exist a partitioning $\ell \in D$ of vertices into $c = \lfloor k/3\ln k \rfloor$ components each containing a cycle that is vertex disjoint from the other cycles with complexity

$$\mathbf{K}(\ell) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H}) <^{\log} \mathbf{K}(n,k) + 2n/k^2 + \mathbf{I}(G;\mathcal{H}).$$

□

3.2.4 Weakly-Frugal-Graph-Coloring

For an undirected graph $G = (V, E)$, a k-coloring assignment $f : V \to \{1, \dots, k\}$ is a $\beta-$*weakly frugal* if for all neighbors of vertices $v \in V$ contain at most β vertices with the same assignment. Note that a weakly frugal coloring assignment differs from a frugal coloring assignment, introduced in [HMR97], by the fact that the former can have two adjacent vertices with the same color.

Theorem 7 *For graph $G = (V, E)$, with $|V| = n$, with max degree $\Delta > 2e$ there is a β-weakly frugal coloring assignment f, with $\beta < \Delta$, using $Q \geq \Delta^{1+4/\beta}/2$ colors with complexity $\mathbf{K}(f) <^{\log} \mathbf{K}(n, Q) + 2n/\beta + \mathbf{I}((G, \beta, Q); \mathcal{H})$.*

Proof. This proof is a modification of the proof in [HMR97], except the restriction is relaxed to weakly-frugal coloring. Let us say each vertex is assigned one of Q colors with uniform randomness. For vertices $\{u_1, \dots, u_{\beta+1}\}$ that are in the neighborhood of a vertex $v \in V$, let $B_{u_1,\dots,u_{\beta+1}}$ be the bad event that the vertices are the same color. $\Pr[B_{u_1,\dots,u_{\beta+1}}] = p = 1/Q^\beta$. Each such bad event is dependent on at most $d = (\beta+1)\Delta\binom{\Delta}{\beta}$ other events. There are at most $m = n\binom{\Delta}{\beta+1}$ such events. The requirement that $ep(d+1) \leq 1$ of the Lovász Local Lemma is fulfilled, because

$$\begin{aligned}
&ep(d+1)\\
=&e\frac{1}{Q^\beta}\left(1 + (\beta+1)\Delta\binom{\Delta}{\beta}\right)\\
\leq&e\frac{1}{Q^\beta}\left(1 + (\beta+1)(\Delta^{\beta+2}/\beta!)\right)\\
\leq&e\frac{1}{Q^\beta}(1 + (\Delta^{\beta+3}/\beta!))\\
\leq&\frac{1}{Q^\beta}(\Delta^{\beta+4}/\beta!))\\
\leq&\frac{1}{Q^\beta}\Delta^{\beta+4}2^{-\beta}\\
\leq&1.
\end{aligned}$$

By Lovász Local Lemma 6,

$$\begin{aligned}
&-\log\Pr\left(\bigcap_{u_1,\dots,u_{\beta+1}}\overline{B}_{u_1,\dots,u_{\beta+1}}\right)\\
<&-m\log\left(1 - \frac{1}{d+1}\right)\\
<&2m\left(1 - \frac{1}{d+1}\right)\\
<&2n\binom{\Delta}{\beta+1}/\left(1 + (\beta+1)\Delta\binom{\Delta}{\beta}\right)\\
<&2n\binom{\Delta}{\beta+1}/\left((\beta+1)\Delta\binom{\Delta}{\beta}\right)\\
<&2n/(\beta+1).\\
<&2n/\beta.
\end{aligned}$$

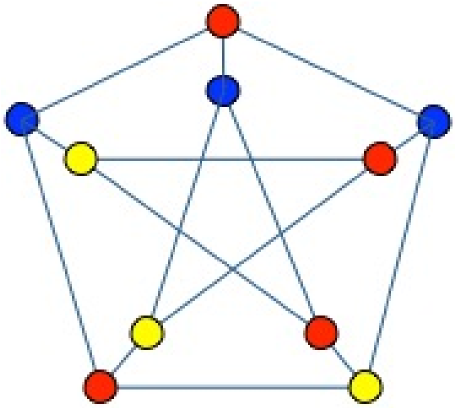


Figure 3.3: An example graph coloring. Nodes that share an edge are assigned different colors.

Let $D \subset \{0,1\}^{n\lceil \log Q\rceil}$ be encodings of all β-weakly frugal coloring of G using Q colors. $\mathbf{K}(D|G,Q,\beta) = O(1)$. Let P be uniform distribution over all Q-color assignments to n vertices. $-\log P(D) <^+ 2n/\beta$. By Theorem 1 and Lemma 6, we have a β-weakly frugal color assignment $f \in D$ of G using Q colors such that

$$\mathbf{K}(f) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H}) <^{\log} \mathbf{K}(n,Q) + 2n/\beta + \mathbf{I}((G,Q,\beta);\mathcal{H}).$$

□

3.2.5 Graph-Coloring

For graph $G = (V,E)$, with undirected edges, a k-coloring is a function $f : V \to \{1,\dots,k\}$ such that if $(v,u) \in E$, then $f(v) \neq f(u)$. An example graph coloring can be seen in Figure 3.3

Theorem 8 *For graph $G = (V,E)$, $|V| = n$ with max degree d, there is a k coloring f with $2d \leq k$, and $\mathbf{K}(f) <^{\log} \mathbf{K}(n,k) + 2nd/k + \mathbf{I}((G,k);\mathcal{H})$.*

Proof. Let us say we randomly assign a color to each vertex. The probability that the color of the ith vertex does not conflict with the previous coloring is at least $(k-d)/k$. Thus the probability of a proper coloring is $\geq ((k-d)/k)^n$. Let $D \subseteq \{0,1\}^{n\lceil \log k\rceil}$ be all encoded proper k colorings of G. $\mathbf{K}(D|G,k) = O(1)$. Let $P : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ be a probability measure that is the uniform distribution over all possible color assignments. Thus, assuming $d/k \leq .5$,

$$-\log P(D) \leq -n\log(1-d/k) \leq 2nd/k.$$

Thus by Theorem 1 and Lemma 1, there is a coloring $f \in D$ with

$$\begin{aligned}\mathbf{K}(f) &<^{\log} -\log \mathbf{m}(D) + \mathbf{I}(D;\mathcal{H})\\ &<^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H})\\ &<^{\log} \mathbf{K}(n,k) + 2nd/k + \mathbf{I}((G,k);\mathcal{H}).\end{aligned}$$

□

3.2.6 Max-Cut

Imagine a graph $G = (E, V)$, $|V| = n$, consisting of vertices V and undirected edges E, and a weight ω_e for each edge $e \in \mathbf{E}$. Let $\omega = \sum_{e \in E} \omega_e$ be the combined weight of all edges. The goal is to find a partition (A, B) of the vertices into two groups that maximizes the total weight of the edges between them.

Theorem 9 *There is a cut f of G that is $1/3$th optimal and $\mathbf{K}(f) <^{\log} \mathbf{K}(n) + \mathbf{I}(G; \mathcal{H})$.*

Proof. Imagine the algorithm that on receipt of a vertex, randomly places it into A or B with equal probability. Then the expected weight of the cut is

$$\mathbf{E}\left[\sum_{e \in E(A,B)} \omega_e\right] = \sum_{e \in E} \omega_e \Pr(e \in E(A, B)) = \frac{1}{2}\omega.$$

This means the expected weight of the cut is at least half the weight of the maximum cut. Some simple math results in the fact that $\Pr\left[\sum_{e \in E(A,B)} \omega_e\right] > \omega/3 \geq 1/4$. We can encode a cut into a binary string of x length n, where $x[i] = 1$, if the ith vertex is in A. Let P be the uniform distribution over strings of size n. Let $D \subset \{0, 1\}^n$ consist of all encoded cuts that are at least $1/3$ optimal. $\mathbf{K}(D|G) = O(1)$ and $P(D) \geq .25$. By Theorem 1 and Lemma 1,

$$\min_{f \in D} \mathbf{K}(f) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D; \mathcal{H}) <^{\log} \mathbf{K}(n) + \mathbf{I}(G; \mathcal{H}).$$

□

3.2.7 Max-3Sat

This problem consists of a boolean formula f in conjunctive normal form, comprised of m clauses, each consisting of a disjunction of 3 literals. Each literal is either a variable or the negation of a variable. We assume that no literal (including its negation) appears more than once in the same clause. There are n variables. The goal is to find an assignment of variables that satisfies as many clauses as possible.

Theorem 10 *There is an assignment x that is $6/7$th optimal and has complexity $\mathbf{K}(x) <^{\log} \mathbf{K}(m) + \mathbf{I}(f; \mathcal{H})$.*

Proof. The randomized approximation algorithm is as follows. The variables are assigned true or false with equal probability. Let Y_i be the random variable that clause i is satisfied. Thus the probability that clause Y_i is satisfied is $7/8$. So the total expected number of satisfied clauses is $7m/8$, which is $7/8$ of optimal. Some simple math shows the probability that number of satified clauses is $> 6m/7$ is at least $1/8$.

Let $x \in \{0, 1\}^n$ encode an assignment of n variables, where $x[i] = 1$ if variable i is true. Let $D \subset \{0, 1\}^n$ encode all assignments that are $6/7$th optimal. Let P be the uniform distribution over strings of length n. $\mathbf{K}(D|f) = O(1)$ and $P(D) \geq 1/8$. By Theorem 1 and Lemma 1,

$$\min_{x \in D} \mathbf{K}(x) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D; \mathcal{H}) <^{\log} \mathbf{K}(n) + \mathbf{I}(f; \mathcal{H}).$$

□

3.2.8 Balancing-Vectors

For a vector $v = (v_1, \dots, v_n) \in \mathbb{R}^n$, $\|v\|_\infty = \max_i |V_i|$. Binary matrix M is a matrix whose values are either 0 or 1. The goal of Binary Matrix, is given M, to find a vector $b \in \{-1, +1\}^n$ that minimizes $\|Mb\|_\infty$.

Theorem 11 *Given $n \times n$ binary matrix M, there is a vector $b = \{-1, +1\}^n$ such that $\|Mb\|_\infty \leq 4\sqrt{n \ln n}$ and $\mathbf{K}(b) <^{\log} \mathbf{K}(n) + \mathbf{I}(M; \mathcal{H})$.*

Proof. Let $v = (v_1, \dots, v_n)$ be a row of M. Choose a random $b = (b_1, \dots, b_n) \in \{-1, +1\}^n$. Let $i_1, \dots, i_m$ be the indices such that $v_{i_j} = 1$. Thus

$$Y = \langle v, b \rangle = \sum_{i=1}^{n} v_i b_i = \sum_{j=1}^{m} v_{i_j} b_{i_j} = \sum_{j=1}^{m} b_{i_j}.$$

$$\mathbf{E}[Y] = \mathbf{E}[\langle v, b \rangle] = \mathbf{E}\left[\sum_i v_i b_i\right] = \sum_i \mathbf{E}[v_i b_i] = \sum v_i \mathbf{E}[b_i] = 0.$$

By the Chernoff inequality and the symmetry Y, for $\tau = 4\sqrt{n \ln n}$,

$$\Pr[|Y| \geq \tau] = 2\Pr[v \cdot b \geq \tau] = 2\Pr\left[\sum_{j=1}^{m} b_{i_j} \geq \tau\right] \leq 2\exp\left(-\frac{\tau^2}{2m}\right) = 2\exp\left(-8\frac{n \ln n}{m}\right) \leq 2n^{-8}.$$

Thus, the probability that any entry in Mb exceeds $4\sqrt{n \ln n}$ is smaller than $2n^{-7}$. Thus, with probability $1 - 2n^{-7}$, all the entries of Mb have value smaller than $4\sqrt{n \ln n}$.

Let $P : \{0,1\}^* \to \mathbb{R}_{\geq 0}$, be the uniform measure over string of length n, with $P(x) = [\|x\| = n]2^{-n}$. Let D consist of all strings that encode vectors $b_x \in \{-1, +1\}^n$ in the natural way such that $\|Mb_x\|_\infty \leq 4\sqrt{n \ln n}$. $\mathbf{K}(D|M) = O(1)$. Thus by the above reasoning $P(D) \geq 1 - 2n^{-7} > 0.5$. By Theorem 1 and Lemma 1, there exists an $x \in D$, such that

$$\mathbf{K}(x) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D; \mathcal{H}) <^{\log} \mathbf{K}(n) + \mathbf{I}(M; \mathcal{H}).$$

Thus there exists a $b_x \in \{-1, +1\}^n$ that satisfies the theorem statement. □

We provide another derandomization example using balancing vectors.

Theorem 12 *Let $v = v_1, \dots, v_n \in \mathbb{R}^n$, all $|v_i| = 1$, Then there exist $\epsilon = \epsilon_1, \dots, \epsilon_n = \pm 1$ such that $|\epsilon_1 v_1 + \cdots + \epsilon_n v_n| \leq \sqrt{2n}$ and $\mathbf{K}(\{\epsilon\}) <^{\log} \mathbf{K}(n) + \mathbf{I}(v; \mathcal{H})$.*

Proof. Let $\epsilon_1, \dots, \epsilon_n$ be selected uniformly and independently from $\{-1, +1\}$. Set

$$X = |\epsilon v_1 + \cdots + \epsilon_n v_n|^2.$$

Then

$$X = \sum_{i=1} \sum_{j=1}^{n} \epsilon_i \epsilon_j v_i \cdot v_j.$$

So

$$\mathbf{E}[X] = \sum_{i=1}^{n} \sum_{j=1}^{n} v_i \cdot v_j \mathbf{E}[\epsilon_i \epsilon_j]$$

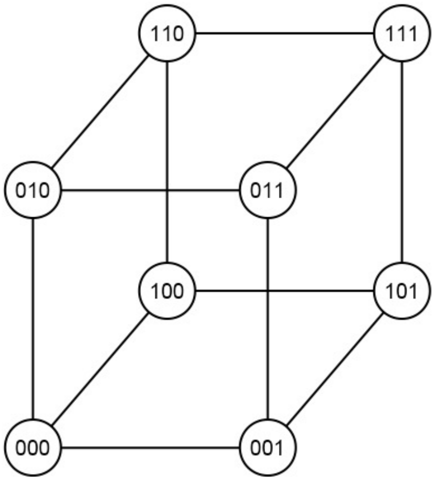


Figure 3.4: A Boolean Hypercube network, for $n = 3$.

When $i \neq j$, $\mathbf{E}[\epsilon_i \epsilon_j] = \mathbf{E}[\epsilon_i]\mathbf{E}[\epsilon_j] = 0$. When $i = j$, $\mathbf{E}[\epsilon_i^2] = 1$, so

$$\mathbf{E}[X] = \sum_{i=1}^{n} v_i \cdot v_i = n$$

So $\Pr[X \leq 2n] \geq 0.5$. Let $D \subseteq \{0,1\}^n$ consist of sequences of length n, each encoding an assignment of ϵ_1 to ϵ_n in the natural way, such that the assignment of ϵ results in an $X_\epsilon \leq 2n$. $\mathbf{K}(D|v) = O(1)$. Let P be the uniform measure over sequences of length n. By the above reasoning $P(D) \geq 0.5$. By Theorem 1 and Lemma 1, there is an assignment $\epsilon \in D$, such that $\mathbf{K}(\epsilon) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D; \mathcal{H}) <^{\log} \mathbf{K}(n) + \mathbf{I}(v; \mathcal{H})$. This assignment has $X_\epsilon \leq 2n$. Thus $|\epsilon_1 v_1 + \ldots \epsilon_n v_n| \leq \sqrt{2n}$, satisfying the theorem. □

3.2.9 Parallel-Routing

The Parallel-Routing problem consists of (G, d), a directed graph $G = (N, V)$ and a set of destinations $d : N \to N$. Each node represents a processor i in a network containing a packet v_i destined for another processor $d(i)$ in the network. The packet moves along a route represented by a path in G. During its transmission, a packet may have to wait at an intermediate node because the node is busy transmitting another packet. Each node contains a separate queue for each of its links and follows a FIFO queuing disciple to route packets, with ties handled arbitrarily. The goal of Parallel-Routing is to provide N routes from $i \in N$ to $d(i)$ that minimize lag time.

We restrict graphs to *Boolean Hypercube* networks, which is popular for parallel processing. The cube network contains $N = 2^n$ processing elements/nodes and is connected in the following manner. if $(i_0, \ldots, i_{n-1})$ and $(j_0, \ldots, j_{n-1})$ are binary representation of node i and node j, then there exist directed edges (i, j) and (j, i) between the nodes if and only if the binary representation differ in exactly one position. An example Boolean Hypercube can be found in Figure 3.4.

One set of solutions, called *oblivious algrithms* satisfies the following property: a route followed by v_i depends on $d(i)$ alone, and not on $d(j)$ for any $j \neq i$. We focus our attention on a 2 phase oblivious routing algorithm, Two-Phase. Under this scheme, packet v_i executes the following two phases independently of all the other packets.

1. Pick a intermediate destination $\sigma(i)$. Packet v_i travels to node $\sigma(i)$.

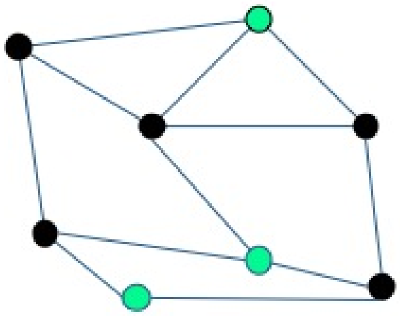


Figure 3.5: A graphical depiction of an independent set, represented by the green vertices. They do not share any edges.

2. Packet v_i travels from $\sigma(i)$ to destination $d(i)$.

The method that the routes use for each phase is the *bit-fixing* routing strategy. Its description is as follows. To go from i to $\sigma(i)$: one scans the bits of $\sigma(i)$ from left to right, and compares them with i. One sends v_i out of the current node along the edge corresponding to the left-most bit in which the current position and $\sigma(i)$ differ. Thus going from (1011) to (0000), the packet would pass through (0011) and then (0001).

Theorem 13 *Given a* PARALLEL-ROUTING *instance* (G,d)*, there is a set of intermediate destinations* $\sigma : \mathbb{N} \to \{0,1\}^n$ *for each* i *such that every packet* i *using* $\sigma(i)$ *and the* TWO-PHASE *algorithm reaches its destination in at most* $14n$ *steps and* $\mathbf{K}(\sigma) <^{\log} \mathbf{I}(G,d;\mathcal{H})$.

Proof. By Theorem 47 in [MR95], if the intermediate destinations are chosen randomly, with probability least $1-(1/N)$, every packet reaches its destination in $14n$ or fewer steps. Let $D \subset \{0,1\}^{nN}$ be the set of all intermediate destinations $\sigma \in D$ such that the lag time of instance (G,d) using σ is $\leq 14n$. Let $\mu : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ be the uniform continuous semi-measure, with $\mu(\emptyset) = 1$, $\mu(x) = 2^{-\|x\|}$. Thus $\mu(D) \geq 0.5$. $\mathbf{K}(D|(G,d)) = O(1)$. Theorem 2 and Lemma 1 results in

$$\mathbf{Km}(D) <^{\log} -\log \mathbf{M}(D) + \mathbf{I}(D;\mathcal{H}) <^{\log} -\log \mu(D) + \mathbf{I}((G,d);\mathcal{H}) <^{\log} \mathbf{I}((G,d);\mathcal{H}).$$

Thus using $y \sqsupseteq x \in D$ that realizes $\mathbf{Km}(D)$, one can construct a function $\sigma : \mathbb{N} \to \{0,1\}^n$ which produces the desired intermediate destinations, and $\mathbf{K}(\sigma) <^{+} \mathbf{K}(y) <^{\log} \mathbf{I}((G,d);\mathcal{H})$. □

3.2.10 INDEPENDENT-SET

An independent set in a graph G is a set of vertices with no edges between then, as shown in Figure 3.5. The INDEPENDENT-SET problem consists of an undirected graph G and the goal is to find the largest independent set of that G.

Theorem 14 *For a graph* G *on* n *vertices with* m *edges, there exists an independent set* S *of size* $0.75\sqrt{n} - 2m/n$ *and complexity* $<^{\log} \mathbf{K}(n,m) + 4(\log n)(m/n) + \mathbf{I}(G;\mathcal{H})$.

Proof. We use a modification of the algorithm in the proof of Theorem 6.5 in [MU05]. The randomized algorithm A is as follows.

1. Delete each vertex (along with its incident edges) independently with probability $1-p$.
2. For each remaining edge, remove it and one of its adjacent vertices.

For X, the number of vertices that survive the first round $\mathbf{E}[X] = np$. Let Y be the number of edges that survive the first step, $\mathbf{E}[Y] = mp^2$. The second steps removes at most Y vertices. The output is an independent set of size at least $\mathbf{E}[X-Y] = np - mp^2$. Let $p = 1/\sqrt{n}$. Thus $\mathbf{E}[X] = \sqrt{n}$, $\mathbf{E}[Y] = m/n$, and $\mathbf{E}[X-Y] = \sqrt{n} - m/n$. By the Markov inequality, $\Pr[Y < 2m/n] > 1/2$. By the Hoeffding's inequality,

$$\Pr[X \leq 0.75\sqrt{n}] \leq e^{-2*(0.75)^2(np)^2/n} \leq e^{-2(.75^2)(n*n^{-.5})^2/n} \leq e^{-2*0.5} = e^{-1}.$$

For a sequence $x \in \{0,1\}^*$, $x_{[1]} = |\{i : x[i] = 1\}|$ and $x_{[0]} = \|x\| - x_1$. Let $P : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ be a computable probability, where for a string $x \in \{0,1\}^n$, $P(x) = (1/\sqrt{n})^{x_{[1]}}(1-1/\sqrt{n})^{x_{[0]}}$. Thus each x represents a selection of vertices selected according to the randomized algorithm A. Let $D \subseteq \{0,1\}^n$ be the set consists of all sequences x such that the X variable resultant from x is $|X_x| > 0.75\sqrt{n}$ and the Y variable resultant from algorithm A is $|Y_x| \leq 2m/n$. Thus $P(D) \geq (1-e^{-1}) + 1/2 - 1 > 1/10$. Furthermore D can be constructed from G, with $\mathbf{K}(D|G) = O(1)$. By Theorem 1 and Lemma 1, there exists an $x \in D$, with

$$\begin{aligned}
\mathbf{K}(x) &<^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H})\\
&<^{\log} \mathbf{K}(n) + \mathbf{I}(G;\mathcal{H}).
\end{aligned}$$

In order for x to represent an independent set, the second step of algorithm A needs to be applied. In this case there are $< 2m/n$ vertices that needs to be removed. Thus a modification x' that has these vertices deleted represents an independent set.

$$\begin{aligned}
\mathbf{K}(x') &<^{\log} \mathbf{K}(x,n,m) + (2\log n)(2m/n)\\
&<^{\log} \mathbf{K}(n,m) + (4\log n)(m/n) + \mathbf{I}(D;\mathcal{H})\\
&<^{\log} \mathbf{K}(n,m) + (4\log n)(m/n) + \mathbf{I}(G;\mathcal{H}).
\end{aligned}$$

This independent set has $X_x > 0.75\sqrt{n}$ and $Y_x < 2m/n$, it size is $\geq 0.75\sqrt{n} - 2m/n$. □

3.2.11 Dominating-Set

A *dominating-set* of an undirected graph $G = (E,V)$ on n vertices is a set $U \subseteq V$ such that every vertex $v \in V - U$ has at least one neighbor in U. An example of a dominating set can be seen in Figure 3.6.

Theorem 15 *Every graph $G = (V,E)$, $|V| = n$ with min degree $\delta > 1$ has a dominating set U of size $\leq 3n\frac{1+\ln(\delta+1)}{\delta+1}$ and complexity $\mathbf{K}(U) <^{\log} \mathbf{K}(n,\delta) + 6(n\log n)/(\delta+1) + \mathbf{I}(G;\mathcal{H})$.*

Proof. Let $p \in [0,1]$. Let the vertices of V be picked randomly and independently, each with probability p. Let X be the random set of all vertices picked. $\mathbf{E}[|X|] = np$. Let $Y = Y_X$ be the random set of all vertices $V - X$ that do not have a neighbor in X. $\Pr(v \in Y_X) \leq (1-p)^{\delta+1}$. Thus $\mathbf{E}[|Y_X| \leq n(1-p)^{\delta+1} \leq ne^{-p(\delta+1)}$. We set $p = \ln(\delta+1)/(\delta+1)$. $\Pr[X \leq 3n\ln(\delta+1)/(\delta+1)] \geq 2/3$. $\Pr[Y_X \leq 3n/(\delta+1)] \geq 2/3$. Thus the probability of the previous two events is $\geq 1/3$.

Let $D \subseteq \{0,1\}^n$ be the set consisting of all sequences $x \in \{0,1\}^n$ where $x[i] = 1$ indicates vertex i was selected, such that the X variable resultant from x is $|X_x| \leq 3n\ln(\delta+1)/(\delta+1)]$

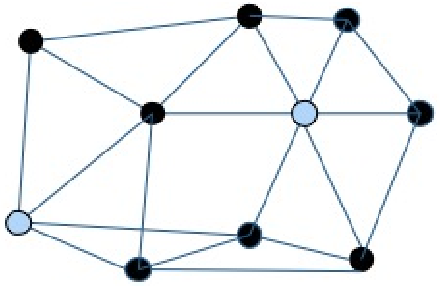


Figure 3.6: A graphical depiction of a dominating set. The two highlighted vertices are adjacent to all other vertices in the graph.

and the Y_x resultant variable is $|Y_x| \leq 3n/(\delta+1)$. Furthermore D can be constructed from G, with $\mathbf{K}(D|G) = O(1)$. Let $P : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ be a probability measure over $x \in \{0,1\}^n$, where $P(x) = \prod_{i=1}^n (px[i] + (1-p)(1-x[i]))$. By definition of D, $P(D) \geq 1/3$. Furthermore by Theorem 1 and Lemma 1, there is a subset of vertices $x \in D$, $x \subseteq V$, with

$$\mathbf{K}(x) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H}) <^{\log} \mathbf{K}(n,\delta) + \mathbf{I}(G;\mathcal{H}).$$

The sequence x represent the first step, however the set Y_x needs to be added to make x a dominating steps. Thus $3n/(\delta+1)$ vertices needs to be added, each can be encoded by $(2\log n)$ bits. Thus a dominating set x' of G exists of size $\leq 3n\frac{1+\ln(\delta+1)}{\delta+1}$ such that

$$\mathbf{K}(x') <^{\log} \mathbf{K}(n,\delta) + 6(n\log n)/(\delta+1) + \mathbf{I}(G;\mathcal{H}).$$

□

3.2.12 Set-Membership

For a set $G \subseteq \{0,1\}^\ell$, a function $f : \{0,1\}^* \to \{0,1\}$ is a partial checker for G, if $f(x) = 1$ if $x \in G$. We use $\mathcal{U}$ to denote the uniform distribution over $\{0,1\}^\ell$. $\text{Error}(G,f) = \Pr_{x\sim\mathcal{U}}[f(x) = 1, x \notin G]$. The goal of Set-Membership, is given a set $G \subseteq \{0,1\}^\ell$, what is the simplest partial checker f for G that reduces $\text{Error}(G,f)$.

Theorem 16 *For large enough n, given $G \subseteq \{0,1\}^\ell$, $|G| = m$, there is a partial checker f such that* $\text{Error}(f,G) \leq 0.878^{n/m}$ *and* $\mathbf{K}(f) <^{\log} \mathbf{K}(n,k,\ell) + n + \mathbf{I}((G,n,k);\mathcal{H})$.

Proof. We derandomize the Bloom filter algorithm [Blo70]. Let there be k random functions $h_i : \{0,1\}^\ell \to \{1,\dots,n\}$, where each h_i maps each input $x \in \{0,1\}^\ell$ to its range with uniform probability. We start with a string $v = 0^n$. For each member $x \in G$, and $i \in \{1,\dots,k\}$, $v[h_i(x)]$ is set to 1. Thus the functions h_i serve as a way to test membership of G. An example of the Bloom filter can be seen in Figure 3.7. If $x \in G$, then all the indicator functions h_i would be one. The probability that a specific bit is 0 is

$$p' = \left(1 - \frac{1}{n}\right)^{km}.$$

Let X be the number of bins that are 0. Due to [MU05],

$$\Pr(|X - np'| \geq \epsilon n) \leq 2e\sqrt{n}e^{-n\epsilon^2/3p'}.$$

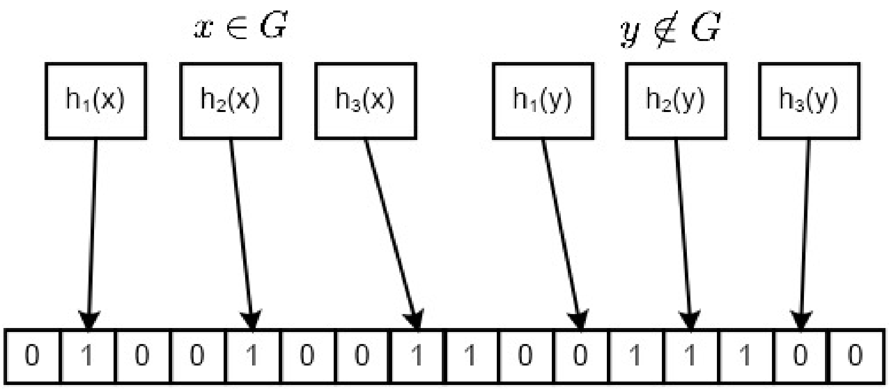


Figure 3.7: A graphical depiction of Bloom filter with $k = 3$ hash functions. The first element x is in G and is thus mapped to ones in the Bloom filter. Thus the Bloom filter would indicate that $x \in G$. The second element y is not in G and has some of the hash functions map to 0. Thus the Bloom filter would indicate that $y \notin G$.

For $\epsilon = p'/10$, we get

$$\Pr(X/n \geq p'9/10) \leq 2e\sqrt{n}e^{-np'/300}. \tag{3.5}$$

Thus for proper choice of k determined later, for large enough n, the right hand side of the above inequality is less than 0.5. Thus with probability $> .5$, the expected false positive rate, r, that is $x \in \{0,1\}^\ell$, $x \notin G$, $h_i(x) = 1$, for all $i \in \{1,\dots,k\}$ is less than

$$\begin{aligned} r \leq & (1 - .9p')^k \\ = & \left(1 - .9\left(1 - \frac{1}{n}\right)^{km}\right)^k \\ \leq & \left(1 - .9e^{-km/n}\right)^k. \end{aligned}$$

Setting $k = \lceil n/m \rceil$, with probability $\geq 1/2$, $r \leq (1 - .5e^{-2})^{m/n} \leq 0.878^{m/n}$. Furthermore, for large enough n, $p' > .5e^{-\lceil n/m \rceil (m/n)} \geq .5e^{-2}$, which can be plugged back into Equation 3.5.

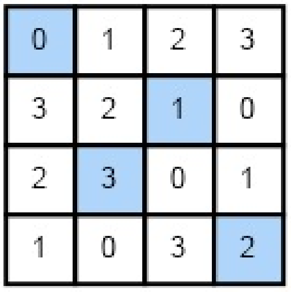
Let $F' \subset \{0,1\}^*$ consist of all encodings of k hash functions $h_i : \{0,1\}^\ell \to \{1,\dots,n\}$. Let $F \subseteq F'$ consist of all hash functions such that the false positive rate r is $\leq 0.878^{m/n}$. Let P be the uniform distribution over F'. By the above reasoning, for large enough n, $P(F) > 1/2$. $\mathbf{K}(F|G,k,n) = O(1)$. By Theorem 1 and Lemma 1, there is an $h \in F$ such that

$$\mathbf{K}(h) <^{\log} \mathbf{K}(P) - \log P(F) + \mathbf{I}(F;\mathcal{H}) <^{\log} \mathbf{K}(n,k,\ell) + \mathbf{I}((G,n,k);\mathcal{H}).$$

Thus h represents a set of k deterministic hash functions. Let x be the Bloom filter using h on G. Using x and h, one can define a partial checker f that is a Bloom filter such that $\mathrm{Error}(f,G) \leq 0.878^{n/m}$. Furthermore,

$$\mathbf{K}(f) <^{\log} \mathbf{K}(x,h) <^{\log} \mathbf{K}(n,k,\ell) + n + \mathbf{I}((G,n,k);\mathcal{H}).$$

□



0	1	2	3
3	2	**1**	0
2	**3**	0	1
1	0	3	**2**

Figure 3.8: A graphical depiction of a Latin Transversal. Each number appears exactly 4 times in the matrix. The transversal is a permutation of the matrix such that all its entries have different values.

3.2.13 Latin-Transversal

Let $A = (a_i)$ be an $n \times n$ matrix with integer entries. A permutation π is called a *Latin Transversal* if the entries $a_{i\pi(i)} (1 \leq i \leq n)$ are all distinct. An example Latin Transversal, where each integer occurs exactly 4 times, can be seen in Figure 3.8

Lemma 3 (Lopsided Lovász Local Lemma[ES91]) *Let $E_1, \ldots, E_n$ be a collection of events with dependency graph $G = (V, E)$. Suppose $\Pr\left(E_i \middle| \bigcap_{j \in S} \overline{E_j}\right) \leq \Pr(E_i)$, for all $i, S \subset V$ with no $j \in S$ adjacent to i. Suppose all events have probability at most p, G has degree at most d, and $4dp \leq 1$. Then $\Pr(\bigcap_i \overline{E_i}) \geq (1-2p)^n$.*

Theorem 17 *Suppose $k \leq (n-1)/16$ and suppose integers appears in exactly k entries of $n \times n$ matrix A. Then for $n \geq 3$, A has a Latin Traversal τ of complexity $\mathbf{K}(\tau) <^{\log} \mathbf{K}(n) + 4(k-1) + \mathbf{I}(A; \mathcal{H})$.*

Proof. Let π be a random permutation $\{1, 2, \ldots, n\}$, chosen according to a uniform distribution P among all possible $n!$ permutations. Define T by the set of all ordered fourtuples (i, j, i', j') with $i < i'$, $j \neq j'$, and $a_{ij} = a_{i'j'}$. For each $(i, j, i', j') \in T$, let $A_{iji'j'}$ denote the bad event that $\pi(i) = j$ and $\pi(i') = (j')$. Thus $A_{iji'j'}$ is the bad event that the random permutation has a conflict at (i, j) and (i', j').

Clearly $P(A_{iji'j'}) = 1/n(n-1)$. The existence of a Latin Transversal is equivalent to the statement that with positive probability, none of these events hold. We define a symmetric digraph G on the vertex set T by making (i, j, i', j') adjacent to (p, q, p', q') if $\{i, i'\} \cap \{p, p'\} \neq \emptyset$ or $\{j, j'\} \cap \{q, q'\} \neq \emptyset$. Thus these two fourtuples are not adjacent iff the four cells (i, j), (i', j'), (p, q) and (p', q') occupy four distinct rows and columns of A.

The maximum degree of G is less than $4nk \leq d$ because for a given $(i, j, i', j') \in T$ there are at most $4n$ choices of (s, t) with either $s \in \{i, i'\}$ or $t \in \{j, j'\}$ and for each of these choices of (s, t) there are less than k choices for $(s', t') \neq (s, t)$ with $a_{st} = a_{s't'}$. Each fourtuple (s, t, s', t') can be uniquely represented as (p, q, p', q') with $p < p'$. Since $4dp \leq 16nk/(n(n-1) \leq 1$, by the Lopsided Lovász Local Lemma, 3, the desired bounds can be achieved if we can show that

$$\Pr\left(A_{iji'j'} \middle| \bigcap_S A_{pqp'q'}\right) \leq 1/n(n-1),$$

for any $(i, j, i', j') \in T$ and any subset S of T which are not-adjacent in G to (i, j, i', j'). By symmetry we can assume $i = j = 1$, $i' = j' = 2$. A permutation π is *good* if it satisfies $\bigcap_S \overline{A}_{pqp'q'}$

and let S_{ij} denote the set of all good permutations π satisfying $\pi(1)=i$ and $\pi(2)=j$. $|S_{12}|\leq|S_{ij}|$ for all $i\leq j$.

Indeed suppose first that $i,j>2$. For each good $\pi\in S_{12}$ define a permutation π^* as follows. Suppose $\pi(x)=i$ and $\pi(y)=j$. Then define $\pi^*(1)=i$, $\pi^*(2)=j$, $\pi^*(x)=1$, $\pi^*(y)=2$ and $\pi^*(t)=\pi(t)$ for all $t\neq 1,2,x,y$. One can easily check that π^* is good, since the cells $(1,i),(2,j),(x,1),(y,2)$ are not part of any $(p,q,p',q')\in S$. Thus $\pi^*\in S_{ij}$ and since the mapping $\pi\to\pi^*$ is injective $|S_{12}|\leq|S_{ij}|$. One can define an injection mappings showing that $|S_{12}|\leq|S_{ij}|$ even when $\{i,j\}\cap\{1,2\}\neq\emptyset$. If follows that $\Pr\left(A_{1122}\cap\bigcap_S\overline{A}_{pqp'q'}\right)\leq\Pr\left(A_{1i2j}\cap\bigcap_S\overline{A}_{pqp'q'}\right)$ and hence $\Pr\left(A_{1122}\middle|\bigcap_S\overline{A}_{pqp'q'}\right)\leq 1/n(n-1)$.

The number of bad events $A_{iji'j'}$ is $\left(\frac{n^2}{k}\right)\binom{k}{2}$, as there are n^2/k distinct numbers, and each number appears k times. Thus by the Lopsided Lovász Local Lemma 3, for $n\geq 3$,

$$\begin{aligned}
\Pr\left(\bigcap_i\overline{A}_{iji'j'}\right)&\geq(1-2/n(n-1))^{\left(\frac{n^2}{k}\right)\binom{k}{2}}\\
-\log\Pr\left(\bigcap_i\overline{A}_{iji'j'}\right)&<\left(\frac{n^2}{k}\right)\binom{k}{2}\log(1-2/n(n-1))\\
&<2\left(\frac{2n^2}{kn(n-1)}\right)\binom{k}{2}\\
&<\left(\frac{8}{k}\right)\binom{k}{2}\\
&\leq 4(k-1).
\end{aligned}\tag{3.6}$$

Let $D\subset\{0,1\}^*$ be all encodings of permutations of A that are Latin Transversals. $\mathbf{K}(D|A)=O(1)$. We recall that P is the uniform distribution over all permutation of A. By the Equation 3.6, $-\log P(D)<4(k-1)$. Thus by Theorem 1 and Lemma 1, for $n\geq 3$, there exists a permutation $\tau\in D$ that is a Latin Transversal and has complexity

$$\mathbf{K}(\tau)<^{\log}\mathbf{K}(P)-\log P(D)+\mathbf{I}(D;\mathcal{H})<^{\log}\mathbf{K}(n)+4(k-1)+\mathbf{I}(A;\mathcal{H}).$$

□

3.2.14 Function-Minimization

Given computable functions $\{f_i\}_{i=1}^n$, where each $f_i:\mathbb{N}\to\mathbb{N}\cup\infty$, the goal of Function-Minimization is to find numbers $\{x_i\}_{i=1}^n$, that minimizes $\sum_{i=1}^n f_i(x_i)$. Let $p:\mathbb{N}\to\mathbb{R}_{\geq 0}$ be a computable probability measure where $\mathbf{E}_p[f_i]\in\mathbb{R}$ for all $i=1,\dots,n$. We define a computable probability $P:\{0,1\}^*\to\mathbb{R}_{\geq 0}$ where $P(\langle a_1\rangle\langle a_2\rangle\dots\langle a_n\rangle)=\prod_{i=1}^n p(a_i)$. $\mathbf{K}(P)<^+\mathbf{K}(p,n)$. Let D' be a (potentially infinite) set of strings where $x\in D$ iff $x=\langle a_1\rangle\langle a_2\rangle\dots\langle a_n\rangle$ and

$$\sum_{i=1}^n f_i(a_i)\leq\left\lceil 2\sum_{\{b_i\}}\left(\prod_{i=1}^n p(b_i)\right)\sum_{i=1}^n f_i(b_i)\right\rceil=\left\lceil 2\sum_{i=1}^n\mathbf{E}_p[f_i]\right\rceil.$$

Let $\tau=\lceil 2\sum_{i=1}^n\mathbf{E}_p[f_i]\rceil$. By the Markov inequality, let the finite set $D\subseteq D'$ be constructed from $(p,\{f_i\},\tau)$, such that $P(D)>1/2$ and $\mathbf{K}(D|(p,\{f_i\},\tau))=O(1)$. By Theorem 1 and Lemma 1, there a string $x \in D$ such that

$$\begin{aligned}\mathbf{K}(x) &<^{\log} -\log \mathbf{m}(D) + \mathbf{I}(D;\mathcal{H})\\ &<^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}((p,\{f_i\},\tau);\mathcal{H})\\ &<^{\log} \mathbf{K}(p,n) + \mathbf{I}((p,\{f_i\},\tau);\mathcal{H}).\end{aligned}$$

Thus given any computable probability p and functions $\{f_i\}_{i=1}^n$, there are numbers $\{x_i\}_{i=1}^n$ such that $\sum_{i=1}^n f(x_i) \leq \lceil 2\sum_{i=1}^n \mathbf{E}_p[f_i]\rceil = \tau$ and $\mathbf{K}(\{x_i\}_{i=1}^n) <^{\log} \mathbf{K}(n,P) + \mathbf{I}((p,\{f_i\},\tau);\mathcal{H})$. Note that there is a version of these results when the functions are uncomputable, but this is out of the scope of the paper.

An instance of this formulation is as follows. Let $n = 1$ and $f_1(a) = [a > 2^m]\infty + [a \leq 2^m]2^{m-\mathbf{K}(a|m)}$. Let $p(a) = [a \leq 2^m]2^{-m}$. Thus this example proves there exists a number x such that $f_1(x) \leq \lceil 2\mathbf{E}_p[f_1]\rceil \leq 2$. Furthermore

$$\mathbf{K}(x) <^{\log} \mathbf{K}(p) + \mathbf{I}((p,f_1);\mathcal{H}) <^{\log} \mathbf{K}(m) + \mathbf{I}((m,f_1);\mathcal{H}).$$

But if $f_1(x) \leq 2$, by the definition of f_1, this means $\mathbf{K}(x) \geq m-1$. This means $m <^{\log} \mathbf{I}((m,f_1);\mathcal{H}) <^{\log} \mathbf{I}(f_1;\mathcal{H})$. This makes sense because f_1 is a deficiency of randomness function and therefore $m <^{\log} \mathbf{K}(f_1)$ and $\mathbf{K}(f_1|\mathcal{H}) <^{+} \mathbf{K}(m)$.

3.2.15 Super-Set

Given a finite set $S \subseteq \{0,1\}^n$, the goal of Super-Set is to find a set $T \supseteq S$, $T \subseteq \{0,1\}^n$ that minimizes $|T|$.

Theorem 18 *Given $m \leq n$, $S \subseteq \{0,1\}^n$, $|S| < 2^{n-m-1}$ there exists a $T \supseteq S$, $T \subseteq \{0,1\}^n$ $|T| = 2^{n-m}$, $\mathbf{K}(T) <^{\log} \mathbf{K}(n,m) + (m+1)|S| + \mathbf{I}((S,m);\mathcal{H})$.*

Proof. Let $P : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ be the the uniform distribution over all sequences of size 2^n that have exactly 2^{n-m} 1s. Let $D \subset \{0,1\}^{2^n}$ consist of all sequences $x_R \in \{0,1\}^{2^n}$ that encode sets $R \subseteq \{0,1\}^n$ in the natural way such that $R \supseteq S$ and $|R| = 2^{m-n}$. Thus if $x \in D$ then x has 2^{n-m} 1s. $P(D) =$

$$\left(\frac{2^{n-m}}{2^n}\right)\left(\frac{2^{n-m-1}}{2^n-1}\right)\cdots\left(\frac{2^{n-m}-|S|}{2^n-|S|}\right) \geq \left(\frac{2^{n-m}-|S|}{2^n-|S|}\right)^{|S|} \geq \left(\frac{2^{n-m-1}}{2^n}\right)^{|S|} = 2^{-(m+1)|S|}.$$

$\mathbf{K}(D|(S,m)) = O(1)$. Thus by Theorem 1 and Lemma 1, there exists a $t \in D$, such that $\mathbf{K}(t) <^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H}) <^{\log} \mathbf{K}(n,m) + (m+1)|S| + \mathbf{I}((S,m);\mathcal{H})$. This t encodes a set $T \supseteq S$, $T \subseteq \{0,1\}^n$ such that $|T| = 2^{n-m}$.

□

Chapter 4

Classical Channels

There are deep connections between classical information theory and algorithmic information theory, with many theorems of the former appearing in an algorithmic form in the latter. In this section we revisit this connection. In particular we prove properties about the compression size of shared codebooks using Kolmogorov derandomization. A standard setup in information theory is two parties Alice and Bob who want to communicate over a noisy channel and share a codebook over a noiseless channel. However one might ask is how many bits did it take to communicate the codebook? By using derandomization, the tradeoff between codebook complexity and communication capacity can be proven.

Definition 1 (Discrete Memoryless Channel) *The input and output alphabets $\mathcal{X}$ and $\mathcal{Y}$ are finite. The channel $(\mathcal{X}, p(y|x), \mathcal{Y})$ is represented by a conditional probability distribution $p(y|x)$. To send multiple symbols, we have $p(y^n|x^n) = \prod_{i=1}^n p(y_i|x_i)$. The capacity of channel with respect to a distribution Q over $\mathcal{X}$ is*

$$C_Q = I(X:Y) \textit{ where random variables } (X,Y) \textit{ are distributed according to } Q(x)p(y|x).$$

The term I is the mutual information between random variables.

Definition 2 (Codebook) *A (M,n) codebook for channel $(\mathcal{X}, p(y|x), \mathcal{Y})$ contains the following:*

1. *An encoder* $\mathrm{Enc}_n : \{1,\dots,M\} \to \mathcal{X}^n$.

2. *A decoder* $\mathrm{Dec}_n : \mathcal{Y}^n \to \{1,\dots,M\}$.

The rate of the codebook is $R = \frac{\log M}{n}$. The conditional probability of error is $\lambda_i = \sum_{y_n} p(y^n|x^n = \mathrm{Enc}(i))[\mathrm{Dec}(y^n) \neq i]$, where $[\cdot]$ is the indicator function. The average error rate of the codebook with respect to a fixed channel p is $P_e^{(n)} = \frac{1}{M}\sum_{i=1}^M \lambda_i$. It is the probability that, given the uniform distribution over $\{1,\dots,M\}$ for the sending symbols, the receiver decodes a symbol different from the encoded one.

This section shows the following high level description of a communication scheme is possible: there is a sender Alice and a receiver Bob that communicate through a noisy memoryless discrete channel and Alice can send a codebook to Bob once on a side noiseless channel. Bob has oracle acess to the channel function $p(y|x)$ but Alice does not. Given a computable distribution Q over the input alphabet, and assuming the channel is non-exotic, Alice can hypothetically send $\sim\mathbf{K}(Q)$ bits plus some encoded parameters describing a codebook to Bob on the side channel. Then Alice and Bob can communicate with any rate R less than the capacity C_Q over the noisy channel. This setup is formalized with Theorem 20. To prove this theorem, some results are needed from classical information theory.

4.0.1 Jointly Typical Sequences

We need the following definition and theorem, which can be found in [CT91], in the proof of Theorem 20. $H(X)$ is the entropy of random variable X, and $I(X:Y)$ is the mutual information between random variables X and Y.

Definition 3 *The set $A_\epsilon^{(n)}$ of jointly typical sequences $\{(x^n, y^n)\}$ with respect to the distribution $p(x,y)$ is the set of n-sequences with empirical entropies ϵ-close to the true entropies. $\mathcal{X}$ and $\mathcal{Y}$ are the finite discrete alphabet of random variables X and Y. Let $p(x^n, y^n) = \prod_{i=1}^n p(x_i, y_i)$.*

$$\begin{aligned} A_\epsilon^{(n)} = &\{(x^n, y^n) \in \mathcal{X}^n \times \mathcal{Y}^n : \\ &\left| -\frac{1}{n} \log p(x^n) - H(X) \right| < \epsilon, \\ &\left| -\frac{1}{n} \log p(y^n) - H(Y) \right| < \epsilon, \\ &\left| -\frac{1}{n} \log p(x^n, y^n) - H(X,Y) \right| < \epsilon \right\}. \end{aligned}$$

The following theorem details properties about the set $A_\epsilon^{(n)}$. A proof for it can be found in [CT91].

Theorem 19 (Joint AEP) *Let (X^n, Y^n) be sequences of length n drawn i.i.d. according to $p(x^n, y^n) = \prod_{i=1}^n p(x_i, y_i)$. Then*

1. $\Pr\left((X^n, Y^n) \in A_\epsilon^{(n)}\right) \to 1 - o(1)$.
2. *If $(\tilde{X}^n, \tilde{Y}^n) \sim p(x^n)p(y^n)$ ($\tilde{X}^n$ and $\tilde{Y}^n$ are independent with the same marginals as $p(x^n, y^n)$), then* $\Pr\left((\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}\right) \leq 2^{-nI(X:Y)-3\epsilon}$.

4.0.2 Naive Sender Paradigm

Theorem 20 *For channel $\mathfrak{C} = (\mathcal{X}, p(y|x), \mathcal{Y})$ and every computable distribution Q over $\mathcal{X}$, for every rate $R < C_Q$, there is a $(2^{nR}, n)$ codebook $(\mathrm{Enc}_n, \mathrm{Dec}_n)$ with rate R and average error rate $o(1)$ such that there is a program p with $\|p\| <^{\log} \mathbf{K}(n, R, Q) + \mathbf{I}((n, R, Q, \mathfrak{C}); \mathcal{H})$ and*

$$\begin{aligned} U(p, x) &= \mathrm{Enc}_n(x), \\ U(p, \mathfrak{C}, x) &= \mathrm{Dec}_n(x). \end{aligned}$$

Proof. We start by generating a $(2^{nR}, n)$ code randomly according to distribution Q. We generate 2^{nR} codewords $x \in \mathcal{X}$ independently according to the distribution

$$Q(x^n) = \prod_{i=1}^n p(x_i).$$

The codewords can be represented as rows of a matrix

$$\mathcal{C} = \begin{bmatrix} x_1(1) & x_2(1) & \dots & x_n(1) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(2^{nR}) & x_2(2^{nR}) & \dots & x_n(2^{nR}) \end{bmatrix}$$

Each entry is generated i.i.d according to $Q(x)$, with

$$\Pr(\mathcal{C}) = \prod_{w=1}^{2^{nR}} \prod_{i=1}^{n} p(x_i(w)).$$

Consider the following algorithm for encoding and decoding a message.

1. A random code $\mathcal{C}$ is generated according to $Q(x)$.
2. The code $\mathcal{C}$ is sent to both the sender and the receiver. *Only the receiver is assumed to know the channel transition matrix $p(y|x)$ for the channel.* This differs from the standard literature, which assumes knowledge of p by the sender.
3. A message W is chosen according to the uniform distribution.

   $$\Pr(W = w) = 2^{-nR}, \qquad w = 1, 2, \ldots, 2^{nR}.$$

4. The wth codeword $X^n(w)$ corresponding to the wth row of $\mathcal{C}$ is sent over the channel.
5. The receiver receives a sequence Y^n according to the distribution

   $$P(y^n|x^n(w)) = \prod_{i=1}^{n} p(y_i|x_i(w)).$$

6. The receiver decares that the index $\hat{W}$ was sent if the following conditions are satisfied:

   - $(X^n(\hat{W}), Y^n)$ is jointly typical, i.e. $(X^n(\hat{W}), Y^n) \in A_\epsilon^{(n)}$.
   - There is no other index $W' \neq \hat{W}$ such that $(X^n(W'), Y^n) \in A_\epsilon^{(n)}$.

   If no such $\hat{W}$ exists or if there are more than one, an error is declared, and the decoder outputs 0.

7. There is a decoding error if $\hat{W} \neq W$. Let $\mathcal{E}$ be this event.

We now analyze the probability of the error with respect to the random codebook $\mathcal{C}$.

$$\begin{aligned}
\Pr(\mathcal{E}) &= \sum_{\mathcal{C}} \Pr(\mathcal{C}) P_e^{(n)}(\mathcal{C}) \\
&= \sum_{\mathcal{C}} \Pr(\mathcal{C}) \frac{1}{2^{nR}} \sum_{w=1}^{2^{nR}} \lambda_w(\mathcal{C}) \\
&= \frac{1}{2^{nR}} \sum_{w=1}^{2^{nR}} \sum_{\mathcal{C}} \Pr(\mathcal{C}) \lambda_w(\mathcal{C}) \\
&= \sum_{\mathcal{C}} \Pr(\mathcal{C}) \lambda_1(\mathcal{C}) \qquad (4.1) \\
&= \Pr(\mathcal{E}|W = 1),
\end{aligned}$$

where Equation 4.1 is due to symmetry of the code construction. We define

$$E_i = \{(X^n(i), Y^n) \in A_\epsilon^{(n)}\}, \qquad i \in \{1, 2, \ldots, 2^{nR}\}.$$

So E_i is the event that the ith code and Y^n are jointly typical, noting that Y^n is the result of sending the first codeword $X^n(1)$ over the channel. So

$$\Pr(\mathcal{E}|W=1) = P(E_1^c \cup E_2 \cup E_3 \ldots E_{2^{nR}}|W=1) \leq P(E_1^c|W=1) + \sum_{i=2}^{2^{nR}} P(E_i|W=1).$$

Due to the code generation procedure, $X^n(1)$ and $X^n(i)$ are independent for $i \neq 1$, and therefore, so are Y^n and $X^n(i)$. Due to Theorem 19 (2), the probability that $X^n(i)$ and Y^n are jointly typical is $\leq 2^{-n(I(X;Y)-3\epsilon)}$, where random variables X and Y are distributed acording to $Q(x)p(y|x)$. So by Theorem 19 (1), for sufficiently large n,

$$\begin{aligned}\Pr(\mathcal{E}) = \Pr(\mathcal{E}|W=1) &\leq P(E_1^c|W=1) + \sum_{i=2}^{2^{nR}} P(E_i|W=1)\\ &\leq \epsilon + \sum_{i=2}^{2^{nR}} 2^{-n(I(X:Y)-3\epsilon)}\\ &= \epsilon + \left(2^{nR-1}\right) 2^{-n(I(X:Y)-3\epsilon)}\\ &\leq \epsilon + 2^{3n\epsilon} 2^{-n(I(X:Y)-R)}\\ &\leq 2\epsilon,\end{aligned}$$

under the condition $R < I(X:Y) - 3\epsilon = C_Q - 3\epsilon$. Hence if $R < C_Q$ we can choose an ϵ and n so the average probability of error, averaged over codebooks is less than 2ϵ. We now remove the average over codebooks. Since the average error rate $P_e(\mathcal{C})$ is small, there exists at least one codebook $\mathcal{C}^*$ with a small average probability of error, with

$$\Pr(\mathcal{E}|\mathcal{C}*) = \frac{1}{2^{nR}} \sum_{i=1}^{2^{Rn}} \lambda_i(\mathcal{C}^*) \leq 2\epsilon.$$

Connection with Algorithmic Information Theory. We now derive the statements of the theorem. Define P to be the probability over codebooks used in earlier in this proof that uses the distribution Q to generate the codewords. Thus $\mathbf{K}(P) <^+ \mathbf{K}(Q,n,R)$. Let D be the set of encoded codebooks that achieve an error rate less than or equal to 2ϵ. By the arguments above, $P(D) \geq 0.5$. This set D is computable from Q, n, R, and $\mathfrak{C}$, with $\mathbf{K}(D|(Q,n,R,\mathfrak{C})) = O(1)$. Thus by Theorem 1 and Lemma 1, there is a codebook $\mathcal{C}^* \in D$ that has an error rate $\leq 2\epsilon$, with

$$\begin{aligned}\mathbf{K}(\mathcal{C}^*) &<^{\log} \mathbf{K}(P) - \log P(D) + \mathbf{I}(D;\mathcal{H})\\ &<^{\log} \mathbf{K}(Q,n,R) + \mathbf{I}((Q,r,n,\mathfrak{C});\mathcal{H}).\end{aligned} \tag{4.2}$$

Thus the sender can use solely $\mathcal{C}^*$ to send messages to the receiver. The receiver needs to determine if sequences are jointly typical, and thus uses $(\mathcal{C}^*, Q, \mathfrak{C})$ to decode the messages. Note that with careful analysis of the proof of Theorem 1 for computable probabilities, one can construct a short program for $\mathcal{C}^*$ (with size less than that of Equation 4.2) that can also compute Q. Thus, we can construct a program p with the properties described in the theorem statement. □

Chapter 5

Deriving Game Derandomization

In this chapter, we prove the two Kolmogorov game derandomization theorems. It involves first derandomizing functions $\mathbb{N} \mapsto \mathbb{N}$ between natural numbers. This chapter uses notions of stochasticity in the field of algorithmic statistics [VS17]. A string x is stochastic, i.e. has a low $\mathbf{Ks}(x)$ score, if it is typical of a simple probability distribution. The deficiency of randomness function of a string x with respect to an elementary probability measure P conditional to $y \in \{0,1\}^*$, is $\mathbf{d}(x|P,y) = \lfloor -\log P(x) \rfloor - \mathbf{K}(x|\langle P\rangle, y)$.

Definition 4 (Stochasticity) *For $x, y \in \{0,1\}^*$,*
$\mathbf{Ks}(x|y) = \min\{\mathbf{K}(P|y) + 3\log\max\{\mathbf{d}(x|P,y),1\} : P$ *is an elementary probability measure*$\}$. $\mathbf{Ks}(x) = \mathbf{Ks}(x|\emptyset)$. $\mathbf{Ks}(a|b) < \mathbf{Ks}(a) + O(\log \mathbf{K}(b))$.

Lemma 4 ([Eps21]) $\mathbf{Ks}(x) <^{\log} \mathbf{I}(x;\mathcal{H})$.

5.1 Function Derandomization

In this section we show how to construct deterministic functions from random ones. The main results of this section are not (directly) implied by the main theorem in [Lev16, Eps19], because Theorem 5 is a statement about probabilities over the Baire space, whereas the result in [Lev16, Eps19] is a statement about lower computable semi measures over $\mathbb{N}$.

We recall the definitions from the introduction. Random functions F over natural numbers are modeled by discrete stochastic processes indexed by $\mathbb{N}$, where each $F(t)$, $t \in \mathbb{N}$, is a random variable over $\mathbb{N}$. $\mathcal{F}$ is the set of all random functions. A random function $F \in \mathcal{F}$ is computable if there is a program that on input $(a_1, \dots, a_n)$ lower computes $\Pr[F(1) = a_1 \cap F(2) = a_2 \cap \dots \cap F(n) = (a_n)]$. Put another way, a random function $F \in \mathcal{F}$ is computable if $X = \Pr[F(a_1) = b_1 \cap \dots \cap F(a_n) = b_n]$ is uniformly computble in $\{(a_i, b_i)\}_{i=1}^n$. The complexity $\mathbf{K}(F)$ of a random function $F \in \mathcal{F}$, is the smallest program that computes X. $\mathcal{G}$ is the set of all deterministic functions $G : \mathbb{N} \to \mathbb{N}$. A sample $S \in \mathcal{S}$ is a finite set of pairs $\{(a_i, b_i)\}_{i=1}^n$. The encoding of a sample is $\langle S\rangle = \langle\{(a_i, b_i\}_{i=1}^n\rangle$. $\mathcal{S}$ is the set of all samples. We say $G(S)$ if G is consistent with S, with $G(a_i) = b_i$, $i = 1, \dots, n$. For random functions, $F(S)$ is the event that F is consistent with S.

To prove function derandomization, we leverage properties about the Baire space $\mathbb{N}^{\mathbb{N}}$. Individual cylinders are $C_n[v] = \{(a_1, a_2, \dots) \in \mathbb{N}^{\mathbb{N}} : a_n = v\}$. Cylinders are generators for cylinder sets. The cylinder sets $C \in \mathcal{C}$ consists of all intersections of a finite number of cylinders. If $C = \bigcap_{i\in I} C_i[v_i]$, then for all $i \in I$, we say $i \in \mathrm{Dom}(C)$. The set of all such cylinder sets provides a basis for the product topology of $\mathbb{N}^{\mathbb{N}}$. The encoding of a cylinder set $C = \bigcap_{i\in I} C_i[v_i]$, is $\langle C\rangle = \langle\{i, v_i\}_{i\in I}\rangle$. The set of all Borel probability measures over $\mathbb{N}^{\mathbb{N}}$ is $\mathcal{P}$. A probability $P \in \mathcal{P}$ is computable if given an encoding of a cylinder set $C \in \mathcal{C}$, $P(C)$ is computable.

We use the following helper proposition and lemma throughout the paper.

Proposition 2
For every $c, n \in \mathbb{N}$, if $x < y + c$ for some $x, y \in \mathbb{N}m$ then $x + n\mathbf{K}(x) < y + n\mathbf{K}(y) + O(n \log n) + 2c$.

Proof. $\mathbf{K}(x) <^+ \mathbf{K}(y) + \mathbf{K}(y - x)$ as x can be computed from y and $(y - x)$. Therefore $n\mathbf{K}(x) - n\mathbf{K}(y) < n\mathbf{K}(y - x) + dn$, for some $d \in \mathbb{N}$ dependent on U. We assume that this equation is not true; then, there exists $x, y, c \in \mathbb{N}$ where $x < y + c$, and $g \leq O(n \log n) + 2c$ where $y - x + g < n\mathbf{K}(x) - n\mathbf{K}(y) < n\mathbf{K}(y - x) + dn$, which is a contradiction for $g =^+ dn + 2c + \max_a\{2n \log a - a\} =^+ dn + 2c + 2n \log n$. □

Theorem 21 *For $F \in \mathcal{F}$, $S \in \mathcal{S}$, if $s = \lceil -\log \Pr[F(S)] \rceil$ and $h = \mathbf{I}(\langle S \rangle; \mathcal{H})$, then $\min_{G \in \mathcal{G}, G(S)} \mathbf{K}(G) < \mathbf{K}(F) + s + h + O(\mathbf{K}(s, h) + \log \mathbf{K}(F))$.*

Proof. Each sample $S \in \mathcal{S}$ where $S = \{(i, v_i)\}_{i \in I}$ can be identified by a cylinder set $C_S \in \mathcal{C}$ where $C_S = \cap_{i \in I} C_i[v_i]$. For every $\alpha \in \mathbb{N}^{\mathbb{N}}$ there is a deterministic function $G_\alpha : \mathbb{N} \to \mathbb{N}$, where $G_\alpha(i) = \alpha[i]$. Furthermore if $\alpha \in C_S$, then for all $(i, v_i) \in S$, $G_\alpha(i) = v_i$. For each random function $F \in \mathcal{F}$, we can identify a Borel probability $P_F \in \mathcal{P}$ over $\mathbb{N}^{\mathbb{N}}$ such that for each sample $S = \{(i, v_i)\}_{i \in I} \in \mathcal{S}$, $\Pr[F(S)] = P_F(C_S)$. This is because random functions and Borel probability measures over $\mathbb{N}^{\mathbb{N}}$ have the same form. Furthermore, if F is computable, then P_F is computable, with

$$\mathbf{K}(P_F | F) = O(1) \tag{5.1}$$

This is because given an encoding $\langle F \rangle$ and an encoded cylinder set $\langle C \rangle$, one can compute $\Pr[F(C)]$, which is equal to $P_F(C)$. Thus given a random function $F \in \mathcal{F}$ and sample $S \in \mathcal{S}$, by Lemma 5 applied to $P_F \in \mathcal{P}$ and $C_S \in \mathcal{C}$, we get the following result, with $h_C = \mathbf{I}(\langle C_S \rangle; \mathcal{H})$, $h_S = \mathbf{I}(\langle S \rangle; \mathcal{H})$, and $s = \lceil -\log P_F(C_S) \rceil$,

$$\begin{aligned}
\min_{\alpha \in C_S} \mathbf{K}(a) <& \mathbf{K}(P_F) + s + h_C + O(\mathbf{K}(s) + \log \mathbf{K}(P_F)) + O(\mathbf{K}(h_C)) \\
\min_{G \in \mathcal{G} : G(S)} \mathbf{K}(G) <& \mathbf{K}(P_F) + s + h_C + O(\mathbf{K}(s) + \log \mathbf{K}(P_F)) + O(\mathbf{K}(h_C)) && (5.2) \\
\min_{G \in \mathcal{G} : G(S)} \mathbf{K}(G) <& \mathbf{K}(F) + s + h_C + O(\mathbf{K}(s) + \log \mathbf{K}(F)) + O(\mathbf{K}(h_C)) && (5.3) \\
\min_{G \in \mathcal{G} : G(S)} \mathbf{K}(G) <& \mathbf{K}(F) + s + h_S + O(\mathbf{K}(s) + \log \mathbf{K}(F)) + O(\mathbf{K}(h_S)) && (5.4) \\
\min_{G \in \mathcal{G}, G(S)} \mathbf{K}(G) <& \mathbf{K}(F) - \log \Pr[F(S)] + \mathbf{I}(\langle S \rangle; \mathcal{H}) && (5.5) \\
& + O(\mathbf{K}(\lceil -\log \Pr[F(S)] \rceil, \mathbf{I}(\langle S \rangle; \mathcal{H})) + \log \mathbf{K}(F)).
\end{aligned}$$

Equation 5.2 is because for the $\alpha \in \mathbb{N}^{\mathbb{N}}$ that minimizes the leftmost term, $G_\alpha \in \mathcal{G}$, with $G_\alpha(S)$ and $\mathbf{K}(G_\alpha) <^+ \mathbf{K}(\alpha)$. Equation 5.3 is because P_F can be constructed from F, i.e. Equation 5.1. Equation 5.4 is due to Proposition 2, Lemma 1 and the fact that $\mathbf{K}(\langle C_S \rangle | \langle S \rangle) = O(1)$. Equation 5.5 is due to the definition of s, where $s = \lceil -\log P_F(C_S) \rceil = \lceil -\log \Pr[F(S)] \rceil$. □

Lemma 5 *For cylinder set $C \in \mathcal{C}$, computable probability $P \in \mathcal{P}$, if $s = \lceil -\log P(C) \rceil$ and $h = \mathbf{I}(\langle C \rangle; \mathcal{H})$, then $\min_{\alpha \in C} \mathbf{K}(\alpha) < \mathbf{K}(P) + s + h + O(\mathbf{K}(s, h) + \log \mathbf{K}(P))$.*

Proof. We put (s, P) on an auxiliary tape to the universal Turing machine U. Thus, all algorithms have access to (s, P), and all complexities implicitly have (s, P) as conditional terms.

Let Q be an elementary probability measure that realizes $\mathbf{Ks}(\langle C\rangle)$. Let $d = \max\{\mathbf{d}(\langle C\rangle|Q), 1\}$ and $c \in \mathbb{N}$ be a constant to be chosen later. Let $n = \max\{m : m \in \mathrm{Dom}(W), W \in \mathcal{C}, \langle W\rangle \in \mathrm{Supp}(Q)\}$. For a list L of a list of numbers and cylinder set $W \in \mathcal{C}$, we say $L \rtimes W$ is the set of all $x \in L$ with $x\mathbb{N}^{\mathbb{N}} \subseteq W$. We define a measure κ over $cd2^s$ lists of lists of n numbers L, where $\kappa(L) = \prod_{i=1}^{cd2^s} P(L[i]\mathbb{N}^{\mathbb{N}})$. Given a list of lists of n numbers L, $\kappa(L)$ is computable (as a program for P is on an auxiliary tape). We use the indicator function $\mathbf{i}(L, W) = [W \in \mathcal{C}, P(W) \geq 2^{-s}, L \rtimes W = \emptyset]$. The function $\mathbf{i}$ is computable, because $P(W)$ and $L \rtimes W$ are computable for all $W \in \mathcal{C}$.

$$\begin{aligned}
\mathbf{E}_{L\sim\kappa}\mathbf{E}_{W\sim Q}[\mathbf{i}(L, W)] &\leq \sum_{W} Q(W) \Pr_{L\sim\kappa}(W \in \mathcal{C}, P(W) \geq 2^{-s}, L \rtimes W = \emptyset)\\
&\leq \sum_{W} Q(W) \prod_{i=1}^{cd2^s}(1 - 2^{-s})\\
&\leq \sum_{W} Q(W)(1 - 2^{-s})^{cd2^s}\\
&< e^{-cd}.
\end{aligned}$$

Thus there exists a list L' of $cd2^s$ sequences of numbers of length n such that $\mathbf{E}_{W\sim Q}[\mathbf{i}(W, L')] = e^{-cd}$. Thus $t(W) = \mathbf{i}(W, L')e^{cd}$ is a Q-test, with $t : \{0, 1\}^* \to \mathbb{R}_{\geq 0}$ and $\sum_W Q(W)t(W) \leq 1$. It must be that $L \rtimes C \neq \emptyset$. Otherwise $t(C) = e^{cd}$, and

$$\begin{aligned}
\mathbf{K}(C|c, d, Q) &<^+ -\log t(C)Q(C)\\
&<^+ -\log Q(C) - (\lg e)cd\\
(\lg e)cd &<^+ -\log Q(C) - \mathbf{K}(C|P) + \mathbf{K}(d, c)\\
(\lg e)cd &< d + \mathbf{K}(d, c) + O(1).
\end{aligned}$$

This is a contradiction for c large enough solely dependent on the universal Turing machine. We roll c into the additive constants of the rest of the proof. Thus there exists $x \in L \rtimes C$ where

$$\begin{aligned}
\mathbf{K}(x) &<^+ \log|L| + \mathbf{K}(L)\\
&<^+ \log|L| + \mathbf{K}(d, Q)\\
&<^+ \log d + s + \mathbf{K}(d) + \mathbf{K}(Q)\\
&<^+ s + 3\log d + \mathbf{K}(Q)\\
&<^+ s + \mathbf{Ks}(C) \qquad (5.6)\\
\min_{\alpha\in C}\mathbf{K}(\alpha) <^+ \mathbf{K}(x) &<^+ s + \mathbf{Ks}(C), \qquad (5.7)
\end{aligned}$$

where Equation 5.6 is due to the definition of stochasticity. Equation 5.7 is because $x\mathbb{N}^{\mathbb{N}} \subseteq C$. Thus making the relativization of (s, P) explicit,

$$\begin{aligned}
\min_{\alpha\in C}\mathbf{K}(\alpha|\langle s, P\rangle) &<^+ s + \mathbf{Ks}(\langle C\rangle|\langle s, P\rangle)\\
\min_{\alpha\in C}\mathbf{K}(\alpha) &< \mathbf{K}(P) + s + \mathbf{Ks}(\langle C\rangle) + O(\mathbf{K}(s) + \log\mathbf{K}(P))\\
\min_{\alpha\in C}\mathbf{K}(\alpha) &< \mathbf{K}(P) + s + \mathbf{I}(\langle C\rangle; \mathcal{H}) + O(\mathbf{K}(s, \mathbf{I}(\langle C\rangle; \mathcal{H})) + \log\mathbf{K}(P)). \qquad (5.8)
\end{aligned}$$

Equation 5.8 follows from Lemma 4, which states $\mathbf{Ks}(x) < \mathbf{I}(x;\mathcal{H}) + O(\mathbf{K}(\mathbf{I}(x;\mathcal{H})))$. □

Theorem 34 can be readily extended to sets of samples $\mathfrak{S} = \{S_1, \dots, S_n\}$, where for deterministic function $G : \mathbb{N} \to \mathbb{N}$, $G(\mathfrak{S})$ if $\bigcup_{i=1}^n G(S_i)$. For random function $F \in \mathcal{F}$, $F(\mathfrak{S})$ is the union of events $F(S_i)$, $i = 1, \dots, n$. The proof of the following corollary follows almost identically to the proofs of Theorem 34 and Lemma 5, noting that $P(\mathfrak{S})$ is computable given a computable probability $P \in \mathcal{P}$ and a finite description of a set of samples $\mathfrak{S}$.

Corollary 1 *For $F \in \mathcal{F}$, if $s = \lceil -\log \Pr[F(\mathfrak{S})] \rceil$ and $h = \mathbf{I}(\langle \mathfrak{S} \rangle; \mathcal{H})$, then $\min_{G\in\mathcal{G}, G(\mathfrak{S})} \mathbf{K}(G) <^{\log} \mathbf{K}(F) + s + h + O(\mathbf{K}(s,h) + \log \mathbf{K}(F))$.*

5.2 Games

Function derandomization has applications to the cybernetic agent model. In this section, we describe two simplified cybernetic agent models. For the first model, the agent $\mathbf{p}$ and environment $\mathbf{q}$ are defined as follows. The agent is a function $\mathbf{p} : (\mathbb{N}\times\mathbb{N})^* \to \mathbb{N}$, where if $\mathbf{p}(w) = a$, $w \in (\mathbb{N}\times\mathbb{N})^*$ is a list of the previous actions of the agent and the environment, and $a \in \mathbb{N}$ is the action to be performed. The environment is of the form $\mathbf{q} : (\mathbb{N}\times\mathbb{N})^* \times \mathbb{N} \to \mathbb{N} \cup \{\mathbf{W}\}$, where if $\mathbf{q}(w,a) = b \in \mathbb{N}$, then b is $\mathbf{q}$'s response to the agent's action a, given history w, and the game continues. If $\mathbf{q}$ responds $\mathbf{W}$ then the agents wins and the game halts. The agent can be randomized. The game can continue forever, given certain agents and environments. This is called a win/no-halt game.

The following theorem is a game-theoretic interpretation of Lemma 6 in [VV10].

Theorem 22 *If 2^r deterministic agents of complexity $< k$ win against environment $\mathbf{q}$, then there is a deterministic agent $\mathbf{p}$ of complexity $<^{\log} k - r + \mathbf{I}(\langle r, k, \mathbf{q}\rangle; \mathcal{H})$ that wins against $\mathbf{q}$.*

Proof. Given $\langle r, k, \mathbf{q}\rangle$, one can construct a finite set D of encoded agents that win against $\mathbf{q}$ and D contains at least 2^r agents of complexity $< k$. Furthermore $\sum_{x\in D} \mathbf{m}(x) \overset{*}{>} 2^r 2^{-k}$, so using Theorem 1, there is an agent $\mathbf{p} \in D$, where, using Lemma 1, $\mathbf{K}(\mathbf{p}) <^{\log} k - r + \mathbf{I}(D;\mathcal{H}) <^{\log} k - r + \mathbf{I}(\langle r, k, \mathbf{q}\rangle; \mathcal{H})$. □

Theorem 23 *If probabilistic agent $\mathbf{p}'$ wins against environment $\mathbf{q}$ with at least probability p, then there is a deterministic agent $\mathbf{p}$ of complexity $<^{\log} \mathbf{K}(\mathbf{p}') - \log p + \mathbf{I}(\langle p, \mathbf{p}', \mathbf{q}\rangle; \mathcal{H})$ that wins against $\mathbf{q}$.*

Proof. Let $\mathcal{I}$ be a set of interactions between an arbitrary agent and the environment $\mathbf{q}$ such that each interaction ends in $\mathbf{W}$ and with probability $> p/2$, $\mathbf{p}'$ will act according to an interaction in $\mathcal{I}$. Thus $\mathbf{K}(\mathcal{I}|p, \mathbf{p}', \mathbf{q}) = O(1)$. $\mathbf{p}'$ can be encoded into a random function F, where the domain $(\mathbb{N}\times\mathbb{N})^*$ of $\mathbf{p}'$ can be encoded into a single number $\mathbb{N}$. $\mathbf{K}(F|\mathbf{p}') = O(1)$. Similarly, $\mathcal{I}$ can be encoded into a set of samples $\mathfrak{C}$, where $\Pr[F(\mathfrak{C})] > p/2$ and $\mathbf{K}(\langle\mathfrak{C}\rangle|\langle\mathcal{I}\rangle) = O(1)$. Using Corollary 1, there is a deterministic function $G : \mathbb{N} \to \mathbb{N}$, such that

$$\begin{aligned}
\mathbf{K}(G) &<^{\log} \mathbf{K}(F) - \log[F(\mathfrak{C})] + \mathbf{I}(\langle\mathfrak{C}\rangle;\mathcal{H}) \\
&<^{\log} \mathbf{K}(\mathbf{p}') - \log[F(\mathfrak{C})] + \mathbf{I}(\langle\mathfrak{C}\rangle;\mathcal{H}) \\
&<^{\log} \mathbf{K}(\mathbf{p}') - \log p + \mathbf{I}(\langle\mathfrak{C}\rangle;\mathcal{H}) \\
&<^{\log} \mathbf{K}(\mathbf{p}') - \log p + \mathbf{I}(\langle\mathcal{I}\rangle;\mathcal{H}) && (5.9) \\
&<^{\log} \mathbf{K}(\mathbf{p}') - \log p + \mathbf{I}(\langle p, \mathbf{p}', \mathbf{q}\rangle;\mathcal{H}), && (5.10)
\end{aligned}$$

where Equations 5.9 and 5.10 are due to Lemma 1. The deterministic function G is an encoding of an agent, $\mathbf{p}$, proving the theorem. □

The second game derandomization theorem is modified such that the environment gives a non-negative rational penalty term to the agent at each round. Furthermore the environment specifies an end to the game without specifying a winner or loser. This is called a penalty game.

Corollary 2 *If given probabilistic agent* $\mathbf{p}$*, environment* $\mathbf{q}$ *halts with probability 1, and* $\mathbf{p}$ *has expected penalty less than* $n \in \mathbb{N}$*, then there is a deterministic agent of complexity* $<^{\log} \mathbf{K}(\mathbf{p}) + \mathbf{I}(\langle \mathbf{p}, n, \mathbf{q}\rangle; \mathcal{H})$ *that receives penalty* $< 2n$ *against* $\mathbf{q}$.

Proof. We create a win/no-halt game from $\mathbf{q}$ where an agent wins if it gets a penalty less than $2n$. Thus $\mathbf{p}$ is a probabilistic agent that wins this new game with probability $> .5$. Theorem 23 then can be used to prove the corollary. □

Chapter 6

Game Derandomization

> *For a given win/no-halt game if there is a good simple randomized player but no simple winning deterministic player, then that game is exotic, in that it has high mutual information with the halting sequence.*

This mirrors derandomization, where if a solution to a problem can be produced easily with a simple probability, but has no simple solutions, then it is exotic. A game instance that has a simple winning probabilistic agent but no simple winning deterministic agent can be found in Example 1.

6.1 Even-Odds

We define the following win/no-halt game, entitled Even-Odds. There are N rounds. At round 1, the environment $\mathbf{q}$ secretly records bit $e_1 \in \{0,1\}$. It sends an empty message to the agent who responds with bit $a_1 \in \{0,1\}$. The agent gets a point if $e_1 \oplus b_1 = 1$. Otherwise the agent loses a point. For round i, the environment secretly selects a bit e_i that is a function of the previous agent's actions $\{a_j\}_{j=1}^{i-1}$ and sends an empty message to the agent, which responds with a_i and the agent gets a point if $e_i \oplus a_i = 1$, otherwise it loses a point. The agent wins after N rounds if it has a score of at least $\sqrt{N}$.

Theorem 24 *For large enough N, there is a deterministic agent $\mathbf{p}$ that can win* Even-Odds *with N rounds, with complexity $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q};\mathcal{H})$.*

Proof. We describe a probabilistic agent $\mathbf{p}'$. At round i, $\mathbf{p}'$ submits 0 with probability 1/2. Otherwise it submits 1. By the central limit theorem, for large enough N, the score of the probabilistic agent divided by $\sqrt{N}$ is $S \sim \mathcal{N}(0,1)$. Let $\Phi(x) = \Pr[S > x]$. A common bound for $\Phi(x)$ is

$$\Phi(x) > \frac{1}{2\pi}\frac{x}{x^2+1}e^{-x^2/2}$$
$$\Phi(1) > \frac{1}{4\pi}e^{-1/2} > \frac{1}{8\pi}.$$

Thus when $S \geq 1$, the score is at least $\sqrt{N}$. Thus $\mathbf{p}'$ wins with probability at least $p = \frac{1}{8\pi}$. Thus by Theorem 23, there exists a deterministic agent $\mathbf{p}$ that can beat $\mathbf{q}$ with complexity

$$\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{K}(\mathbf{p}') - \log p + \mathbf{I}((p,\mathbf{p}',\mathbf{q});\mathcal{H}) <^{\log} \mathbf{I}(\mathbf{q};\mathcal{H}).$$

□

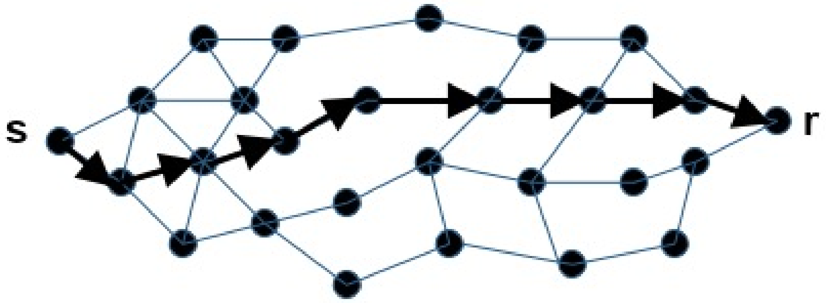


Figure 6.1: A graphical depiction of a winning deterministic player to the Graph-Navigation game. The player starts at s and chooses a path to reach the goal state r, (assuming $t_G = 8$).

6.2 Graph-Navigation

The win/no-halt game is as follows. The environment $\mathbf{q}$ consists of (G, s, r). $G = (E, V)$ is a non-bipartite graph with undirected edges, $s \in V$ is the starting vertex, and $r \in V$ is the goal vertex. Let t_G be the time it takes for any random walk starting anywhere to converge to the stationary distribution $\pi(v)$, for all $v \in V$, up to a factor of 2.

There are t_G rounds and the agent starts at $s \in V$. At round 1, the environment gives the agent the degree $s \in V$, Deg(s). The agent picks an number between 1 and Deg(s) and sends it to $\mathbf{q}$. The agent moves along the edge the number is mapped to and is given the degree of the next vertex it is on. Each round's mapping of numbers to edges to be a function of the agent's past actions. This process is repeated t_G times. The agent wins if it is on $r \in V$ at the end of round t_G. A graphical depiction of this can be seen in Figure 6.1.

Theorem 25 *There is a deterministic agent* $\mathbf{p}$ *that can win the* Graph-Navigation *game with complexity* $\mathbf{K}(\mathbf{p}) <^{\log} \log |E| + \mathbf{I}((G, s, r); \mathcal{H})$.

Proof. It is well known if G is non-bipartite, a random walk starting from any vertex will converge to a stationary distribution $\pi(v) = \deg(v)/2|E|$, for each $v \in V$.

A probabilistic agent $\mathbf{p}'$ is defined as selecting each edge with equal probability. After t_G rounds, the probability that $\mathbf{p}'$ is on the goal r is close to the stationary distribution π. More specifically the probability is $\overset{*}{>} |E|^{-1}$. Thus by Theorem 23, there is a deterministic agent $\mathbf{p}$ that can find r in t_G turns and has complexity $\mathbf{K}(\mathbf{p}') <^{\log} \log |E| + \mathbf{I}((G, s, t); \mathcal{H})$.

6.3 Interactive-k-Sat

The penalty game Interactive-k-Sat is as follows. The environment $\mathbf{q}$ has access to a hidden k-Sat formula, with n variables and some number of clauses, each containing k literals, which are a variable instance or its negation. Each variable appears in at most $2^k/ke$ clauses. The environment's first action is to send the number of variables to the agent. After this step, the agent, $\mathbf{p}$ has n variables, each initially set to true. At each subsequent round, the environment gives to the agent the clauses which are not satisfied. The player can change up to k variables,

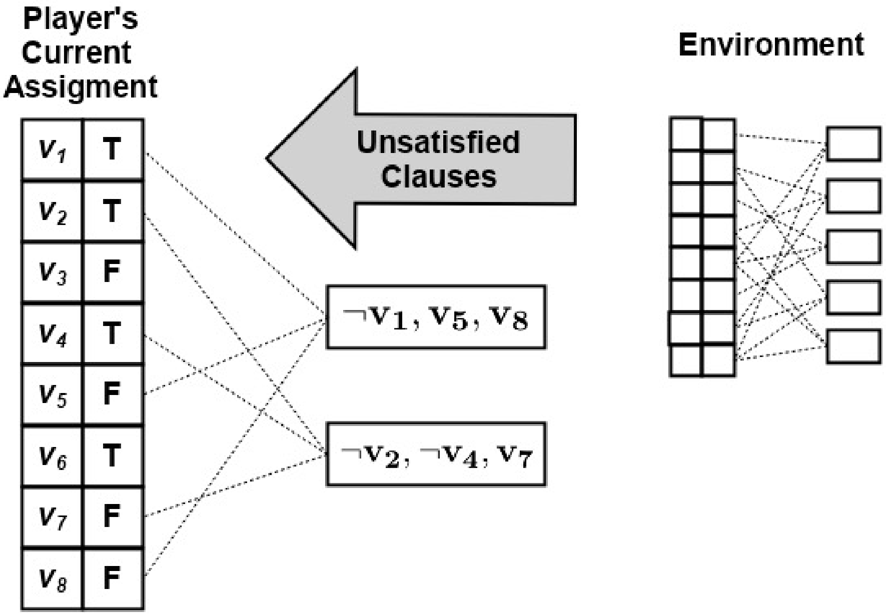


Figure 6.2: A graphical depiction of a turn in the Interactive-k-Sat game. The player's current assignment of variables has resulted in two unsatisfied clauses, which the environment sends to the player. The environment has a complete picture of the variables and clauses, which are hidden to the player.

and sends these changes to the environment. If the k-Sat formula is satisfied, the game stops. Otherwise the game continues. When the game ends, the penalty is the number of terms. A graphical representation of this game can be seen in Figure 6.2. Obviously there is a deterministic player of complexity $O(1)$ that can try every possible assignment of variables. This is a winning strategy of at most 2^n turns. However, the following theorem shows that a much more successful player exists without much more complexity bounds.

Theorem 26 *There exists a deterministic player* $\mathbf{p}$ *that can achieve a penalty of* $(1+\epsilon)(n/2k + n/(2^k/e-k))$ *with complexity* $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q};\mathcal{H}) - \log\epsilon$*, for* $\epsilon \in (0,1)$*.*

Proof. We use the Algorithmic Lovász Local Lemma, from [MT10]. The randomized algorithm, when applied to k-Sat is as follows.

1. Given is a random assignment of n variables.
2. While there exists an unsatisfied clause.
   (a) Pick an unsatisfied clause at random.
   (b) Reassign the variables of this clause randomly.
3. Return the variable assignment.

It was proved in [MT10] that this algorithm has n/D expected steps, where D is the size of the dependency between events. So $D = (2^k/e - k)$. Thus the goal of this proof is to construct a randomized player $\mathbf{p}'$ that simulates the above randomized algorithm. The player $\mathbf{p}'$ first starts out by randomizing its variables. Thus at each turn in this phase, it selects up to k untouched variables and gives them random assignments. This takes at most $\lceil n/k \rceil$ steps. However, noting that player $\mathbf{p}'$ only needs to update the variables that are set to true, this task takes $n/2k$ expected steps. Once this is complete, whenever the environment sends back unsatsified clauses, the randomized agent chooses one at random and randomly resamples its k variables. Thus, by [MT10], this has $n/(2^k/e - k)$ expected steps before a satisfying assignment is found and the game halts. So the randomized player runs in $(n/2k + n/(2^k/e - k))$ expected steps. Thus by Theorem **??**, there is a deterministic player that can find a satisfying assignment in less than $(1+\epsilon)(n/2k + n/(2^k/e - k))$ turns with complexity

$$\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q};\mathcal{H}) - \log \epsilon.$$

□

Note that as the game proceeds, the environment reveals more and more information about its hidden k-Sat formula to the player. This leaves open the possibility of a deterministic player of Kolmogorov complexity $O(1)$ that coerces the environment to reveal all the clauses and then manually set the corresponding satisfying assignment. Whether or not this will take less steps than that proved in the above theorem is unknown.

6.4 Penalty-Tests

An example penalty game is as follows. The environment $\mathbf{q}$ plays a game for N rounds, for some very large $N \in \mathbb{N}$, with each round starting with an action by $\mathbf{q}$. At round i, the environment gives, to the agent, an encoding of a program to compute a probability P_i over $\mathbb{N}$. The choice of P_i can be a computable function of i and the agent's previous turns. The agent responds with a number $a_i \in \mathbb{N}$. The environment gives the agent a penalty of size $T_i(a_i)$, where $T_i : \mathbb{N} \to \mathbb{Q}_{\geq 0}$ is a computable test, with $\sum_{a\in\mathbb{N}} P_i(a)T_i(a) < 1$. After N rounds, $\mathbf{q}$ halts. A graphical representation of this game can be found in Figure 6.3.

Theorem 27 *There is a deterministic agent* $\mathbf{p}$ *that can receive a penalty* $< (1+\epsilon)N$ *and has complexity* $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q};\mathcal{H}) - \log \epsilon$*, for* $\epsilon \in (0,1)$*.*

Proof. A very successful probabilistic agent $\mathbf{p}'$ can be defined. Its algorithm is simple. On receipt of a program to compute P_i, the agent randomly samples a number $\mathbb{N}$ according to P_i. At each round the expected penalty is $\sum_a P_i(a)T_i(a) < 1$, so the expected penalty of $\mathbf{p}$ for the entire game is $< N$. Thus by Theorem **??**, there is a deterministic agent $\mathbf{p}$ such that

1. The agent $\mathbf{p}$ receives a penalty of $< (1+\epsilon)N$,
2. $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q};\mathcal{H}) - \log \epsilon$.

□

Example 1 *Let* $\mathbf{q}$ *be defined so that* $P_i(a) = [a \leq 2^i]2^{-i}$ *and* $T_i = [a \leq 2^i]2^{i-\mathbf{K}(a|i)}$*, where* $[A] = 1$ *if* A *is true, and 0 otherwwise. Thus each* T_i *is a randomness deficiency function (to the power of 2). The probabilistic algorithm* $\mathbf{p}'$ *will receive an expected penalty* $< N$*. However any deterministic agent* $\mathbf{p}$ *that receives a penalty* $< 2N$ *must be very complex, as it must select many numbers with low randomness deficiency. Thus, by the bounds above,* $\mathbf{I}(\mathbf{q};\mathcal{H})$ *must be very high. This makes sense because* $\mathbf{q}$ *encodes* N *randomness deficiency functions.*

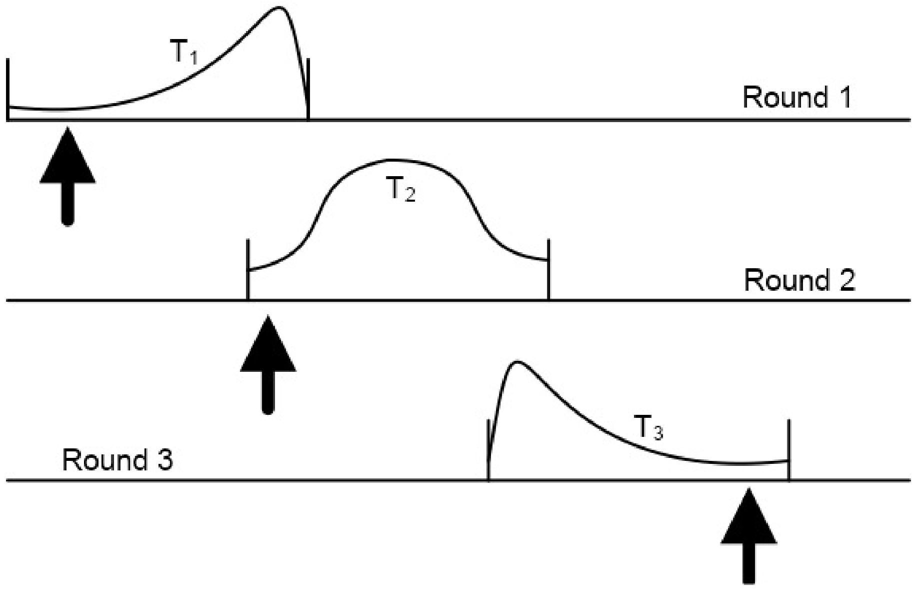


Figure 6.3: Three rounds of the PENALTY-TEST game. At each round i the probability, P_i, that the environment gives to the player is a uniform measure over a unique interval. The environment has three tests $\{T_1, T_2, T_3\}$, which is not shared with the player, that represent the penalties. In this depiction, the player's moves are numbers in the interval (represented by the arrows) and result in low total penalty.

6.5 SET-SUBSET

We define the following win/no-halt game, entitled SET-SUBSET. There are k rounds. At round $i = \{1, \ldots, k\}$, the environment $\mathbf{q}$ gives n numbers $A_i \subset \mathbb{N}$ to the agent $\mathbf{p}$. The environment secretly selects $m \leq n$ numbers $B_i \subseteq A_i$. The player selects a number $a_i \in A_i$. Each A_i and B_i are a function of the player's previous actions. The player wins if for every round, his selection a_i is in the secret set B_i. So for all $i \in \{1, \ldots, k\}$, $a_i \in B_i$. A graphical depiction of this game can be seen in Figure 6.4.

Theorem 28 *There is a deterministic agent* $\mathbf{p}$ *that win against* SET-SUBSET *environment* $\mathbf{q}$*, where* $\mathbf{K}(\mathbf{p}) <^{\log} k \log(n/m) + \mathbf{I}(\mathbf{q}; \mathcal{H})$*.*

Proof. Let $\mathbf{p}'$ be the randomized the player that selects a member of the given set with A_i with uniform probability. The probability that $\mathbf{p}'$ picks a member of B_i is $|B_i|/|A_i| = m/n$. The probability that $\mathbf{p}'$ picks a member of B_i for all $i \in \{1, \ldots, k\}$ is $(m/n)^k$, which is the probability that $\mathbf{p}'$ wins. $\mathbf{K}(\mathbf{p}') = O(1)$. Thus by Theorem 23, there exists a deterministic player $\mathbf{p}$ that wins against $\mathbf{q}$ with complexity bounded by the theorem statement. □

6.6 INTERACTIVE-HYPERGRAPH

We define the following penalty game, entitled INTERACTIVE-HYPERGRAPH. The environment has access to a hidden k-regular -hypergraph. A *hypergraph* is a pair $J = (V, E)$ of vertices V and edges $E \subseteq \mathcal{P}(V)$. Thus each edge can connect ≥ 2 vertices. A hypergraph is *k-regular* of the size $|e| = k$ for all edges $e \in E$. A 2-regular hypergraph is just a simple graph. The player has access

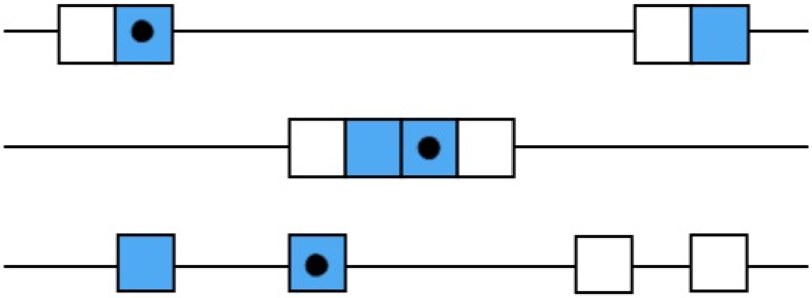


Figure 6.4: A graphical depiction of the Set-Subset game. Each line represents a round of the game. The boxes of the ith line represent A_i, and the filled boxes represent the secret set $B_i \subset A_i$. A winning player is shown, by placing a circle in B_i for each round i.

to a list of vertices and the goal of the player is to produce a valid 2-coloring of the hypergraph. A valid 2-coloring of a hypergraph (V, E) is a mapping $f : V \to \{1, 2\}$ where every edge $e \in E$ is not *monochromatic* $|\{f(v) : v \in e\}| = 2$. We assume that for each edge f of the graph, there are at most $2^{k-1}/e - 1$ edges h such that $f \cap h \neq \emptyset$.

The game proceeds as follows. For the first round, environment gives the number of vertices to the player. The player has n vertices, each with starting color 1. At each subsequent turn, the environment sends to the player the edges which are monochromatic. The player can change the color of up to k vertices and sends these changes to the environment. The game ends when the player has a valid 2-coloring of the graph.

Theorem 29 *There exists a deterministic player* $\mathbf{p}$ *that can beat the environment* $\mathbf{q}$ *in* $(1 + \epsilon)(n/2k + n/(2^{k-1}/e - 1))$ *turns of complextiy* $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q}; \mathcal{H}) - \log \epsilon$.

Proof. We use the Algorithmic Lovász Local Lemma, from [MT10]. The randomized algorithm, when applied to hypergraphs is as follows.

1. Given is a random 2-color assignment of n vertices.
2. While there exists a monochromatic edge.
   (a) Pick a monochromatic edge random.
   (b) Reassign the colors of the vertices of this edge randomly.
3. Return the valid 2-coloring.

It was proved in [MT10] that this algorithm has n/D expected steps, where D is the size of dependency between the events. So $D = 2^{k-1}/e - 1$. Thus the goal of this proof is to construct a randomized player $\mathbf{p}'$ that simulates the above randomized algorithm. The player $\mathbf{p}'$ first starts out by randomizing the color assignments of each vertex. Thus at each turn in this phase, it selects up to k untouched vertices and gives them random assignments. This takes at most $\lceil n/k \rceil$ steps. However, noting that player $\mathbf{p}'$ only needs to update the colors that are set to 2, this task takes $n/2k$ expected steps. Once this is complete, whenever the environment sends back unsatsified

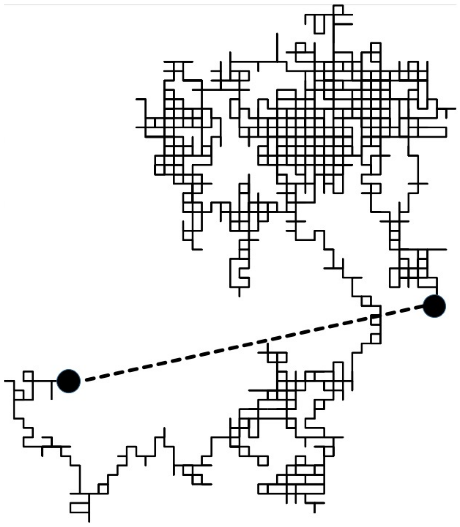


Figure 6.5: A graphical depiction of a random two dimensional walk. The radius r is the length of the line between the starting and ending points.

edges, the randomized agent chooses one edge at random and randomly recolors its k vertices. Thus, by [MT10], this has $n/D = n/(2^{k-1}/e - 1)$ expected steps before a valid two-coloring is found and the game halts. So the randomized player runs in $(n/2k + n/(2^{k-1}/e - 1))$ expected steps. Thus by Theorem **??**, there is a deterministic player that can find a svalid 2-coloring in less than $(1 + \epsilon)(n/2k + n/(2^{k-1}/e - 1))$ turns with complexity

$$\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q}; \mathcal{H}) - \log \epsilon.$$

□

6.7 Grid-Walk

We define the following win/no-halt game, entitled Grid-Walk. The player starts at the origin of a two dimension grid with integer coordinates. At the start of the turn, the player chooses a number from $\{1, \dots, 4\}$ and the the environment maps this number to a direction (North, South, East, West) based on a function of the player's previous actions. The enviroment moves the player in the choseen direction and the next round begins. No messages are sent from the environment to the player. The radius of the player is its Euclidean distance to the origin. The player wins if after N rounds, it has a radius at least $\sqrt{N}$. An example random walk can be seen in figure 6.5.

Theorem 30 *For large enough N, there is a deterministic agent* $\mathbf{p}$ *that can win against* Grid-Walk *environment* $\mathbf{q}$ *such that* $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}((\mathbf{q}, N); \mathcal{H})$.

Proof. We define a probabilistic player $\mathbf{p}'$ that randomly chooses a number from $\{1, 2, 3, 4\}$ with uniform probability. Thus $\mathbf{p}'$ performs a random walk on the two dimensional grid. For large enough N, the probability density function for the radius r of random walker $\mathbf{p}'$ is the Rayleigh density function $P(r) = \frac{2r}{N} e^{-r^2/N}$. The culmulative distribution function is $F(r) = 1 - e^{-r^2/N}$. Thus $c = 1 - F(\sqrt{N}) = 1 - e^{-1}$. Thus with at least constant probability c the radius of $\mathbf{p}'$ is at least $\sqrt{N}$, and $\mathbf{p}'$ wins. So by Theorem 23, there exists a deterministic player $\mathbf{p}$ that wins against

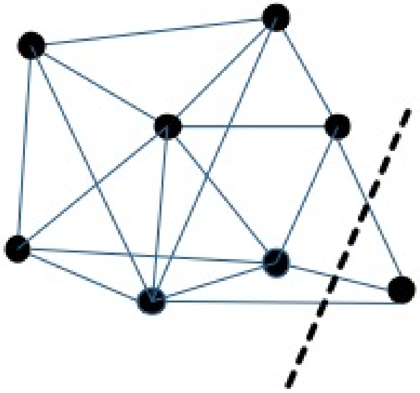


Figure 6.6: A graphical depiction of a minimum cut. By removing the edges along the dotted line, two components are created.

q where

$$\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{K}(\mathbf{p}') - \log c + \mathbf{I}((\mathbf{p}', N, \mathbf{q}); \mathcal{H}) <^{\log} \mathbf{I}((N, \mathbf{q}); \mathcal{H}).$$

□

6.8 Min-Cut

We define the following win/no-halt game, entitled Min-Cut. The game is defined by an undirected graph G and a mapping ℓ from numbers to edges. At round i, the environment **q** sends the number of edges of G. The player responds with a number. The environment maps the number to an edge, and this mapping can be dependent on the player's previous actions. The environment then contracts the graph G along the edge. The game halts when the graph G has contracted into two vertices. The player wins if the cut represented by the contractions is a min cut. A minimum cut of a graph is the minimum number of edges, that when removed from the graph, produces two components. A graphical depiction of a min cut can be seen in Figure 6.6.

Theorem 31 *There is a deterministic agent* **p** *that can win against* Min-Cut *instance* (G, S, ℓ)*,* $|G| = n$*, such that* $\mathbf{K}(\mathbf{p}) <^{\log} 2\log n + \mathbf{I}((G, \ell); \mathcal{H})$*.*

Proof. We define the following randomized agent $\mathbf{p}'$. At each round, $\mathbf{p}'$ chooses an edge at random. Thus the interactions of $\mathbf{p}'$ and **q** represent an implementation of Karger's algorithm. Karger's algorithm has an $\Omega(1/n^2)$ probability of returning a min-cut. Thus $\mathbf{p}'$ has an $\Omega(1/n^2)$ chance of winning. By Theorem 23, there exist a deterministic agent **p** and c where **p** can beat **q** and has complexity $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{K}(p') - \log c/n^2 + \mathbf{I}(\mathbf{q}; \mathcal{H}) <^{\log} 2\log n + \mathbf{I}((G, \ell); \mathcal{H})$. □

6.9 Cover-Time

We define the following interactive penalty game. Let $G = (E, V)$ be a graph consisting of n vertices V and undirected edges E. The environment **q** consists of (G, s, ℓ). $G = (E, V)$ is a non-bipartite graph with undirected edges, $s \in V$ is the starting vertex. ℓ is a mapping from numbers to edges to be described later.

The agent starts at $s \in V$. At round 1, the environment gives the agent the degree $s \in V$, $\mathrm{Deg}(s)$. The agent picks a number between 1 and $\mathrm{Deg}(s)$ and sends it to **q**. The agent moves along

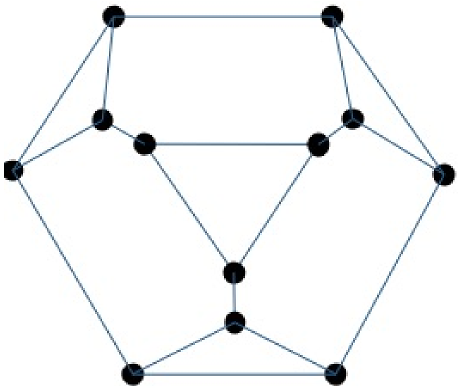


Figure 6.7: An example vertex-transitive graph.

the edge the number is mapped to and is given the degree of the next vertex it is on. Each round's mapping of numbers to edges, ℓ, is a computable function of the agent's past actions. The game stops if the agent has visited all vertices and the penalty is the number of turns the agents takes.

Theorem 32 *There is a deterministic agent* $\mathbf{p}$ *that can play against* Cover-Time *instance* (G, S, ℓ), $|G| = n$, *and achieve penalty* $\frac{8}{27}n^3 + o(n^3)$ *and* $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}((G, s, \ell); \mathcal{H})$.

Proof. A probabilistic agent $\mathbf{p}'$ is defined as selecting each edge with equal probability. Thus the agent performs a random walk. The game halts with probability 1. Due to [Fei95], the expected time (i.e. expected penalty) it takes to reach all vertices is $\frac{4}{27}n^3 + o(n^3)$. Thus by Theorem **??** there is a deterministic agent $\mathbf{p}$ that can reach each vertex with a penalty of $\frac{8}{27}n^3 + o(n^3)$ and has complexity

$$\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{K}(\mathbf{p}') + \mathbf{I}((G, s, \ell); \mathcal{H}) <^{\log} \mathbf{I}((G, s, \ell); \mathcal{H}).$$

□

6.10 Vertex-Transitive-Graph

We describe the following graph based game. The environment $\mathbf{q} = (G, \ell, u, 2k)$ consists of an undirected vertex-transitive graph $G = (V, E)$, a start vertex $u \in V$, the number of rounds $2k$, and a mapping ℓ from numbers to vertices. A vertex-transitive graph $G = (V, E)$ has the property that for any vertices $u, v \in V$, there is an automorphism of G that maps u into v. An example of a vertex transitive graph can be seen in Figure 6.7. At round 1, the agent starts at vertex $u \in V$ and the environment send to the agent the degree of u. The agent picks a number from 1 to $\mathrm{Deg}(u)$ and the environment moves the agent along the edge specified by the mapping ℓ from numbers to edges. The mapping ℓ can be a function of the agents previous actions. The agent wins if after $2k$ rounds, the agent is back at u.

Theorem 33 *There is a deterministic agent* $\mathbf{p}$ *that can win at the* Vertex-Transitive-Graph *game* $(G = (V, E), \ell, u, k)$, $|V| = n$ *with complexity* $\mathbf{K}(\mathbf{p}) <^{\log} \log n + \mathbf{I}((G, \ell, u, k); \mathcal{H})$.

Proof. We define the following randomized agent $\mathbf{p}'$. At each round, after being given the degree d of the current vertex, $\mathbf{p}'$ chooses a number randomly from 1 to d. $\mathbf{K}(\mathbf{p}') = O(1)$. This is equivalent to a random walk on G. Let $P^l(u, v)$ denote the probability that a random walk of length l starting at u ends at v. Then due to [AS04], for vertex-transitive graph G,

$$P^{2k}(u, u) \geq P^{2k}(u, v).$$

So after $2k$ rounds the randomized agent is back at u with probability $P^{2k}(u, u) \geq 1/n$, which lower bounds the winning probability of $\mathbf{p}'$ against $\mathbf{q}$. By Theorem 23, there exists a deterministic agent $\mathbf{p}$ that can beat $\mathbf{q}$ with complexity

$$\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{K}(\mathbf{p}') + n + \mathbf{I}((n, \mathbf{p}', \mathbf{q}); \mathcal{H}) <^{\log} n + \mathbf{I}((G, \ell, u, k); \mathcal{H}).$$

□

Chapter 7

Resource Bounded Derandomization

7.1 Resource Bounded EL Theorem

In this section we derive the resource bounded EL theorem. We also derive an interesting corollary to Theorem 4.1 in [AF09] which states to invert a hash function $f^{-1}(x)$, one can find a secret key π of size approximiately equal to x that will efficiently decompress to a pre-image of x with respect to f. The results in this section are not unconditional, they require the existence of the pseudorandom generator, introduced in [Nis94].

Assumption 1 ***Crypto*** *is the assumption that there exists a language in* $\mathbf{DTIME}(2^{O(n)})$ *that does not have size* $2^{o(n)}$ *circuits with* Σ_2^p *gates. This asssumption is need in the proof of Theorem 34 in [AF09] to assume the existence of a pseudorandom generator* $g:\{0,1\}^{k\log n}\rightarrow\{0,1\}^n$*, computable in time polynomial in* n*.*

Definition 5 $\mathbf{FP}'=\{f : f\in\mathbf{FP}$ *and* $\|x\|=\|f(x)\|\}$.

Definition 6 *For* $A\in\mathbf{FP}'$ *we say that* A *samples* $D\subset\{0,1\}^n$ *with probability* γ*, if* $|\{0,1\}^n\cap A^{-1}(D)|/2^n>\gamma$*.*

Theorem 34 **([AF09])** *Assume* ***Crypto****. Let* $F\in\mathbf{FP}'$*. Let* $m,n\in\mathbb{N}$ *where* $\{0,1\}^n\supseteq f(\{0,1\}^m)$*. Let* $T_y=\{w\in\{0,1\}^m : F(w)=y\}$ *and* $V_k=\{y:\|y\|=n$ *and* $|T_y|\geq 2^k\}$*. There exists a function*

$$G:\Sigma^{m-k+O(\log m)}\rightarrow\Sigma^m$$

computable in polynomial time such that for all $y\in V_k$*,* $\mathrm{range}(G)\cap T_y\neq\emptyset$*.*

Remark 1 *In the previous theorem, the running time of* G *is a polynomial function of the running time of* F*. This was noted in [LOZ22]. In addition, in subsequent theorems and corollaries of this section, the polynomial time function* p *in the resource bounded complexity* $\mathbf{K}^p$ *is a polynomial function of the running times of the algorithms of the theorem/corollary statements. Furthermore, due to [AF09],* G *can be encoded in* $O(1)$ *bits.*

The following corollary implies that to invert x with a hash function f, one can find a secret key π of size approximately equal to x that efficiently expands to an element in $f^{-1}(x)$.

Corollary 3 *Assume* ***Crypto****. Let* $f\in\mathbf{FP}'$*, where* $f(\{0,1\}^n)\subseteq\{0,1\}^{n-k}$*. Then for some polynomial* p *where for* $\{0,1\}^n\supseteq D=f^{-1}(x)$*,*

$$\min_{y\in D}\mathbf{K}^p(y)=n-\log|D|+O(\log n).$$

Proof. Follows directly from Theorem 34. □

Corollary 4 (Resource EL) *Assume* ***Crypto****. Let $L \in \mathbf{P}$, $A \in \mathbf{FP}'$, and assume A samples L_n with probability δ_n. Then for some polynomial p,*

$$\min_{x \in L_n} \mathbf{K}^p(x) < -\log \delta_n + O(\log n).$$

Proof. Let $F \in \mathbf{FP}'$ where $F(\{0,1\}^n) \subseteq \{0,1\}^n$ and for $x \in \{0,1\}^n$, $F(x) = 1^n$ if $A(x) \in L_n$ and $F(x) = 0^n$ otherwise. Let $k \in \mathbb{N}$ be maximal such that $\delta_n \geq 2^{k-n}$. Let $\ell = n - k + O(c \log n)$. By Theorem 34, there exists a function $G : \{0,1\}^\ell \to \{0,1\}^n$ running in polynomial time such that there exists $x \in \ell$, with $G(x) = 1^n$. This is because $1^n \in T_k$, using the definition in Theorem 34, because A produces a member of L_n with probability at least δ_n and all of L_n is mapped to 1^n. We define a program P that uses G to map x to a string y, then use A to map y to a string $z \in L_n$. This program P is of size ℓ and runs in polynomial time. □

A verifier $V : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ is a function computable in polynomial time with respect to the first argument. For a given x, $\text{Proofs}(x) = \{y : V(x,y) = 1\}$.

Corollary 5 *Assume* ***Crypto****. Let $\{x_n\}$ be uniformly computable in polynomial time. For a verifier $V(x,y)$, let $A \in \mathbf{FP}'$ sample* $\text{Proofs}(x_n)$ *with probability γ_n. Thus there is a polynomial p and $y \in \text{Proofs}(x_n)$ with*

$$\mathbf{K}^p(y) < -\log \gamma_n + O(\log n).$$

7.2 Resource Bounded Derandomization

In this section, we use Corollary 5 to produce three examples of resource bounded derandomization.

Lemma 6 (Lovasz Local Lemma) *Let $E_1, \ldots, E_n$ be a collection of events such that $\forall i : \Pr[E_i] \leq p$. Suppose further that each event is dependent on at most d other events, and that $ep(d+1) \leq 1$. Then,* $\Pr\left[\bigcap_i \overline{E}_i\right] > \left(1 - \frac{1}{d+1}\right)^n$.

Proposition 3 (Mutual Independence Principle) *Suppose that $Z_1, \ldots Z_m$ is an underlying sequence of independent events and suppose that each event A_i is completely determined by some subset $S_i \subset \{Z_1, \ldots, Z_m\}$. If $S_i \cap S_j = \emptyset$ for $j = j_1, \ldots, j_k$ then A_i is mutually independent of $\{A_{j_1}, \ldots, A_{j_k}\}$.*

7.2.1 Vertex-Disjoint-Cycles

Theorem 35 *Assume **Crypto**. Let $\{G_n\}$ be a uniformly computable in polynomial time sequence of k-regular graphs, with $k \geq 5$. There is a polynomial p where for each G_n, there is a partition x of $\lfloor \frac{k}{3\ln k} \rfloor$ components each containing a cycle with*

$$\mathbf{K}^p(x) < 2n/k^2 + O(\log n).$$

Proof. We partition the vertices of G into $c = \lfloor k/3\ln k \rfloor$ components by assigning each vertex to a component chosen independently and uniformly at random. With positive probability, we show that every component contains a cycle. It is sufficient to prove that every vertex has an edge leading to another vertex in the same component. This implies that starting at any vertex there exists a path of arbitrary length that does not leave the component of the vertex, so a sufficiently long path must include a cycle. A bad event $A_v = \{\text{vertex } v \text{ has no neighbor in the same component}\}$. Thus

$$\begin{aligned}\Pr[A_v] &= \prod_{(u,v)\in E} \Pr[u \text{ and } v \text{ are in different components}]\\ &= \left(1 - \frac{1}{c}\right)^k < e^{-k/c} \leq e^{-3\ln k} = k^{-3}.\end{aligned}$$

A_v is determined by the component choices of itself and of its out neighbors $N^{\text{out}}(v)$ and these choices are independent. Thus by the Mutual Independence Principle, (Proposition 3) the dependency set of A_v consist of those u that share a neighor with v, i.e., those u for which $(\{v\} \cup N(v)) \cap (\{u\} \cup N(u)) \neq 0$. Thus the size of this dependency is at most $d = (k+1)^2$.

Take $d = (k+1)^2$ and $p = k^{-3}$, so $ep(d+1) = e(1 + (k+1)^2)/k^3 \leq 1$, holds for $k \geq 5$. Thus, noting that $k \geq 5$, by Lovasz Local Lemma, (Lemma 6),

$$\Pr\left[\bigcap_{v\in G} \overline{A}_v\right] > \left(1 - \frac{1}{d+1}\right)^n = \left(1 - \frac{1}{(k+1)^2+1}\right)^n > \left(1 - \frac{1}{k^2}\right)^n. \tag{7.1}$$

Graphs G_n of size n are encoded in strings of size $kn\lceil \log n \rceil$ and partitions are the proofs, encoded in strings of size $n\lceil \log k \rceil$. The verify V returns 1 if each partition contains a cycle. The verifier runs in time $O(n \log n)$. We define a sampling function $A \in \mathbf{FP}'$ over the partition/proofs that is the same as the probability used in the Lovasz Local Lemma, i.e. the uniform distribution. Thus $A(x) = x$. A samples $\text{Proofs}(G_n)$ with probability γ_n, where by Equation 7.1,

$$-\log \gamma_n < -n \log(1 - 1/k^2) < 2n/k^2.$$

Thus by Corolloray 5, there is a polynomial p, where for each graph $G_n \in Q$ of n vertices, there is a partition $x \in \text{Proofs}(G_n)$ with

$$\mathbf{K}^p(x) < 2n/k^2 + O(\log n).$$

□

7.2.2 Balancing-Vectors

Corollary 6 *Assume **Crypto**. For vector v, $\|v\|_\infty = \max_i |v_i|$. A binary matrix M has entries of 0s or 1s. Let $\{M_n\}$ be a uniformly polynomial time computable sequence of $n \times n$ binary matrices. There is a polynomial p where for each M_n there is a vector $b \in \{-1, 1\}^n$ such that $\|M_n b\|_\infty \leq 4\sqrt{n \ln n}$ and*

$$\mathbf{K}^p(b) = O(\log n).$$

Proof. Let $v = (v_1, \dots, v_n)$ be a row of M. Choose a random $b = (b_1, \dots, b_n) \in \{-1, +1\}^n$. Let $i_1, \dots, i_m$ be the indices such that $v_{i_j} = 1$. Thus

$$Y = \langle v, b \rangle = \sum_{i=1}^n v_i b_i = \sum_{j=1}^m v_{i_j} b_{i_j} = \sum_{j=1}^m b_{i_j}.$$

$$\mathbf{E}[Y] = \mathbf{E}[\langle v, b \rangle] = \mathbf{E}\left[\sum_i v_i b_i\right] = \sum_i \mathbf{E}[v_i b_i] = \sum v_i \mathbf{E}[b_i] = 0.$$

By the Chernoff inequality and the symmetry Y, for $\tau = 4\sqrt{n \ln n}$,

$$\Pr[|Y| \geq \tau] = 2\Pr[v \cdot b \geq \tau] = 2\Pr\left[\sum_{j=1}^m b_{i_j} \geq \tau\right] \leq 2\exp\left(-\frac{\tau^2}{2m}\right) = 2\exp\left(-8\frac{n \ln n}{m}\right) \leq 2n^{-8}.$$

Thus, the probability that any entry in Mb exceeds $4\sqrt{n \ln n}$ is smaller than $2n^{-8}$. Thus, with probability $1 - 2n^{-7}$, all the entries of Mb have value smaller than $4\sqrt{n \ln n}$.

Let $A(x) = x$ be the uniform sampling function. The verifier V takes in a matrix M and a vector b and returns 1 iff $\|Mb\|_\infty \leq 4\sqrt{n \ln n}$. Let $D \subset \{0, 1\}^n$ consist of all strings that encode vectors $b_x \in \{-1, +1\}^n$ in the natural way such that $\|Mb_x\|_\infty \leq 4\sqrt{n \ln n}$. By the above reasoning, A samples D with probability $\geq 1 - 2n^{-7} > 0.5$. So by Corollary 5, there is a polynomial p, where for each $n \times n$ matrix M_n there is a binary vector $b \in \{-1, 1\}^n$ with $\|Mb\|_\infty \leq 4\sqrt{n \ln n}$ and

$$\mathbf{K}^p(b) = O(\log n).$$

□

7.2.3 k-Sat

Corollary 7 *Assume **Crypto**. Let Φ_n be a $k(n)$-SAT formula, using n variables, $m(n)$ clauses, uniformly polynomial time computable in n. Furthermore, each variable occurs in at most $2^{k(n)}/k(n)e-1$ clauses. There is a polynomial p and a satisfying assignment x of Φ_n where*

$$\mathbf{K}^p(x) < 2m(n)e2^{-k(n)} + O(\log n).$$

Proof. The sample space is the set of all 2^n assigments. We choose a random assignment, where each variable is independently equally likely to have a true or false assignment. For each clause C_J, E_j is the bad event "C_j is not satisfied". Let $p = 2^{-k(n)}$ and $d = (2^{k(n)}/e) - 1$. Thus $\forall j$, $\Pr[E_j] \leq p$ as each clause has size $k(n)$ and each E_j is dependent on at most d other events since each variable appears in at most $2^{k(n)}/k(n)e - 1$ other clauses, and each clause has $k(n)$ variables. Thus since $ep(d+1) \leq 1$, by the Lovasz Local Lemma 6, we have that,

$$\Pr\left[\bigcap_j \overline{E_j}\right] > \left(1 - \frac{1}{d+1}\right)^{m(n)} = \left(1 - \frac{e}{2^{k(n)}}\right)^{m(n)}. \tag{7.2}$$

Let $D_n \subset \{0,1\}^n$ be the set of all assignments that satisfy ϕ_n. We use a uniform sampler, with $A(x) = x$. By the above reasoning, A samples D_n with probability $\gamma_n > \left(1 - \frac{e}{2^{k(n)}}\right)^{m(n)}$. Thus

$$-\log \gamma_n < -m(n) \log\left(1 - e/2^{k(n)}\right) < 2em(n)2^{-k(n)}.$$

By Corollary 5, there is a polynomial p, where for all n, there is a satisfying assignment $x \in D_n$ of $\Phi(n)$ with

$$\mathbf{K}^p(x) < 2m(n)e2^{-k(n)} + O(\log n).$$

□

Bibliography

[AF09] L. Antunes and L. Fortnow. Worst-Case Running Times for Average-Case Algorithms. In *2009 24th Annual IEEE Conference on Computational Complexity*, pages 298–303, 2009.

[AS04] N. Alon and J. Spencer. *The Probabilistic Method.* Wiley, New York, 2004.

[Blo70] B. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, page 422–426, 1970.

[CT91] T. Cover and J. Thomas. *Elements of Information Theory.* Wiley-Interscience, New York, NY, USA, 1991.

[EL] P. Erdos and L. Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. *Infinite and finite sets*, 10:609–627.

[Eps19] S. Epstein. On the algorithmic probability of sets. *CoRR*, abs/1907.04776, 2019.

[Eps21] Samuel Epstein. All sampling methods produce outliers. *IEEE Transactions on Information Theory*, 67(11):7568–7578, 2021.

[Eps22] S. Epstein. The outlier theorem revisited. *CoRR*, abs/2203.08733, 2022.

[ES91] P. Erdös and J. Spencer. Lopsided lovász local lemma and latin transversals. *Discret. Appl. Math.*, 30:151–154, 1991.

[Fei95] U Feige. A tight upper bound on the cover time for random walks on graphs. *Random Struct. Algorithms*, 6(1):51–54, 1995.

[HMR97] H. Hind, M. Molloy, and B. Reed. Colouring a graph frugally. *Combinatorica*, 17(4):469–482, 1997.

[Lev16] L. A. Levin. Occam bound on lowest complexity of elements. *Annals of Pure and Applied Logic*, 167(10):897–900, 2016.

[LOZ22] Z. Lu, I. Oliveira, and M. Zimand. Optimal coding theorems in time-bounded kolmogorov complexity. *CoRR*, abs/2204.08312, 2022.

[MR95] R Motwani and P Raghavan. *Randomized Algorithms.* Cambridge University Press, Cambridge; NY, 1995.

[MT10] R. Moser and G. Tardos. A Constructive Proof of the General LováSz Local Lemma. *J. ACM*, 57(2), 2010.

[MU05] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis.* Cambridge University Press, 2005.

[Nis94] Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

[VS17] Nikolay K. Vereshchagin and Alexander Shen. Algorithmic statistics: Forty years later. In *Computability and Complexity*, pages 669–737, 2017.

[VV10] N. Vereshchagin and P. Vitányi. Rate Distortion and Denoising of Individual Data using Kolmogorov Complexity. *IEEE Transactions on Information Theory*, 56, 2010.