22 Examples of Solution Compression via Derandomization

Samuel Epstein*

September 23, 2022

Abstract

We provide bounds on the compression size of the solutions to 22 problems in computer science. For each problem, we show that solutions exist with high probability, for some simple probability measure. Once this is proven, derandomization can be used to prove the existence of a simple solution.

1 Introduction

This paper showcases different bounds on the compression level of problems. An example problem is k-Sat, which is a conjunction of m clauses, each consisting of exactly k literals. A literal is either a variable v_i or the negation of a variable $\overline{v}_i$. Say there are n variables. The goal is to find an assignment of n variables that satisfy the conjunction. This is an NP-complete problem.

One of the goals in the field of algorithmic information theory, is to prove upper bounds on the smallest compression size of objects with combinatorial properties. In this paper we provide 22 examples of how to compress solutions to combinatorial problems.

For the k-Sat, for every variable size n, there are k-Sat instances which admit a single solution S. By permutating the variables and applying proper negations to this single solution $S \in \{0,1\}^n$, where $S[i] = 1$ iff variable x_i of the solution is *true*, S can be transformed so it cannot be compressed, with $n < \mathbf{K}(S) + O(1)$, where $\mathbf{K}$ is the prefix-free Kolmogorov complexity. However given more assumptions, and by using derandomization techinques, introduced in [Eps22], one can improve upon the bounds for compressing solutions to k-Sat. By Theorem 4 in Section 3.1,

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

The term $\mathbf{I}(\phi; \mathcal{H})$ is the amount of information that ϕ has with the halting sequence, defined in Section 2. Another area covered in this paper are examples of the compression size of players in interactive games. Take for example the Even-Odds game in Section 3.16. At every round, the player and environment each play a 0 or a 1. If the sum is odd the player gets a point. If the sum is even the player loses a point. The player can't see the environment's actions but the environment can see the player's actions. The environment can have its actions be a function of the player's previous actions. By using derandomization, in Section 3.16, we prove the following result.

Theorem. *Playing* Even-Odds *for large enough N rounds, for every environment $\mathbf{q}$ there is a deterministic agent $\mathbf{p}$ that can achieve a score of $\sqrt{N}$ and $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q}; \mathcal{H})$.*

*JP Theory Group. samepst@jptheorygroup.org

The bounds in this paper are achieved using derandomization. This process entails showing that there exists a simple probability measure over the solution-candidate space such that the solutions have a large enough probability. In the next step, Theorem 1 is used to show that there exists a simple solution based on this probability.

Contents

2 Tools

Let $\mathbb{N}$, $\mathbb{R}$, $\{0,1\}$, and $\{0,1\}^*$, consist of natural numbers, reals, bits, and finite sequences. $\mathbb{R}_{\geq 0}$ denotes nonnegative reals. $\|x\|$ is the length of a string. For mathematical statement A, $[A] = 1$ if A is true, otherwise $[A] = 0$. For $x \in \{0,1\}^*$ and $y \in \{0,1\}^*$, we use $x \sqsubseteq y$ if there is some string $z \in \{0,1\}^*$ where $xz = y$. We say $x \sqsubset y$ if $x \sqsubseteq y$ and $x \neq y$.

For positive real functions f the terms $<^+ f$, $>^+ f$, $=^+ f$ represent $<f+O(1)$, $>f-O(1)$, and $=f\pm O(1)$, respectively. The term $\stackrel{*}{=} f$ denotes $\stackrel{*}{<} f$ and $\stackrel{*}{>} f$. For the nonnegative real function f, the terms $<^{\log} f$, $>^{\log} f$, and $=^{\log} f$ represent the terms $<f+O(\log(f+1))$, $>f-O(\log(f+1))$, and $=f\pm O(\log(f+1))$, respectively.

The function $\mathbf{K}(x|y)$ is the conditional prefix Kolmogorov complexity. By the chain rule $\mathbf{K}(x,y) =^+ \mathbf{K}(x) + \mathbf{K}(y|x,\mathbf{K}(x))$. The universal probability of a string $x \in \{0,1\}^*$, conditional

on $y \in \{0,1\}^* \cup \{0,1\}^\infty$, is $\mathbf{m}(x|y) = \sum\{2^{-\|p\|} : U_y(p) = x\}$. Given any lower-computable semi-measure P over $\{0,1\}^*$, $D \subseteq \{0,1\}^*$, $O(1)\mathbf{m}(D) > 2^{-\mathbf{K}(P)}P(D)$. The coding theorem states $-\log \mathbf{m}(x|y) =^+ \mathbf{K}(x|y)$. The mutual information of a string x with the halting sequence $\mathcal{H} \in \{0,1\}^\infty$ is $\mathbf{I}(x;\mathcal{H}) = \mathbf{K}(x) - \mathbf{K}(x|\mathcal{H})$.

Lemma 1 (Symmetric 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, $\Pr\left[\bigcap_i \overline{E}_i\right] > \left(1 - \frac{1}{d+1}\right)^n$.*

Lemma 2 ([Eps22]) *For partial computable $f : \mathbb{N} \to \mathbb{N}$, for all $a \in \mathbb{N}$, $\mathbf{I}(f(a);\mathcal{H}) <^+ \mathbf{I}(a;\mathcal{H}) + \mathbf{K}(f)$.*

Proof.

$$\mathbf{I}(a;\mathcal{H}) = \mathbf{K}(a) - \mathbf{K}(a|\mathcal{H}) >^+ \mathbf{K}(a, f(a)) - \mathbf{K}(a, f(a)|\mathcal{H}) - \mathbf{K}(f).$$

The chain rule ($\mathbf{K}(x,y) =^+ \mathbf{K}(x) + \mathbf{K}(y|x, \mathbf{K}(x))$) applied twice results in

$$\begin{aligned}
\mathbf{I}(a;\mathcal{H}) + \mathbf{K}(f) &>^+ \mathbf{K}(f(a)) + \mathbf{K}(a|f(a), \mathbf{K}(f(a))) - (\mathbf{K}(f(a)|\mathcal{H}) + \mathbf{K}(a|f(a), \mathbf{K}(f(a)|\mathcal{H}), \mathcal{H}) \\
&=^+ \mathbf{I}(f(a);\mathcal{H}) + \mathbf{K}(a|f(a), \mathbf{K}(f(a))) - \mathbf{K}(a|f(a), \mathbf{K}(f(a)|\mathcal{H}), \mathcal{H}) \\
&=^+ \mathbf{I}(f(a);\mathcal{H}) + \mathbf{K}(a|f(a), \mathbf{K}(f(a))) - \mathbf{K}(a|f(a), \mathbf{K}(f(a)), \mathbf{K}(f(a)|\mathcal{H}), \mathcal{H}) \\
&>^+ \mathbf{I}(f(a);\mathcal{H}).
\end{aligned}$$

□

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

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 $\mathbf{Km}(G) \stackrel{\text{def}}{=} \min\{\|p\| : U(p) \in x \sqsupseteq y \in G\}$. 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 $\mathbf{Km}(G) <^{\log} -\log \mathbf{M}(G) + \mathbf{I}(G;\mathcal{H})$.*

Derandomization can also be applied to games (see [Hut05]), which consists of an interaction between an agent and an environment. In this paper we use two versions of the cybernetic agent model. 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 but the environment cannot be. The game can continue forever, given certain agents and environments. This is called a win/no-halt game.

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

Figure 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 $x_1 = \text{True}, x_2 = \text{True}, x_3 = \text{False}, x_4 = \text{False}, x_5 = \text{True}$.

Theorem 3 ([Eps22]) *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* $<^{\log} \mathbf{K}(\mathbf{p}') - \log p + \mathbf{I}((p, \mathbf{p}', \mathbf{q}); \mathcal{H})$ *that wins against* $\mathbf{q}$*.*

The second game is modified such that the environment gives a nonnegative 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 1 ([Eps22]) *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* $<^{\log} \mathbf{K}(\mathbf{p}) + \mathbf{I}((\mathbf{p}, n, \mathbf{q}); \mathcal{H})$ *that receives penalty* $< 2n$ *against* $\mathbf{q}$*.*

3 Examples

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

3.1 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 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 1.

Theorem 4 *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 Lovasz Local Lemma 1, we have that,

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

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 1, assuming $k \geq 3$,

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

Thus by Theorem 1 and Lemma 1, 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 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 5 uses the union bound whereas Theorem 6 uses the Lovasz Local Lemma.

Theorem 5 *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.

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

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 2, 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 Lovasz Local Lemma.

Theorem 6 *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 Lovasz 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 $\Pr\left[\bigcap_f \overline{E}_f\right] > 0$.

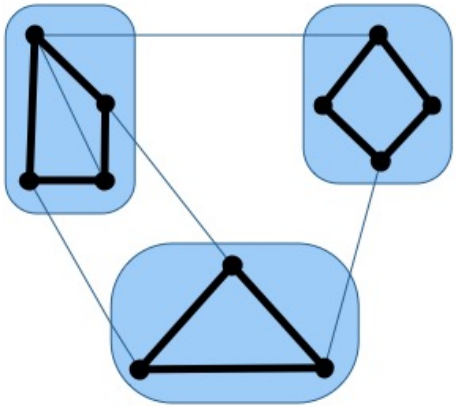

Figure 2: An example graph partitioned into 3 groups such that each group contains a cycle.

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 Lovasz Local Lemma 1,

$$\Pr\left[\bigcap_f \overline{E}_f\right] > \left(1 - \frac{1}{1+d}\right)^m = \left(1 - \frac{e}{2^{k-1}}\right)^m. \tag{3}$$

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, assuming $k \geq 4$

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

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

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

□

3.3 Vertex-Disjoint-Cycles

Proposition 1 (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 $\{A_{j_1}, \ldots, A_{j_k}\}$.*

This section deals with partitioning graphs into subgraphs such that each subgraph contains an independent cycle. An example partition can be seen in Figure 2.

Theorem 7 *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 $\mathbf{K}(\ell) <^{\log} \mathbf{K}(n,k) + 2n/k^2 + \mathbf{I}(G;\mathcal{H})$.*

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

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

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 1) 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 $k \geq 5$. One can trivially find a partition of a k-regular graph when $k < 5$ because $c = 1$. Thus, noting that $k \geq 5$,

$$\Pr\left[\bigcap_{v\in G} \overline{A}_v\right] > \left(1 - \frac{1}{d+1}\right)^n = \left(1 - \frac{1}{(k+1)^2+1}\right)^n > \left(1 - \frac{1}{k^2}\right)^n. \tag{4}$$

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

By Theorem 1 and Lemma 2, 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

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

□

3.4 Weakly-Frugal-Graph-Coloring

For an undirected graph $G = (V, E)$, a k-coloring assignment $f : V \to \{1, \ldots, k\}$ 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 8 *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 $\{u_1, \ldots, u_{\beta+1}\}$ that are in the neighborhood of a vertex $v \in V$, let $B_{u_1,\ldots,u_{\beta+1}}$ be the bad event that the vertices are the same color. $\Pr[B_{u_1,\ldots,u_{\beta+1}}] = p = 1/Q^\beta$. 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 $ep(d+1) \le 1$ of the Lovasz Local Lemma is fulfilled, because

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

By Lovasz Local Lemma 1,

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

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 1, we have a β-weakly frugal color assignment $f \in D$ of G using Q colors such that

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

□

3.5 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

Theorem 9 *For graph $G = (V,E)$, $|V| = n$ with max degree d, there is a k coloring f with $2d \le k$, and $\mathbf{K}(f) <^{\log} \mathbf{K}(n,k) + 2nd/k + \mathbf{I}((G,k);\mathcal{H})$.*

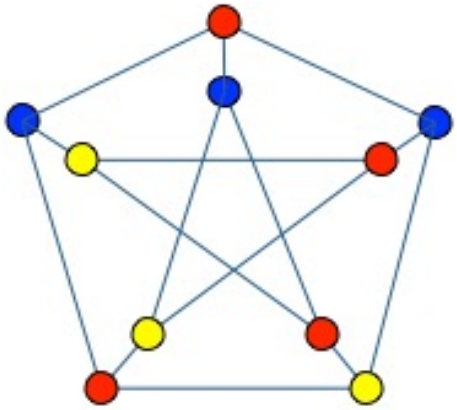

Figure 3: An example graph coloring. Nodes that share an edge are assigned different colors.

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 2, there is a coloring $f \in D$ with

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

□

3.6 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 10 *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 2,

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

□

3.7 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 11 *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 2,

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

□

3.8 Balancing-Vectors

For a vector $v = (v_1, \ldots, v_n) \in \mathbb{R}^n$, $\|v\|_\infty = \max_i |V_i|$. 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 12 *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 $v = (v_1, \ldots, v_n)$ be a row of M. Choose a random $b = (b_1, \ldots, b_n) \in \{-1,+1\}^n$. Let $i_1, \ldots, i_m$ 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 2, 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 13 *Let $v = v_1, \ldots, v_n \in \mathbb{R}^n$, all $|v_i| = 1$, Then there exist $\epsilon = \epsilon_1, \ldots, \epsilon_n = \pm 1$ 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 $\epsilon_1, \ldots, \epsilon_n$ be selected uniformly and independently from $\{-1,+1\}$. Set

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

Then

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

So

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

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 2, 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. □

3.9 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

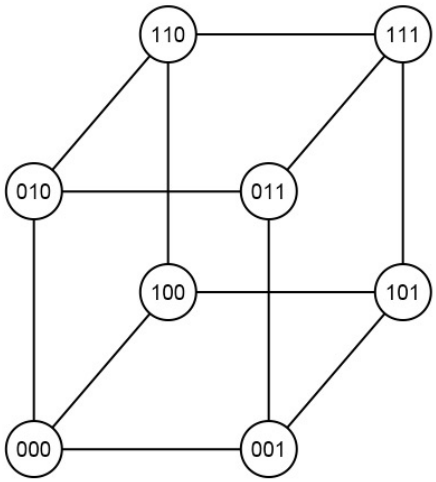

Figure 4: A Boolean Hypercube network, for $n = 3$.

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 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 $\sigma(i)$.
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 14 *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 2 results in

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

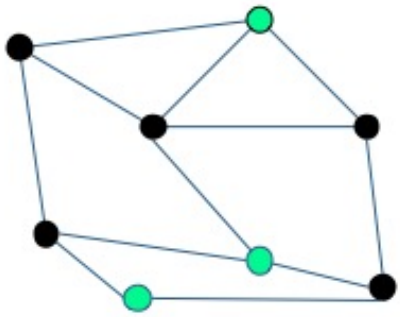

Figure 5: A graphical depiction of an independent set, represented by the green vertices. They do not share any edges.

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 $\mathbf{K}(\sigma) <^+ \mathbf{K}(y) <^{\log} \mathbf{I}((G,d);\mathcal{H})$. □

3.10 Independent-Set

An independent set in a graph G is a set of vertices with no edges between then, as shown in Figure 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 15 *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 2, there exists an $x \in D$, with

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

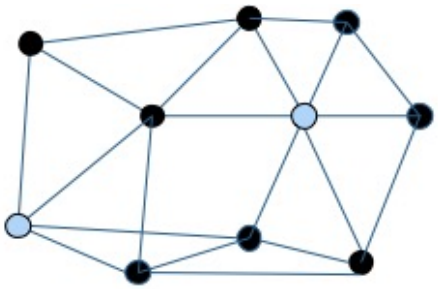

Figure 6: A graphical depiction of a dominating set. The two highlighted vertices are adjacent to all other vertices in the graph.

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 $\geq 0.75\sqrt{n} - 2m/n$. □

3.11 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 6.

Theorem 16 *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 $|X_x| \leq 3n\ln(\delta+1)/(\delta+1)]$ 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 2, 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

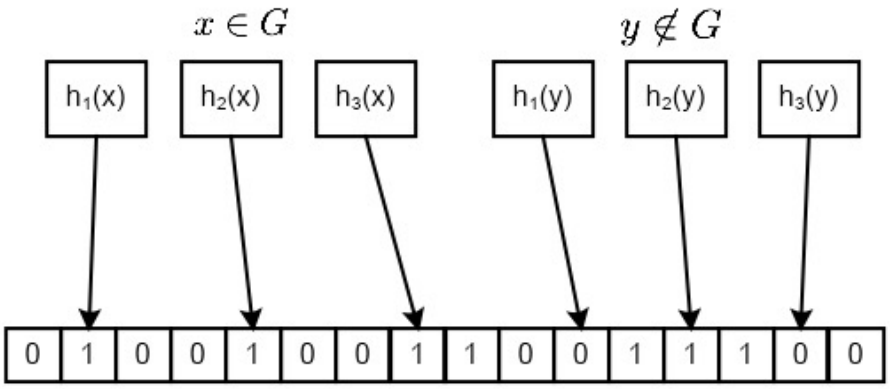

Figure 7: A graphical depiction of Bloom filter with $k = 3$ hash functions. The first element x is in G and is thus mapped to ones in the Bloom filter. Thus the Bloom filter would indicate that $x \in G$. The second element y is not in G and has some of the hash functions map to 0. Thus the Bloom filter would indicate that $y \notin G$.

dominating set x' of G exists of size $\leq 3n\frac{1+\ln(\delta+1)}{\delta+1}$ such that

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

□

3.12 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 17 *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 $h_i : \{0,1\}^\ell \to \{1,\ldots,n\}$, 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 $i \in \{1,\ldots,k\}$, $v[h_i(x)]$ 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 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],

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

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

$$\Pr(X/n \geq p'9/10) \leq 2e\sqrt{n}e^{-np'/300}. \tag{5}$$

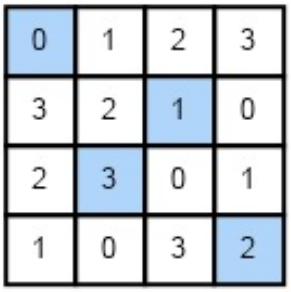

Figure 8: A graphical depiction of a Latin Transversal. Each number appears exactly 4 times in the matrix. The transversal is a permutation of the matrix such that all its entries have different values.

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 $i \in \{1, \ldots, k\}$ 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 5.

Let $F' \subset \{0,1\}^*$ consist of all encodings of k hash functions $h_i : \{0,1\}^\ell \to \{1, \ldots, n\}$. 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 2, 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 $\text{Error}(f, G) \leq 0.878^{n/m}$. Furthermore,

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

□

3.13 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 8

Lemma 3 (Lopsided Lovasz 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 18 *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 Lovasz Local Lemma, 3, the desired bounds can be achieved if we can show that

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

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 $\Pr\left(A_{1122} \big| \bigcap_S \overline{A}_{pqp'q'}\right) \leq 1/n(n-1)$.

The number of bad events $A_{iji'j'}$ is $\binom{n^2}{k}\binom{k}{2}$, as there are n^2/k distinct numbers, and each

number appears k times. Thus by the Lopsided Lovasz Local Lemma 3, for $n \geq 3$,

$$\begin{aligned}
\Pr\left(\bigcap_i \overline{A}_{iji'j'}\right) &\geq (1-2/n(n-1))^{\binom{\frac{n^2}{k}}{\binom{k}{2}}} \\
-\log \Pr\left(\bigcap_i \overline{A}_{iji'j'}\right) &< \binom{\frac{n^2}{k}}{\binom{k}{2}} \log(1-2/n(n-1)) \\
&< 2\left(\frac{2n^2}{kn(n-1)}\right)\binom{k}{2} \\
&< \binom{\frac{8}{k}}{\binom{k}{2}} \\
&\leq 4(k-1). \qquad (6)
\end{aligned}$$

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 6, $-\log P(D) < 4(k-1)$. Thus by Theorem 1 and Lemma 2, for $n \geq 3$, there exists a permutation $\tau \in D$ that is a Latin Transversal and has complexity

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

□

3.14 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 $i = 1, \ldots, n$. We define a computable probability $P : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ where $P(\langle a_1\rangle\langle a_2\rangle \ldots \langle a_n\rangle) = \prod_{i=1}^n p(a_i)$. $\mathbf{K}(P) <^+ \mathbf{K}(p,n)$. Let D' be a (potentially infinite) set of strings where $x \in D$ iff $x = \langle a_1\rangle\langle a_2\rangle \ldots \langle a_n\rangle$ 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 2, 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 $\mathbf{K}(f_1|\mathcal{H}) <^{+} \mathbf{K}(m)$.

3.15 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 19 *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 2, 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 $|T| = 2^{n-m}$.

□

3.16 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 20 *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 3, there exists a deterministic agent $\mathbf{p}$ that can beat $\mathbf{q}$ with complexity

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

□

3.17 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.

There are t_G rounds and the agent starts at $s \in V$. At round 1, the environment gives the agent the degree $s \in V$, $\mathrm{Deg}(s)$. The agent picks an number between 1 and $\mathrm{Deg}(s)$ 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 current vertex, round number, and 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.

Theorem 21 *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 (see [Lov96]), 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 $v \in V$. 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.

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 $\stackrel{*}{>} 1/|E|$. Thus by Theorem 3, there is a deterministic agent $\mathbf{p}$ that can find r in t_G turns and has complexity $\mathbf{K}(\mathbf{p}') <^{\log} \log |E| + \mathbf{I}((G, s, t); \mathcal{H})$.

3.18 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, 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.

Theorem 22 *There is a deterministic agent* $\mathbf{p}$ *that can receive a penalty* $< 2N$ *and has complexity* $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{I}(\mathbf{q}; \mathcal{H})$.

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 Corollary 1, there is a deterministic agent $\mathbf{p}$ such that

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

□

Let $\mathbf{q}$ be defined so that $P_i(a) = [a \leq 2^i] 2^{-i}$ and $T_i = [a \leq 2^i] 2^{i - \mathbf{K}(a|i)}$. Thus each T_i is a randomness deficiency function. 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.

3.19 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 $\mathbf{q}$ 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 $\mathbf{q}$. 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, ℓ, is a computable function of the current vertex, round number, and 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 23 *There is a deterministic agent* $\mathbf{p}$ *that can play against* Cover-Time *instance* (G, S, ℓ), $|G| = n$, *and achieve penalty* $\frac{4}{27}n^3 + o(n^3)$ *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 Corollary 1 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

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

□

3.20 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 $\mathbf{q}$ 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 a function of the round number and 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 9.

Theorem 24 *There is a deterministic agent* $\mathbf{p}$ *that can play against* Cover-Time *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 $\mathbf{q}$ 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 3, there exist a deterministic agent $\mathbf{p}$ and c where $\mathbf{p}$ can beat $\mathbf{q}$ and has complexity $\mathbf{K}(\mathbf{p}) <^{\log} \mathbf{K}(p') - \log c/n^2 + \mathbf{I}(\mathbf{q}; \mathcal{H}) <^{\log} 2\log n + \mathbf{I}((G, \ell); \mathcal{H})$. □

3.21 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

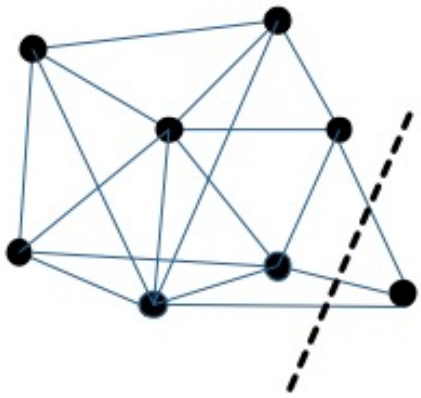

Figure 9: A graphical depiction of a minimum cut. By removing the edges along the dotted line, two components are created.

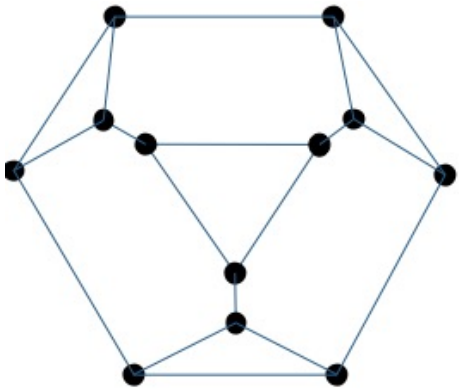

Figure 10: An example vertex-transitive graph. For every

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 10. 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 round number and the agents previous actions. The agent wins if after $2k$ rounds, the agent is back at u.

Theorem 25 *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 3, there exists a deterministic agent

p that can beat **q** with complexity

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

□

3.22 Classification

In machine learning, Classification is the task of learning a binary function c from $\mathbb{N}$ to bits $\{0,1\}$. The learner is given a sample consisting of pairs (x, b) for string x and bit b and outputs a binary classifier $h : \mathbb{N} \to \{0,1\}$ that should match c as much as possible. Occam's razor says that "the simplest explanation is usually the best one." Simple hypothesis are resilient against overfitting to the sample data. The question is, given a particular problem in machine learning, how simple can the hypotheses be?

We use a probabilistic model. The target concept is modeled by a random variable $\mathcal{X}$ with distribution p over ordered lists of natural numbers. The random variable $\mathcal{Y}$ models the labels, and has a distribution over lists of bits, where the distribution of $\mathcal{X} \times \mathcal{Y}$ is $p(x, y)$ with conditional probability requirement $p(y|x) = \prod_{i=1..|x|} p(y_i|x_i)$. Each such (x_i, y_i) is a labeled sample. A binary classifier f is consistent with labelled samples (x, y), if for all i, $f(x_i) = y_i$. Let $\Gamma(x, y)$ be the minimum Kolmogorov complexity of a classifier consistent with (x, y). $\mathcal{H}(\mathcal{Y}|\mathcal{X})$ is the conditional entropy of $\mathcal{Y}$ given $\mathcal{X}$.

Theorem 26

1. $\mathcal{H}(\mathcal{Y}|\mathcal{X}) \leq \mathbf{E}[\Gamma(\mathcal{X}, \mathcal{Y})] <^{\log} \mathcal{H}(\mathcal{Y}|\mathcal{X}) + \mathbf{K}(p)$.
2. *For each $c, b \in \mathbb{N}$, there exists random labeled samples $\mathcal{X} \times \mathcal{Y}$ with distribution p, such that, up to precision $O(\log cb)$, $\mathbf{E}[\Gamma(\mathcal{X}, \mathcal{Y})] = b + c$, $\mathcal{H}(\mathcal{Y}|\mathcal{X}) = b$, and $\mathbf{K}(p) = c$.*

Proof. We start with the lower bound of part 1. $\mathbf{E}[\Gamma(\mathcal{X}, \mathcal{Y})] = \sum_x p(x) \sum_y p(y|x)\Gamma(x, y)$. Each $\Gamma(x, y)$ represents a self-delimiting program to compute a classifier f such that $f(x_i) = y_i$. Thus if $y \neq y'$, $\Gamma(x, y)$ and $\Gamma(x, y')$ represents two programs v and v' such that $v \not\sqsubseteq v'$ and $v' \not\sqsubseteq v$. Thus for a fixed x, ranged over y, $\Gamma(x, y)$ represents the length of a self-delimiting code. Due to properties of conditional entropy, which is minimal over all self-delimiting codes,

$$\mathbf{E}[\Gamma(\mathcal{X}, \mathcal{Y})] = \sum_x p(x) \sum_y p(y|x)\Gamma(x, y) \geq \sum_x p(x) \sum_y p(y|x)(-\log p(y|x)) = \mathcal{H}(\mathcal{Y}|\mathcal{X}).$$

We now prove the upper bound of part 2. To do so, we need the following lemma. The following lemma is perhaps surprising because it shows that the $\mathbf{I}(\cdot; \mathcal{H})$ terms in inequalities can be removed by averaging over a computable probability.

Lemma 4 *For computable probability p, $\sum_x p(x)\mathbf{I}(x; \mathcal{H}) <^{+} \mathbf{K}(p)$.*

Proof. This follows from Theorem 3.1.3 in [Gác21], and we will reproduce its arguments. Since $\mathbf{K}(x/\mathcal{H})$ is the length of a self delimiting code,

$$\sum_x p(x)\mathbf{K}(x/\mathcal{H}) \geq \mathcal{H}(p),$$

where $\mathcal{H}(p)$ is the entropy of p. Furthermore, for all $x \in \{0,1\}^*$, $\mathbf{K}(x) <^+ -\log p(x) + \mathbf{K}(p)$. Therefore

$$\sum_x p(x)\mathbf{K}(x) <^+ \sum_x p(x)(-\log p(x)) + \mathbf{K}(p) <^+ \mathcal{H}(p) + \mathbf{K}(p).$$

So

$$\sum_x p(x)\mathbf{I}(x;\mathcal{H}) = \sum_x p(x)\left(\mathbf{K}(x) - \mathbf{K}(x/\mathcal{H})\right) <^+ \mathcal{H}(p) + \mathbf{K}(p) - \sum_x p(x)\mathbf{K}(x/\mathcal{H}) <^+ \mathbf{K}(p).$$

□

Binary classifiers are identified by infinite sequences $\alpha \in \{0,1\}^\infty$. We define the computable measure $S : \{0,1\}^* \to \mathbb{R}_{\geq 0}$ over $\{0,1\}^\infty$, where $S(x) = \prod_{n=1..|x|} p(x_n|n)$, where $\mathbf{K}(S|p) = O(1)$. Let $\{(x_i, y_i)\}$ be a set of labelled samples and we define clopen set $C_{x,y} = \{\alpha : \alpha \in \{0,1\}^\infty, \alpha[x_i] = y_i\}$. Then $S(C_{x,y}) = p(y|x)$. By Theorem 2, relativized to p,

$$\begin{aligned}\min_{\alpha \in C_{x,y}} \mathbf{K}(\alpha|p) &<^{\log} \mathbf{K}(S|p) - \log S(C_{x,y}) + \mathbf{I}(C_{x,y};\mathcal{H}|p)\\ &<^{\log} -\log S(C_{x,y}) + \mathbf{I}(C_{x,y};\mathcal{H}|p)\\ &<^{\log} -\log p(y|x) + \mathbf{I}(C_{x,y};\mathcal{H}|p)\end{aligned}$$

Averaging over all x and y using probability p, one gets

$$\sum_{x,y} p(x,y) \min_{\alpha \in C_{x,y}} \mathbf{K}(\alpha|p) <^{\log} \sum_{x,y} p(x,y)(-\log p(y|x)) + \sum_{x,y} p(x,y)\mathbf{I}(C_{x,y};\mathcal{H}|p). \tag{7}$$

Applying Lemma 4 relative to p, we get

$$\sum_{x,y} p(x,y)\mathbf{I}(C_{x,y};\mathcal{H}|p) = \sum_{x,y} p(C_{x,y})\mathbf{I}(C_{x,y};\mathcal{H}|p) <^+ \mathbf{K}(p|p) = O(1). \tag{8}$$

Combining equations 7 and 8,

$$\begin{aligned}\sum_{x,y} p(x,y) \min_{\alpha \in C_{x,y}} \mathbf{K}(\alpha|p) &<^{\log} \sum_{x,y} p(x,y)(-\log p(y|x))\\ \left(\sum_{x,y} p(x,y) \min_{\alpha \in C_{x,y}} \mathbf{K}(\alpha)\right) - \mathbf{K}(p) &<^{\log} \sum_{x,y} p(x,y)(-\log p(y|x))\\ \mathbf{E}[\Gamma(\mathcal{X},\mathcal{Y})] &<^{\log} \mathcal{H}(\mathcal{Y}|\mathcal{X}) + \mathbf{K}(p).\end{aligned}$$

We now prove part 2. We ignore all $O(\log cd)$ terms. So equality $=$ is equivalent to $= \pm O(\log cd)$. We define a probability $p(x,y)$ over the first $n = 2c + 2b + 2$ numbers and corresponding bits. Thus we can describe p as a probability measure over strings of size n, making sure to maintain p's conditional probability restriction described earlier.

Let $z \in \{0,1\}^c$ be a random string of size c, with $c <^+ \mathbf{K}(z)$. For all strings $w \in \{0,1\}^b$ of size b, $p(\langle z\rangle\langle w\rangle) = 2^{-b}$, with $\|\langle z\rangle\langle w\rangle\| = n$. $\mathcal{H}(\mathcal{Y}|\mathcal{X}) = -\sum_{w\in\{0,1\}^b} 2^{-b}(\log p(\langle z\rangle\langle w\rangle)) = -\sum_{w\in\{0,1\}^b} 2^{-b}(\log 2^{-b}) = b$. Furthermore $\mathbf{K}(p) = c$. The infinite sequence $\alpha = \langle z\rangle\langle w\rangle 0^\infty$ realizes $\Gamma(\langle z\rangle\langle w\rangle)$ up to an additive constant for each $w \in \{0,1\}^b$. Thus $\mathbf{K}(\alpha) = \mathbf{K}(z,w)$.

$$\mathbf{E}[\Gamma(\mathcal{X},\mathcal{Y})] = 2^{-b} \sum_{w\in\{0,1\}^b} \mathbf{K}(\langle z\rangle\langle w\rangle) = \mathbf{K}(z) + 2^{-b} \sum_{w\in\{0,1\}^b} \mathbf{K}(w/z, \mathbf{K}(z)).$$

Using Theorem 3.1.3 in [Ǵ21] conditioned on $\langle z, \mathbf{K}(z)\rangle$, we get that $\sum_{w\in\{0,1\}^b} 2^{-b}\mathbf{K}(w/z,\mathbf{K}(z)) = \mathcal{H}(\mathcal{U}_b) \pm \mathbf{K}(b/z,\mathbf{K}(z)) = b$, where $\mathcal{U}_b$ is the uniform measure over strings of size b. So $\mathbf{E}[\Gamma(\mathcal{X},\mathcal{Y})] = \mathbf{K}(z) + b = b + c$. □

References

[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.

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

[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.

[Gác21] Peter Gács. Lecture notes on descriptional complexity and randomness. *CoRR*, abs/2105.04704, 2021.

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

[Hut05] Ml Hutter. *Universal Artificial Intelligence*. Texts in Theoretical Computer Science. An EATCS Series. Springer, Berlin and Heidelberg, 2005.

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

[Lov96] L. Lovász. Random walks on graphs: A survey. In D. Miklós, V. T. Sós, and T. Szőnyi, editors, *Combinatorics, Paul Erdős is Eighty*, volume 2, pages 353–398. János Bolyai Mathematical Society, 1996.

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

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