Codes

Jonathan L.F. King
University of Florida, Gainesville FL 32611-2082, USA
squash@ufl.edu
Webpage http://squash.1gainesville.com/

10 May, 2023 (at *11:10*)

(Folks, help me proofread and correct this.)

§An Overview

Formal languages

Words. Use $\varnothing$ for the empty set. An ***alphabet*** $\mathbf{G}$ is a non-empty set, whose members are called ***letters***; usually $2 \le |\mathbf{G}| < \infty$. A ***word*** (over an alphabet $\mathbf{G}$) is a finite string of letters; Use $\mathbf{G}^\star$ for the set of all words, and use $\boldsymbol{\varepsilon}$ for the ***nullword*** $\boldsymbol{\varepsilon}$, the unique length-zero word. E.g if $\mathbf{G} = \{\mathtt{a}, \mathtt{b}\}$, then $\mathbf{G}^\star$ equals

$$\{\boldsymbol{\varepsilon}, \mathtt{a}, \mathtt{b}, \mathtt{aa}, \mathtt{ab}, \mathtt{ba}, \mathtt{bb}, \mathtt{aaa}, \mathtt{aab}, \ldots\}.$$

Write $\mathbf{G}^+$ for $\mathbf{G}^\star \smallsetminus \{\boldsymbol{\varepsilon}\}$.

Concatenation of words $\mathbf{v},\mathbf{z} \in \mathbf{G}^\star$ is written $\mathbf{v} \triangleright \mathbf{z}$ or just $\mathbf{vz}$. Thus $\mathtt{cat} \triangleright \mathtt{nip} = \mathtt{catnip}$. *So $\mathbf{G}^\star$ is a semigroup under concatenation, with $\boldsymbol{\varepsilon}$ the identity element.* Use $\mathrm{Len}(\mathbf{v})$ or $|\mathbf{v}|$ for the ***length*** of word $\mathbf{v}$, and have $\mathbf{v} \overset{|\cdot|}{>} 3$ mean that $|\mathbf{v}| > 3$. For n a natnum, let $\mathbf{v}^n$ mean the concatenation $\mathbf{v}\mathbf{v} \overset{n}{\ldots} \mathbf{v}$. So $\mathbf{v}^0 = \boldsymbol{\varepsilon}$.

Languages. A "***language*** *over* alphabet $\mathbf{G}$" is a subset $\mathcal{L} \subset \mathbf{G}^\star$. Here are six distinct languages over alphabet $\{\mathtt{a}, \mathtt{b}, \ldots, \mathtt{z}\}$:

$$\varnothing = \{\}, \{\boldsymbol{\varepsilon}\}, \{\mathtt{catnip}\}, \{\mathtt{cat}, \mathtt{nip}\}, \{\boldsymbol{\varepsilon}, \mathtt{cat}, \mathtt{nip}\}$$
$$\underline{\text{and}} \quad \{\mathtt{bc}, \mathtt{bac}, \mathtt{baac}, \mathtt{baaac}, \ldots\} = \{\mathtt{ba}^n\mathtt{c}\}_{n=0}^\infty.$$

The first five are finite languages, having cardinalities $0, 1, 1, 2, 3$. Call $\varnothing$ the ***void language*** and call $\{\boldsymbol{\varepsilon}\}$ the ***nullword language***. The "concatenation of *languages*" $\mathcal{K}, \mathcal{L} \subset \mathbf{G}^\star$ is

$$\mathcal{K}\mathcal{L} = \mathcal{K} \triangleright \mathcal{L} \quad := \quad \{\mathbf{v} \triangleright \mathbf{w} \mid \mathbf{v} \in \mathcal{K} \text{ and } \mathbf{w} \in \mathcal{L}\}.$$

[So $\varnothing \triangleright \mathcal{L} = \varnothing = \mathcal{L} \triangleright \varnothing$ and $\{\boldsymbol{\varepsilon}\} \triangleright \mathcal{L} = \mathcal{L} = \mathcal{L} \triangleright \{\boldsymbol{\varepsilon}\}$.] Let $\mathcal{L}^n$ mean $\mathcal{L}\mathcal{L} \overset{n}{\ldots} \mathcal{L}$. Hence $\mathcal{L}^0 = \{\boldsymbol{\varepsilon}\}$, since the nullword language is the identity element for language-concatenation.[♡1]

For languages $\mathcal{K} \subset \mathcal{L}$, language $\mathcal{L}$ is an ***extension*** of $\mathcal{K}$, and $\mathcal{K}$ is a ***restriction*** of $\mathcal{L}$.

[♡1] Aside: We already knew that $(\mathbf{G}^\star, \triangleright, \boldsymbol{\varepsilon})$ is a [non-commutative] semigroup. And letting $\mathbb{L} = \mathbb{L}_\mathbf{G}$ denote the set of *all* languages over $\mathbf{G}$, this generalizes to a [non-commutative] semigroup

$$(\mathbb{L}, \triangleright, \{\boldsymbol{\varepsilon}\}).$$

Of course, we also have the two commutative semigroups $(\mathbb{L}, \cup, \varnothing)$ and $(\mathbb{L}, \cap, \mathbf{G}^\star)$.

Star. Define the ***Kleene star*** operator by

$$\mathcal{L}^\star \quad := \quad \bigcup_{n=0}^\infty \mathcal{L}^n.$$

[*Language $\mathcal{L}^\star$ is the minimal extension of $\mathcal{L}$ which is sealed (closed) under finite concatenation of words. Since $\mathcal{L}^\star$ contains the concatenation of zero-many words, $\mathcal{L}^\star$ owns $\boldsymbol{\varepsilon}$.*] In particular $\varnothing^\star = \{\boldsymbol{\varepsilon}\} = \{\boldsymbol{\varepsilon}\}^\star$. Similarly, the ***Kleene plus*** operator is

$$\mathcal{L}^+ \quad := \quad \bigcup_{n=1}^\infty \mathcal{L}^n.$$

Hence $[\boldsymbol{\varepsilon} \in \mathcal{L}^+] \Leftrightarrow [\boldsymbol{\varepsilon} \in \mathcal{L}] \Leftrightarrow [\mathcal{L}^+ = \mathcal{L}^\star]$. Each Kleene op is ***idempotent***: $[\mathcal{L}^\star]^\star = \mathcal{L}^\star$ and $[\mathcal{L}^+]^+ = \mathcal{L}^+$.

Prefix/Suffix. For words, say "$\mathbf{v}$ is a ***prefix*** of $\mathbf{w}$" if there exists a word $\mathbf{z}$ with $\mathbf{vz} = \mathbf{w}$; write $\mathbf{v} \preccurlyeq \mathbf{w}$ for this relation. If, also, $\mathbf{v} \neq \mathbf{w}$, then $\mathbf{v}$ is a ***proper prefix*** of $\mathbf{w}$, written $\mathbf{v} \prec \mathbf{w}$.

If $\exists \mathbf{z} \in \mathbf{G}^\star$ with $\mathbf{zv} = \mathbf{w}$, then "$\mathbf{v}$ is a ***suffix*** of $\mathbf{w}$". [However, we have no special symbol for the relation.]

Codes

For the time being, a ***code*** $\mathcal{C}$ means a non-void subset $\mathcal{C} \subset \mathbf{G}^+$; usually $2 \le |\mathcal{C}| < \infty$. [Occasionally it is convenient to consider collections $\mathcal{C}$ which *might* own $\boldsymbol{\varepsilon}$. So if all we know is that $\mathcal{C} \subset \mathbf{G}^\star$, then we call $\mathcal{C}$ a ***nullishcode***. If we can later on prove that $\mathcal{C} \not\ni \boldsymbol{\varepsilon}$, then we'll have shown $\mathcal{C}$ to be a code.]

Call $\mathcal{C}$ a ***block code*** if all its codewords have the same length. E.g, $\{\mathtt{FBI}, \mathtt{CIA}\}$ is a blockcode, whereas $\{\mathtt{Go}, \mathtt{Gators}\}$ is *not* a blockcode, –although it *is* (see below) a prefixcode. [Caveat: "block code" is used with slightly different meanings in the literature. Perhaps ***constant-length code*** is a more accurate term.]

A code $\mathcal{C}$ is ***uniquely decodable*** (a ***UD-code***) if each code-message $\mathbf{z} \in \mathcal{C}^\star$ has a unique decomposition w.r.t $\mathcal{C}$. That is, if words $\mathbf{v}_j, \mathbf{w}_k \in \mathcal{C}$ satisfy

1.1: *If* $\mathbf{v}_1\mathbf{v}_2\ldots\mathbf{v}_J = \mathbf{z} = \mathbf{w}_1\mathbf{w}_2\ldots\mathbf{w}_K$ *then* $J = K$ *and* $\forall i$: $\mathbf{v}_i = \mathbf{w}_i$.

A ***prefix code*** $\mathcal{C}$ (more accurately called a "prefix-free code") has no codeword being a proper prefix of another. Prefix-codes are UD-codes since, stronger than (1.1), they have the ***RI-UD property*** (the "right-infinite-UD property") that

1.2: $\begin{bmatrix}\mathbf{v}_1\mathbf{v}_2\mathbf{v}_3\cdots = \mathbf{w}_1\mathbf{w}_2\mathbf{w}_3\cdots \\ \text{with each } \mathbf{v}_j,\mathbf{w}_k \in \mathcal{C}\end{bmatrix} \Rightarrow \begin{bmatrix}\forall i\in\mathbb{Z}_+: \\ \mathbf{v}_i = \mathbf{w}_i\end{bmatrix}.$

We have these non-reversible implications

1.2′: $\text{Block} \implies \text{Prefixcode} \overset{*1}{\implies} \text{RI-UD} \overset{*2}{\implies} \text{UD}.$

A code showing (∗2) non-reversible has these words

1.2″: $\mathbf{v} := \mathtt{b},\ \mathbf{w} := \mathtt{ba},\ \mathbf{z} := \mathtt{aa}.$

It is uniquely decodable (Exer. E1), yet fails (1.2), since $\mathbf{vzzz}\cdots$ equals $\mathbf{wzzz}\cdots$. Finally, that (∗1) is non-reversible will be shown by (1.5), the "Chris code".

A ***suffix code*** (no codeword is a proper suffix of another) is automatically a UD-code. Dually to (1.2′) we have non-reversible implications

1.3′: $\text{Block} \implies \text{Suffixcode} \implies \text{LI-UD} \implies \text{UD},$

where a ***left-infinite-UD–code*** (a ***LI-UD–code***) satisfies

1.3: $\begin{bmatrix}\cdots\mathbf{v}_{-2}\mathbf{v}_{-1} = \cdots\mathbf{w}_{-2}\mathbf{w}_{-1} \\ \text{with each } \mathbf{v}_j,\mathbf{w}_k \in \mathcal{C}\end{bmatrix} \Rightarrow \begin{bmatrix}\forall i\in\mathbb{Z}_-: \\ \mathbf{v}_i = \mathbf{w}_i\end{bmatrix}.$

Note (1.2″) is an example of a suffixcode which is not a prefixcode.

Bi-infinite. A bi-∞ **G**-string σ can be viewed as a map $\sigma:\mathbb{Z}\to\mathbf{G}$. A ***$\mathcal{C}$-parsing*** of σ is a sequence

$$\cdots < k_{-2} < k_{-1} < k_0 < k_1 < k_2 < k_3 < \cdots$$

of integers st. each substring $\sigma\rfloor_{[k_\ell\,..\,k_{\ell+1})}$ is a codeword, that is, lies $\mathcal{C}$. Write sequence $(k_\ell)_{\ell\in\mathbb{Z}}$ as $\vec{\mathbf{k}}$.

Say that $\mathcal{C}$ has the ***bi-infinite-UD property*** (is ***BI-UD***) if

1.4: *Each bi-∞ string σ which has a $\mathcal{C}$-parsing, has only one $\mathcal{C}$-parsing. I.e, with $\vec{\jmath}$ and $\vec{\mathbf{k}}$ two $\mathcal{C}$-parsings of σ, then the sets $\{j_i\}_{i\in\mathbb{Z}}$ and $\{k_\ell\}_{\ell\in\mathbb{Z}}$ are equal.*

Slightly weaker, consider two parsings $\vec{\jmath}$ and $\vec{\mathbf{k}}$, and let $\mathbf{v}_\ell := \sigma\rfloor_{[j_\ell\,..\,j_{\ell+1})}$ and $\mathbf{w}_\ell := \sigma\rfloor_{[k_\ell\,..\,k_{\ell+1})}$. The ***weak-BI-UD*** property asserts

For each σ and parsings as above, there exists a translation $T\in\mathbb{Z}$ so that:

1.4$^{\text{weak}}$: $\forall\ell\in\mathbb{Z}:\quad \mathbf{v}_{\ell+T} = \mathbf{w}_\ell.$

(I.e, one parsing may be a shift of the other, but the codeword sequences are the same.)

Immediately,

1.4′: $\text{BI-UD} \overset{*3}{\implies} \text{weak-BI-UD} \overset{*4}{\implies} \begin{bmatrix}\text{Both LI-UD} \\ \text{and RI-UD}\end{bmatrix}.$

The code {bbb} produces $\sigma := \cdots\mathtt{bbbb}\cdots$, which is its only bi-∞ string. This σ has *three* parsings, since the cutpoints $\vec{\jmath}$ can all be mod-3 congruent to −1 or 0 or 1. Yet each parsing yields the *same* codeword sequence, namely $\cdots$ [bbb][bbb][bbb] $\cdots$. Hence (∗3) is *not reversible.*

The "Pirate code" {OH, HO} is trivially LI-UD and RI-UD, since it is a blockcode. Yet the Pirate code admits bi-∞ string ⋯HOHOHOH⋯, which can be parsed as ⋯[OH][OH][OH]⋯ or as ⋯[HO][HO][HO]⋯, two different codeword sequences. Yup; (∗4) *ain't reversible either.*

The "Chris code" (evidently a cry for help)

1.5: $$\{\mathtt{S}, \mathtt{SOS}\}$$

is BI-UD, since each occurrence of "O" must lie in [SOS], and every other codeword must be [S]. Not being a prefixcode, (1.5) proves (∗1) not reversible. [So (1.5) is neither a prefix nor suffix code, yet is UD.]

Trees. Here, a (rooted) ***tree*** is a set T of nodes, equipped with two operators: Root(T) is the root-node of T. For each node $v\in T$, let Kids(v) be the *set* of children of v. A node w is a ***leaf-node*** if: The set Kids(w) is empty. A tree has the property that, from the root-node, one can get to an arbitrary node, by applying the Kids(·) operator finitely-many times.

Trees T and S are (***tree-***)***isomorphic*** if *there exists* a bijection $f:T\to S$ such that:

TI 1: $f(\text{Root}(T)) = \text{Root}(S)$.

TI 2: For each $v\in T$:
$$\{f(k) \mid k\in\text{Kids}(v)\} = \text{Kids}(f(v)).$$

For a $\Gamma\in\mathbb{Z}_+$, a tree is ***Γ-bounded*** if each node has at most Γ many children. The tree is ***Γ-full*** if every node is either has *no* children [is a ***leaf-node***], or has precisely Γ many children; otherwise, the tree is ***Γ-deficient***.

Inequalities

Kraft proved (2a) for *prefix-codes*, as well as its converse, (2b). McMillan strengthened (2a) to UD-codes.

2: Kraft-McMillan Inequality. *Consider a countable code $\mathcal{C}$ over finite alphabet* **G**. *If $\mathcal{C}$ is a UD-code then*

2a: $$\sum_{\mathbf{v}\in\mathcal{C}} 1/\,\Gamma^{\mathrm{Len}(\mathbf{v})} \;\leq\; 1\,,$$

where Γ is the number of letters in **G**.
Conversely, consider posints $\vec{\boldsymbol{\ell}} = (\ell_1, \ell_2, \ldots, \ell_R)$.

2b: *If $\sum_{j=1}^{R} 1/\,\Gamma^{\ell_j} \;\leq\; 1$ then there exists a prefix* **G**-*code $\mathcal{C} = (\mathbf{v}_1, \ldots, \mathbf{v}_R)$ with each $\mathrm{Len}(\mathbf{v}_j) = \ell_j$.*

[The also result holds for <u>*infinite*</u> *tuples $\vec{\boldsymbol{\ell}} = (\ell_1, \ell_2, \ell_3, \ldots)$ that satisfy $[\sum_{j=1}^{\infty} 1/\,\Gamma^{\ell_j}] \leq 1$.]* ◇

Exer. E2. Give an example of a code, $\mathcal{X}$, that violates (2a). [So $\mathcal{X}$ must fail to be UD.] □

Defn. A code $\mathcal{C}$ is ***weakly-UD*** if the following holds. *For each posint N and words $\mathbf{v}_i, \mathbf{w}_i \in \mathcal{C}$:*

1.1′: *If $\mathbf{v}_1\mathbf{v}_2\ldots\mathbf{v}_N \;=\; \mathbf{w}_1\mathbf{w}_2\ldots\mathbf{w}_N$ then $\forall i$: $\mathbf{v}_i = \mathbf{w}_i$.*

Contrast this with the (1.1) defn of ***UD***. □

Exer. E3. Posting race: Who can be the first to post a code which is weakly-UD, but not UD? □

Preliminaries for (2a). The below proof uses $S_{n,\ell}$, the number of length-ℓ <u>strings</u> which are concatenations of n many codewords. E.g, consider a code $\mathcal{C} = \{\mathbf{v},\mathbf{w},\mathbf{z}\}$ have lengths $5, 7, 8$, respectively.

$S_{1,15} = |\varnothing| = 0.$ $\quad S_{2,15} = |\{\mathbf{wz},\mathbf{zw}\}| = ?$

$S_{3,15} = |\{\mathbf{vvv}\}| = 1.$ $\quad S_{4,15} = |\varnothing| = 0.$

Indeed, $S_{n,15}$ is zero for each $n \geq 4$. As for $S_{2,15}$: If $\mathbf{wz} = \mathbf{zw}$ then $S_{2,15} = 1$, else $S_{2,15} = 2$. □

Proof of (2a). WLOGenerality, $\mathcal{C}$ is finite. (Exer. E4)
With $\Gamma := |\mathbf{G}|$, our goal is

$2a'$: $$\sum_{\mathbf{v}\in\mathcal{C}} 1/\,\Gamma^{\mathrm{Len}(\mathbf{v})} \;\stackrel{?}{\leq}\; 1\,.$$

WELOG, suppose the shortest and longest words in $\mathcal{C}$ have lengths 3 and 7. For $n = 1, 2, \ldots$, each string in $\mathcal{C}^n$ has a length, ℓ, in $[3n\,..\,7n]$; let $S_{n,\ell}$ be the number of such strings. Certainly $S_{n,\ell} \leq \Gamma^\ell$, the number of *all* length-ℓ strings over **G**. So the "generating function"

$$F_n(x) \quad:=\quad \sum_{\ell=3n}^{7n} [S_{n,\ell}\cdot x^\ell]$$

satisfies, for $x > 0$, that $F_n(x) \leq \sum_{\ell=3n}^{7n} \Gamma^\ell \cdot x^\ell$. Thus

$*$: $$F_n(\tfrac{1}{\Gamma}) \;\leq\; \sum_{\ell=3n}^{7n} \Gamma^\ell\cdot\tfrac{1}{\Gamma^\ell} \;\stackrel{\text{note}}{=}\; 1 + 7n - 3n \;\stackrel{\text{note}}{\leq}\; 5n\,,$$

for each posint n.

Using uniqueness. Fix n and an $\ell \in [3n\,..\,7n]$.

The coefficient of x^ℓ in $[F_1(x)]^n$ is the number of $\mathcal{C}$-n-parsings of length-ℓ strings, whereas $S_{n,\ell}$ is the number of length-ℓ strings which admit a $\mathcal{C}$-n-parsing.

The UD-hypothesis [actually, only "weakly-UD" is being used] says these two numbers are equal. Hence our two polynomials are equal,

$$[F_1(x)]^n \;=\; F_n(x). \quad \textit{So } (*) \textit{ implies}$$
$$[F_1(\tfrac{1}{\Gamma})]^n \;\leq\; 5n\,.$$

The LhS is exponential in n, whilst the RhS is linear. Thus $F_1(\frac{1}{\Gamma}) \leq 1$. Finally, observe that $F_1(\frac{1}{\Gamma})$ is a rewriting of LhS(2a). ♦

Proof of (2b). We'll show the idea for $\Gamma = 2$. Arrange the lengths as $\ell_1 \leq \ell_2 \leq \ldots \leq \ell_R$. On the full binary-tree of depth $D := \ell_R$, put weight $1/2^D$ on each leaf-node. All the nodes start as ***free***; we will iteratively mark some as ***busy*** as we create words $\mathbf{v}_1, \mathbf{v}_2, \ldots$. Call a node ***very-free*** if it and all its descendants are *free*, i.e not *busy*.

Let $\mathbf{v}_1$ be the leftmost path down to depth ℓ_1; so $\mathbf{v}_1 = 000\overset{\ell_1}{\ldots}0$. Mark $\mathbf{v}_1$ and all its descendants as *busy*. This action creates busy *leaf*-nodes of total weight

$$2^{D-\ell_1}\cdot\tfrac{1}{2^D} \;\stackrel{\text{note}}{=}\; 1/2^{\ell_1}\,.$$

With $d := \ell_1$, note that

$*$: *Each free node at depth* $\geq d$ *is very-free.*

Let $\mathbf{v}_2$ be the leftmost path to a free node at depth ℓ_2. [So $\mathbf{v}_2$ has $\ell_1 - 1$ many 0s, then a 1, then $\ell_2 - \ell_1$ many 0s.] Mark $\mathbf{v}_2$ and its descendants as *busy*. Now the total weight of busy leaf-nodes is

$$\frac{1}{2^{\ell_1}} + \frac{1}{2^{\ell_2}} .$$

Moreover, with $\boxed{d := \ell_2}$, note $(*)$ holds, since $\ell_2 \geq \ell_1$.

We'd like to continue using depth ℓ_3, depth ℓ_4, ..., depth ℓ_k, ... The only obstruction at a stage k, is if there is no *free* node at depth ℓ_k. But the total leaf-weight we've used up so far, is

$$W := \sum_{j=1}^{k-1} 1/\,2^{\ell_j} .$$

Since this sum is *strictly* less than 1, there exists a free-node at depth ℓ_{k-1}. (*Indeed, the number of such free-nodes is precisely* $[1-W]/2^{k-1}$.) Finally, since $\ell_k \geq \ell_{k-1}$, there is certainly a free-node at depth ℓ_k. ♦

2c: ***Defn.*** For a Γ-code with lengths $\vec{\boldsymbol{\ell}} = (\ell_1, \ldots, \ell_R)$, use

$$\Sigma(\vec{\boldsymbol{\ell}}) := \Sigma_\Gamma(\vec{\boldsymbol{\ell}}) := \sum_{j=1}^{R} 1/\Gamma^{\ell_j}$$

for its ***Kraft-sum***. Kraft's thm says –if the code is UD– that $\Sigma(\vec{\boldsymbol{\ell}}) \leq 1$. If equality, then the code [ditto the tuple] is ***complete***, otherwise it is ***redundant***; more precisely, Γ***-complete*** and Γ***-redundant***.

Given tuples $\vec{\boldsymbol{\ell}} = (\ell_1, \ldots, \ell_N)$ and $\vec{\mathbf{s}} = (s_1, \ldots, s_R)$, write $\vec{\boldsymbol{\ell}} \preccurlyeq \vec{\mathbf{s}}$ if $N{=}R$ and $\boxed{\forall j\colon \ell_j \leq s_j}$. Write $\vec{\boldsymbol{\ell}} \prec \vec{\mathbf{s}}$ if $\vec{\boldsymbol{\ell}} \preccurlyeq \vec{\mathbf{s}}$ yet $\vec{\boldsymbol{\ell}} \neq \vec{\mathbf{s}}$. [Ditto for ∞ tuples.] Note $\vec{\boldsymbol{\ell}} \preccurlyeq \vec{\mathbf{s}}$ implies $\Sigma(\vec{\boldsymbol{\ell}}) \geq \Sigma(\vec{\mathbf{s}})$ □

Exer. E5. A finite Γ-bounded tree T with R many leaves, yields a length-***spectrum*** $\vec{\boldsymbol{\ell}} = (\ell_1, \ldots, \ell_R)$; so terms "$\Gamma$-complete" and "$\Gamma$-redundant" makes sense for the tree. Prove:

2d: Completeness Lemma. *A finite Γ-bounded tree, T, is Γ-complete IFF it is Γ-full.* ♢

In (2e), below, we first consider only *binary* prefix-codes; $\Gamma = 2$.

2e: K-M Completeness corollary. *If finite tuple $\vec{\mathbf{s}}$ has $\Sigma(\vec{\mathbf{s}}) \leq 1$, then there exists a complete prefix-code with tuple $\vec{\boldsymbol{\ell}} \preccurlyeq \vec{\mathbf{s}}$.* ♢

Proof. We need but produce a complete $\vec{\boldsymbol{\ell}} \preccurlyeq \vec{\mathbf{s}}$, since Kraft's thm will hand us a prefix-code with lengths $\vec{\boldsymbol{\ell}}$.

It suffices, given a redundant $\vec{\mathbf{s}}$, to produce an $\vec{\boldsymbol{\ell}} \prec \vec{\mathbf{s}}$ with $\Sigma(\vec{\boldsymbol{\ell}}) \leq 1$. After all, there are only finitely-many tuples $\prec\vec{\mathbf{s}}$, so iterating will eventually halt, at a complete tuple.

WLOG, $T := s_1$ is a max-length in $\vec{\mathbf{s}}$; so each $1/2^{s_j}$ is a multiple of $1/2^T$, hence so is $\Sigma(\vec{\mathbf{s}})$. As $\vec{\mathbf{s}}$ is redundant, the ***gap*** $1 - \Sigma(\vec{\mathbf{s}})$ dominates $1/2^T$. So define $\vec{\boldsymbol{\ell}}$ by $\ell_2 := s_2, \ldots, \ell_R := s_R$, and $\ell_1 := s_1 - 1$. ♦

Exer. E6. POSTING RACE: Does (2e) hold for larger alphabet-sizes? If so, how does the proof need to be modified? □

Exer. E7. POSTING RACE: A block code is an example of a ***prefix/suffix-code***, i.e, both. (Dis)Prove: *There exists a complete prefix/suffix-code $\mathcal{C}$ whose length-spectrum is not constant.* □

Sardinas-Patterson Algorithm. An example of a UD-code [indeed, it is a suffixcode], for which the SarPat algorithm eventually cycles (as it must), but not with the empty prefix-list, is

$$\{\texttt{bc}, \texttt{b}, \texttt{Xc}, \texttt{cX}\} .$$

(On hold...)

Decoding-delay for UD-codes. Consider a long word $\mathbf{w}$ which is the initial part...
(On hold...)

Cryptography

Affine codes. Breaking affine codes with known/chosen plaintext.

Diffie-Hellman and El Gamal.

RSA. Pollard-ρ algorithm and Floyd cycle-finding alg..

Data compression

[Huffman codes. Source coding. In Spring2019: Skipped Ziv-Lempel.]

Expected coding-length

The binary numeral for posint K has form $1\text{Bits}(K)$, where $\text{Bits}(K)$ is a $\{0,1\}$-word. E.g, $\text{Bits}(23) = 0111$ because $\text{Binary}(23) = 10111$. Also $\text{Bits}(3) = 1$ and $\text{Bits}(2) = 0$ and $\text{Bits}(1) = \varepsilon$, the nullword. Let

$$|K|_{\text{Bit}} := |\text{Bits}(K)|\,.$$

So $|23|_{\text{Bit}} = 4$, $|2|_{\text{Bit}} = 1$ *and* $|1|_{\text{Bit}} = 0$.

With $n := |K|_{\text{Bit}}$, then, $2^{n+1} > K \geq 2^n$.

Exer.E8. Posting race: *Produce an infinite prefix-code* $\mathcal{C} = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots\}$ *such that* $\lim_{K\to\infty} \frac{|\mathbf{v}_K|}{|K|_{\text{Bit}}} = 1$. □

Exer.E8.1. Infinite prefix-code $\mathcal{C} = \{\mathbf{w}_1, \mathbf{w}_2, \ldots\}$ has the property that each

$$\dagger\text{:} \qquad |\mathbf{w}_K| \leq |K|_{\text{Bit}} + f(|K|_{\text{Bit}})\,,$$

where $f:\mathbb{Z}_+\to\mathbb{N}$. Prove that $\lim_{n\to\infty} f(n) = \infty$, using that $\mathcal{C}$ satisfies the Kraft inequality. □

Exer.E8.2. (Dis)Prove: $\exists$ prefix code $\mathcal{C}=\{\mathbf{w}_1, \mathbf{w}_2, \ldots\}$ satisfying (†), *with* $f(n) \leq \text{Const} + [1.007]\cdot\log(n)$. □

Exer.E8.3. (Dis)Prove: $\exists$ prefcode $\{\mathbf{w}_1, \mathbf{w}_2, \ldots\}$ with $\lim_{K\to\infty} \frac{|\mathbf{w}_K|}{|K|_{\text{Bit}}} = 1$ and subseq $K_1 < K_2 < \ldots$ with each $|\mathbf{w}_{K_\ell}| \leq |K_\ell|_{\text{Bit}} + 99$. □

Probability distr. A ***probability distribution*** on a codeword-set $\mathcal{C}$ is a map $P{:}\mathcal{C}{\to}[0,1]$ st.

3a: $$\sum_{\mathbf{v}\in\mathcal{C}} P(\mathbf{v}) \quad = \quad 1\,.$$

We will usually discard from the code all probability-zero words. In practice, then, a "probability distribution" is a map $P{:}\mathcal{C}{\to}(0,1)$ fulfilling (3a)

The ***expected***[♡2] ***coding length*** of $\mathcal{C}$ is

3b: $$\mathrm{ECL}(\mathcal{C}) \;:=\; \sum_{\mathbf{v}\in\mathcal{C}} P(\mathbf{v})\cdot \mathrm{Len}(\mathbf{v})\,.$$

E.g, consider code $\mathcal{C} := \{\mathbf{w}_1,\ldots,\mathbf{w}_4\}$ where

3c: $$\mathbf{w}_1 := \texttt{00}\,,\ \mathbf{w}_2 := \texttt{010}\,,\ \mathbf{w}_3 := \texttt{011}\,,\ \mathbf{w}_4 := \texttt{1}\,,$$

where $P(\mathbf{w}_4) = \frac{1}{2}$, and the other three words have probability $\frac{1}{6}$. Then $\mathrm{ECL}(\mathcal{C})$ is then

$$\tfrac{1}{2}\cdot 1 \;+\; \tfrac{1}{6}\cdot[2+3+3] \quad = \quad \tfrac{11}{6}\,.$$

Codemap. A ***source alphabet*** Ω, also called a "message set", might be

$$\{\texttt{a},\texttt{b},\ldots,\texttt{z},\ \texttt{.}\,,\mathit{Space}\}\,,$$

or might be $\{\texttt{tank},\texttt{ship},\ldots,\texttt{plane}\}$. Fixing a ***code-alphabet*** $\mathbf{G}$, a map $f{:}\Omega{\to}\mathbf{G}^{+}$ is a ***codemap*** (or ***cipher***) if

i: f is injective, *and*

ii: $\mathcal{C} := \mathrm{Range}(f)$ is a code. [Phrased this way, so that if we change our defn of "code" for a given context, then the defn of ***codemap*** changes with it.]

Every adjective applying to a code, also applies to a codemap; e.g, "a ***block/prefix/UD*** codemap".

ECL. Consider a [finite or countably-infinite] message set Ω and a probability distribution $P{:}\Omega{\to}[0,1]$. A codemap $f{:}\Omega{\to}\mathbf{G}^{+}$ puts a probability-distribution on $\mathcal{C} := \mathrm{Range}(f)$ by assigning, for $\mathbf{w}\in\mathcal{C}$,

4a: $$P(\mathbf{w}) \;:=\; P(f^{-1}(\mathbf{w}))\,.$$

Thus the code has an *expected coding-length*, which we may write as

$$\mathrm{ECL}(\mathcal{C}) \quad\text{or}\quad \mathrm{ECL}(f)\,.$$

[♡2] "Expected" is what probabilists use for "average".

MECL. Use ***MECL*** for Minimum ECL. Consider a *finite* prob-vector $\vec{\mathbf{p}} = (p_1,\ldots,p_L)$. A code [for the moment, assume a binary code] $\mathcal{C} = (\mathbf{v}_1,\ldots,\mathbf{v}_L)$ has

3b′: $$\mathrm{ECL}(\mathcal{C}) \;=\; \sum_{j=1}^{L} p_j\cdot \mathrm{Len}(\mathbf{v}_j)\,.$$

The minimum of (3b′) taken over *all* prefix-codes, or over all UD-codes, we will call

4b: $$\text{PC-MECL}(\vec{\mathbf{p}}) \quad\text{and}\quad \text{UD-MECL}(\vec{\mathbf{p}})\,,$$

respectively. Evidently

4c: $$\text{PC-MECL}(\vec{\mathbf{p}}) \;\geq\; \text{UD-MECL}(\vec{\mathbf{p}})$$

since, for UD-codes, we are taking a minimum over the larger collection of codes. By the way, I'll sometimes use $\mathrm{MECL}(\vec{\mathbf{p}})$ as a synonym for $\text{UD-MECL}(\vec{\mathbf{p}})$.

The minimum in (3b′) *depends on* $\Gamma := |\mathbf{G}|$, the number of letters in our code alphabet. [We can compress English more by coding into a 3-letter alphabet, rather than a 2-letter alphabet.] To indicate the dependency on cardinality Γ, we may write

4d: $$\text{PC-MECL}_{\Gamma}(\vec{\mathbf{p}}) \quad\text{and}\quad \text{UD-MECL}_{\Gamma}(\vec{\mathbf{p}})\,.$$

Huffman codes

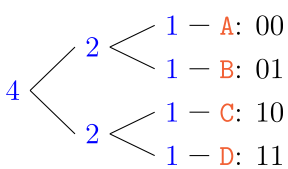
[Binary HCs will be described in class.]

Interpret a tuple such as (3:A 1:B 5:C) as putting prob-distribution $(\frac{3}{9}, \frac{1}{9}, \frac{5}{9})$ on letters (A, B, C); the 9 is the sum of the ***weights***, $3+1+5$.

Our convention is that the branch going *up-right* is labeled with bit 0; the down-right with bit 1. [On exams, all coalescings will be of *distinct* probabilities, and I'll ask that you put the *smaller* probability on the 0-branch.]

Non-uniqueness of Huffman Codes. Frequency-tuple $F :=$ (1:A 1:B 1:C 1:D) admits HC

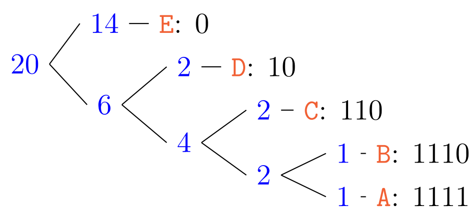
5a:



But F also admits each other permutation of {A, B, C, D} being attached to those leaves. So this Freq-tuple admits several HCs.

For a more interesting example, consider Frequency-tuple $F' :=$ (1:A 1:B 2:C 2:D 14:E). This admits HC $\mathcal{C}_1$:

5b:



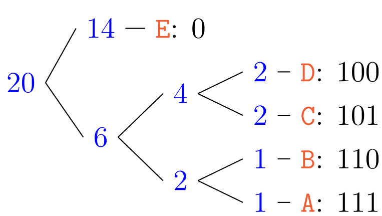
So 20·ECL($\mathcal{C}_1$) equals [Weight · WordLen · Count]

$$\overbrace{1{\cdot}4{\cdot}2}^{\text{B,A}} + \overbrace{2{\cdot}3{\cdot}1}^{\text{C}} + \overbrace{2{\cdot}2{\cdot}1}^{\text{D}} + \overbrace{14{\cdot}1{\cdot}1}^{\text{E}} = 32\,.$$

Thus $\text{ECL}(\mathcal{C}_1) = \frac{32}{20} = \frac{8}{5}$ bits-per-letter.

Our F' also admits HC $\mathcal{C}_2$:

5c:



Thus 20·ECL($\mathcal{C}_2$) equals

$$\overbrace{1{\cdot}3{\cdot}2}^{\text{B,A}} + \overbrace{2{\cdot}3{\cdot}2}^{\text{D,C}} + \overbrace{14{\cdot}1{\cdot}1}^{\text{E}} = 32\,.$$

We see that $\text{ECL}(\mathcal{C}_2) = \text{ECL}(\mathcal{C}_1)$. It is worth noticing that codes $\mathcal{C}_1$ and $\mathcal{C}_2$ are not only different, they are not even tree-isomorphic. □

6: HC-same-ECL Thm. *Fix a probability L-vec* $\vec{\mathbf{p}}$, *with* $L \geq 2$. *Then all* $\vec{\mathbf{p}}$*-HCs have the same* ECL. ♢

Proof. We proceed by induction on L, with proposition

$R(L)$: *For every prob. L-vec* $\vec{\mathbf{q}}$*: Each two* $\vec{\mathbf{q}}$*-HCs have the same* ECL.

The base $L{=}2$ case is easy, since the only Huffman-tree is Root $<$ Prob. Prob. whose ECL is 1.

Induction step. Fix an $L \geq 3$ st. $R(L{-}1)$.

Let $J := L{-}2$. Given $\vec{\mathbf{p}}$, let α, β denote its two lowest probabilities,[♡3] and write $\vec{\mathbf{p}}$ as $(\alpha, \beta, p_1, \ldots, p_J)$.

Consider two HCs, $\mathcal{C}$ and $\mathcal{X}$, with length-spectra that I have written above and below $\vec{\mathbf{p}}$, here.

$$\begin{array}{rcccccc}
\mathcal{C}: & D & D & d_1 & d_2 & \ldots & d_J \\
 & (\alpha, & \beta, & p_1, & p_2, & \ldots, & p_J) \\
\mathcal{X}: & Y & Y & y_1 & y_2 & \ldots & y_J\,.
\end{array}$$

So code $\mathcal{C}$ assigns length-D codewords to the first two nodes it joins, which have probs α and β. Computing

$$\dagger\!: \quad \begin{aligned}
\text{ECL}(\mathcal{C}) &= D{\cdot}\alpha + D{\cdot}\beta + \textstyle\sum_{i=1}^{J}[d_i \cdot p_i]\,; \\
\text{ECL}(\mathcal{X}) &= Y{\cdot}\alpha + Y{\cdot}\beta + \textstyle\sum_{i=1}^{J}[y_i \cdot p_i]\,.
\end{aligned}$$

After joining two nodes, the codes now recursively act on $\vec{\mathbf{q}} := (\alpha{+}\beta,\, p_1,\, p_2,\, \ldots,\, p_J)$ and assign length-spectra as follows:

$$\begin{array}{rcccccc}
\mathcal{C}: & D{-}1 & d_1 & d_2 & \ldots & d_J \\
 & (\alpha{+}\beta, & p_1, & p_2, & \ldots, & p_J) \\
\mathcal{X}: & Y{-}1 & y_1 & y_2 & \ldots & y_J\,.
\end{array}$$

Since $\vec{\mathbf{q}}$ is an $[L{-}1]$-vector, proposition $R(L{-}1)$ says that the above two ECLs are equal, i.e

$$\ddagger\!: \quad \begin{aligned}
&[D{-}1]{\cdot}[\alpha{+}\beta] + \textstyle\sum_{i=1}^{J}[d_i \cdot p_i] \\
&\quad = [Y{-}1]{\cdot}[\alpha{+}\beta] + \textstyle\sum_{i=1}^{J}[y_i \cdot p_i]\,.
\end{aligned}$$

And this implies equality in the two RhSs of (†). ♦

[♡3] They might be equal; indeed, perhaps $\beta = \alpha$, with 8 nodes all having probability α. We are not picking two *nodes*; we are picking two **probabilities**. In particular, I am not assuming that HCs $\mathcal{C}$ and $\mathcal{X}$ join the same two nodes, at the first step.

7a: Depth Lemma. *Fix a probability L-vector $\vec{\mathbf{p}}$, and a $\vec{\mathbf{p}}$–PC-MECL. Consider two leaf-nodes with probabilities α and α', at depths D and D', respectively. If $\alpha > \alpha'$, then necessarily $D \le D'$.* ◊

Exer. E9. Prove the above Depth Lemma. □

7b: Huffman's theorem.

i: HCs are PC-MECLs.

ii: HCs are UD-MECLs. ◊

Pf of (i). We induct on L, with proposition

HUFF(L): *Each probability L-vector $\vec{\mathbf{q}}$, admits a Huffman Code which is a PC-MECL.*

The base $L{=}2$ case is immediate, since the only tree is Root $<$ Prob. Prob. , which *is* a Huffman-tree.

Induction step. Fix an $L \ge 3$ st. HUFF($L{-}1$). Fix $\vec{\mathbf{p}}$, a prob. L-vector, and consider a $\vec{\mathbf{p}}$–PC-MECL, viewed as a tree.

Let $\alpha{\le}\beta$ denote the two smallest probabilities of $\vec{\mathbf{p}}$. At the tree's deepest level, D, consider two joined leaf-nodes, and call their probabilities x and y. It suffices to show:

∗: *We can permute the probabilities of the leaves, without changing the* ECL, *so that, now, these two nodes have probabilities α and β.*

For then, we collapse these two into a single node, producing prob.-vec $\vec{\mathbf{q}} := (\alpha{+}\beta, p_2, p_3, \ldots, p_{L-1})$. By the induction hypothesis, there is a $\vec{\mathbf{q}}$-HC which is a $\vec{\mathbf{q}}$–PC-MECL. Expanding the collapsed node back into $< \begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}$ automatically produces a Huffman-tree[♡4]. which is a $\vec{\mathbf{p}}$–PC-MECL. And all HCs have the same ECL, by (6).

[♡4]The permuting of probabilities, because it is done recursively, can permute interior-nodes of the tree. So the final Huffman-tree can be non-isomorphic to the original PC-MECL tree. This kind of argument is called ***tree surgery***.

Establishing (∗). If $x = \alpha$, then leave that leaf-node alone. Otherwise, $x > \alpha$. Our Depth Lemma, (7a), says that no α-node can be shallower than x, so [since x is at max depth], every α-node has to be at D, the deepest level. Switch some α-leaf with our x-leaf.

> *This does not change the* ECL, *since the nodes are at the same depth.*

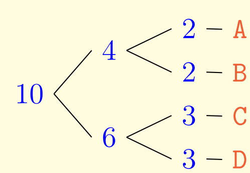
Now our joined-pair is $< \begin{smallmatrix}\alpha\\ y\end{smallmatrix}$. Do the same operation with y w.r.t β. *Now* our joined-pair is $< \begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}$, as desired. ♦

Exer. E10. Prove (ii), that every HC is a UD-MECL. □

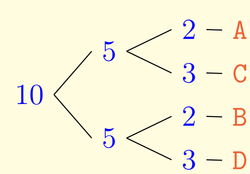
Pf of (ii), (E10). Fix $\vec{\mathbf{p}}$ and a $\vec{\mathbf{p}}$–UD-MECL; write its length-***spectrum*** as $\vec{\boldsymbol{\ell}} = (\ell_1, \ldots, \ell_R)$. By Kraft's thm, there is a PC-code with the same spectrum hence, when assigned to the same probabilities, has the same ECL. And part (i) shows there is a HC with the same ECL. ♦

Exer. E11. POSTING RACE: (Dis)Prove: *If prefix code $\mathcal{C}$ is a PC-MECL, then $\mathcal{C}$ is a Huffman code.* □

Solution to E11. False. Consider frequency-tuple (2:A 2:B 3:C 3:D). Its only Huffman-tree is

7c:


(This admits eight HCs, since at each of the three nodes we can choose which edge is labeled 0 and which is 1.) This codetree has ECL = 2. But *so does this tree,*

7d:


which is *not* a Huffman code. ♦

Entropy/Distropy

Define $\boldsymbol{\eta}\colon[0,1]\to[0,\infty)$ by $\boldsymbol{\eta}(x) := x\cdot\log_2(1/x)$, and extend by continuity, so that $\boldsymbol{\eta}(0)=0$. (Use l'Hôpital's rule, if you like.)

The ***distribution entropy***, which I call ***distropy***, of a probability-vector $\vec{\mathbf{v}}$ is

$$\mathcal{H}(\vec{\mathbf{v}}) := \sum_{p\in\vec{\mathbf{v}}} \boldsymbol{\eta}(p)\,.$$

For a probability-distr $P()$ on a code[♡5] $\mathcal{C}$, then, $\mathcal{H}(P)$ equals $\sum_{\mathbf{v}\in\mathcal{C}} \boldsymbol{\eta}(P(\mathbf{v}))$.

8: Distropy UD-code Inequality. *Fix a binary code $\mathcal{C}$ and probability distribution $P{:}\mathcal{C}\to(0,1)$. If $\mathcal{C}$ is uniquely decodable, then*

8a: $$\mathrm{ECL}(\mathcal{C}) \geq \mathcal{H}(P)\,.$$

There is equality in (8a) IFF

8b: $$\forall \mathbf{v}\in\mathcal{C}\colon\ P(\mathbf{v}) = 1/\,2^{\widehat{\mathbf{v}}}\,,$$

where, here, $\widehat{\mathbf{v}}$ means $\mathrm{Len}(\mathbf{v})$. ◊

Pf of (8a). Let "$\sum_{\mathbf{v}}$" mean "$\sum_{\mathbf{v}\in\mathcal{C}}$".

With $\mathscr{L}() := \log_2()$, note $\mathrm{ECL}(\mathcal{C})$ equals $\sum_{\mathbf{v}} P(\mathbf{v})\cdot\widehat{\mathbf{v}}$, which equals $\sum_{\mathbf{v}} P(\mathbf{v})\mathscr{L}(2^{\widehat{\mathbf{v}}})$. Consequently, we can write $\mathcal{H}(P)-\mathrm{ECL}(\mathcal{C})$ as

$$\Big[\sum_{\mathbf{v}} P(\mathbf{v})\,\mathscr{L}\big(\tfrac{1}{P(\mathbf{v})}\big)\Big] - \Big[\sum_{\mathbf{v}} P(\mathbf{v})\,\mathscr{L}(2^{\widehat{\mathbf{v}}})\Big] = \sum_{\mathbf{v}} P(\mathbf{v})\,\mathscr{L}\Big(\tfrac{1}{P(\mathbf{v})}\cdot\tfrac{1}{2^{\widehat{\mathbf{v}}}}\Big)\,.$$

Since $\mathscr{L}()$ is strictly convex-down, Jensen's inequality, (12), applies to say

†: $$\mathcal{H}(P)-\mathrm{ECL}(\mathcal{C}) \leq \mathscr{L}\Big(\sum_{\mathbf{v}} P(\mathbf{v})\cdot\tfrac{1}{P(\mathbf{v})}\tfrac{1}{2^{\widehat{\mathbf{v}}}}\Big) \overset{\text{note}}{=\!=} \mathscr{L}\Big(\sum_{\mathbf{v}} 1/2^{\widehat{\mathbf{v}}}\Big)\,.$$

By (2a) the Kraft-McMillan inequality, $\sum_{\mathbf{v}} 1/2^{\widehat{\mathbf{v}}} \leq 1$. And $\mathscr{L}()$ is order-preserving. Thus the above yields

$$\mathcal{H}(P)-\mathrm{ECL}(\mathcal{C}) \leq \mathscr{L}(1) = 0\,,$$

as desired. ♦

[♡5] For comparison with (binary) distropy/entropy, we will usually be examining a ***binary code***; a code over a 2-symbol alphabet, $\mathbb{B}$. (Typically, $\mathbb{B}=\{0,1\}$.) So a *binary code* is a subset $\mathcal{C}\subset\mathbb{B}^+$.

Pf of (8b). Suppose $\mathrm{ECL}(\mathcal{C})=\mathcal{H}(P)$. This forces equality in Kraft, so $\sum_{\mathbf{v}} 1/2^{\widehat{\mathbf{v}}}=1$, *and* in Jensen's, so the map $\mathbf{v}\mapsto\frac{1}{P(\mathbf{v})}\cdot\frac{1}{2^{\widehat{\mathbf{v}}}}$ is constant; say κ.

Thus $P(\mathbf{v})\cdot\kappa = 1/2^{\widehat{\mathbf{v}}}$, for each $\mathbf{v}$. Summing over all $\mathbf{v}\in\mathcal{C}$ implies that $1\cdot\kappa = 1$. Hence $\kappa=1$. ♦

Convention. For $p \in [0,1]$, let p^c mean $1-p$, in analogy with $P(B^c)$ equaling $1 - P(B)$ on a probability space. [See APPENDIX for independence, $\perp$, defns.]

9: Distropy fact. *For partitions* P, Q, R *on probability space.*

a: *$\mathcal{H}(\mathsf{P}) \le \log(^{\#}\mathsf{P})$, with equality IFF* P *is an equi-mass partition.*

b: *$\mathcal{H}(\mathsf{Q} \vee \mathsf{R}) \le \mathcal{H}(\mathsf{Q}) + \mathcal{H}(\mathsf{R})$, with equality IFF* $\mathsf{Q} \perp \mathsf{R}$.

c: *For $p \in [0, \frac{1}{2}]$, the function $p \mapsto \mathcal{H}(p, p^c)$ is strictly increasing.* ◊

Proof. Use the strict concavity of $\boldsymbol{\eta}()$, together with Jensen's Inequality. ♦

10: Binomial Lem. *Fix $p \in [0, \frac{1}{2}]$ and let $\boldsymbol{H} := \mathcal{H}(p, p^c)$. Then for each $n \in \mathbb{Z}_+$:*

10′: $$\sum_{j \in [0\,..\,pn]} \binom{n}{j} \le 2^{\boldsymbol{H} \cdot n}.$$ ◊

Proof. Let $X \subset \{0,1\}^n$ be the set of $\mathbf{x}$ with $^{\#}\{i \in [1\,..\,n] \mid x_i = 1\} \le p \cdot n$. On X, let $\mathsf{P}_1, \mathsf{P}_2, \ldots$ be the coordinate partitions; e.g $\mathsf{P}_7 = (A_7, {A_7}^c)$, where $A_7 := \{\mathbf{x} \mid x_7 = 1\}$. Weighting each point by $\frac{1}{|X|}$, the uniform distribution $\mu()$ on X, gives that $\mu(A_7) \le p$.

So $\mathcal{H}(\mathsf{P}_7) \le \boldsymbol{H}$, by (9c).

Finally, the join $\mathsf{P}_1 \vee \ldots \vee \mathsf{P}_n$ separates the points of X. So

$$\log(^{\#}X) = \mathcal{H}(\mathsf{P}_1 \vee \ldots \vee \mathsf{P}_n)$$
$$\le \mathcal{H}(\mathsf{P}_1) + \ldots + \mathcal{H}(\mathsf{P}_n) \le \boldsymbol{H}n,$$

making use of (9a,b). And $^{\#}X$ equals LhS(10′). ♦

NOTE: *Below, several quantities need to be natnums, and so some floor or ceiling symbols are needed. I have omitted them, to show the overall idea of the proof.*

11: Shannon source-coding thm. *Fix probability $0 < p < \frac{1}{2}$, and set $\boldsymbol{H} := \mathcal{H}(p, p^c)$. Consider the iid-process on alphabet $\{0,1\}$ with $P(1) = p$ (hence $P(0) = 1 - p$). Fix $\varepsilon > 0$. Then $\forall_{\text{large}} N$, there exists a block-code, mapping*

$$N \text{ bits} \to [\boldsymbol{H} + \varepsilon] \cdot N \text{ bits},$$

with error-probability $< \varepsilon$. ◊

Pf. Pick $\delta > 0$ so small that $\mathcal{H}(p+\delta, [p+\delta]^c) < \boldsymbol{H} + \varepsilon$. Define

$$X_N := \left\{ \vec{\mathbf{x}} \in \{0,1\}^N \;\middle|\; p - \delta < \mathrm{Freq}(1 \text{ in } \vec{\mathbf{x}}) < p + \delta \right\},$$

where the frequency is $\frac{1}{N}$ times the number of 1s in bit-string $\vec{\mathbf{x}}$. Courtesy the Binomial Lemma (10),

$$|X_N| \le 2^{[\boldsymbol{H}+\varepsilon] \cdot N}, \quad \textit{for all } N \in \mathbb{Z}_+.$$

And WLLN (13b) allows us to fix a large enough N such that

$$P(X_N) \ge 1 - \varepsilon. \quad \textit{Henceforth, } X := X_N.$$

Codemap. Let $K := \lceil [\boldsymbol{H} + \varepsilon] N \rceil$. Our Nbit→Kbit code, maps X [enumerated in, say, lexicographic order] to bit-strings

$$\overbrace{0\ldots00}^{K},\ \overbrace{0\ldots01}^{K},\ \overbrace{0\ldots10}^{K},\ \overbrace{0\ldots11}^{K},\ \ldots.$$

And the code maps each $\vec{\mathbf{x}} \in X^c$ to, say, $1 \overset{K}{\ldots} 1$.

Every word in X is decoded correctly, so the probability of error is $< \varepsilon$. ♦

Error-correcting codes

Hamming codes, distance, weight, bound.
Shannon's Noisy-channel Thm ...

§A Appendix

Various general tools.

12: Jensen's inequality. *On an interval $J \subset \mathbb{R}$, consider points $Q_{\mathbf{v}} \in J$, for each $\mathbf{v}$ in a countable indexing-set $\mathcal{C}$. We have a probability-distr $P()$ on $\mathcal{C}$. Then for each convex-down fnc $\mathscr{L}: J \to \mathbb{R}$*

12a: $$\mathscr{L}\Big(\sum_{\mathbf{v}\in\mathcal{C}} P(\mathbf{v}) \cdot Q_{\mathbf{v}}\Big) \geq \sum_{\mathbf{v}\in\mathcal{C}} P(\mathbf{v}) \cdot \mathscr{L}(Q_{\mathbf{v}}) .$$

Now suppose $\mathscr{L}$ is strictly convex-down. Then:

12b: *Equality in (12a) IFF the probability-distr is concentrated on a single point.*

IOWords, having removed all zero-probability elements from $\mathcal{C}$, the map $\mathbf{v} \mapsto Q_{\mathbf{v}}$ is constant.

Proof. Exercise. [Or see picture on blackboard.] ♢

Probability

A ***random variable*** [***r.var***] is a measurable map $\mathsf{Y}: \Omega \to \mathbb{R}$ where Ω is a probability space. [Can take Ω to be $[0,1)$.] Unless both the positive and negative parts of Y have infinite integral, the "***expectation*** of Y", $\mathsf{E}(\mathsf{Y}) := \int_\Omega \mathsf{Y}$, is a value in $[-\infty, +\infty]$.

When finite, it is common to call $\boldsymbol{\mu} := \mathsf{E}(\mathsf{Y})$ the ***mean*** of Y. Then ***variance*** $\mathrm{Var}(\mathsf{Y}) := \mathsf{E}([\mathsf{Y} - \boldsymbol{\mu}]^2)$ is well-defined, and could be $+\infty$.

Independence. Events A, B are ***independent***, written $A \perp B$, if $P(A \cap B) = P(A)P(B)$. A *family* $\mathfrak{C}$ of events is independent, written $\perp(\mathfrak{C})$ or $\perp(\{A\}_{A\in\mathfrak{C}})$, if each finite subset $A_1, \ldots, A_N$ has $P(A_1 \cap \ldots \cap A_N)$ equalling $\prod_{j=1}^N P(A_j)$. This property of $\mathfrak{C}$ is much stronger than ***pairwise independence***, where each pair of events in $\mathfrak{C}$ is independent.

Random variables X, Y are ***independent***, $\mathsf{X} \perp \mathsf{Y}$, if for each pair of measurable sets $S, T \subset \mathbb{R}$, events $\{\mathsf{X} \in S\}$ and $\{\mathsf{Y} \in T\}$ are independent. It turns out that this is equivalent to saying, for each pair $x, y \in \mathbb{R}$, that events $\{\mathsf{X} \leq x\} \perp \{\mathsf{Y} \leq y\}$. When $\mathsf{X} \perp \mathsf{Y}$ have finite expectations, then $\mathsf{E}(\mathsf{X} \cdot \mathsf{Y}) = \mathsf{E}(\mathsf{X}) \cdot \mathsf{E}(\mathsf{Y})$.

Extend notions of ***independence*** and ***pairwise independence*** to *collections* of random variables.

13a: Markov Lemma. *Consider posint n and random variable Y. For each $\varepsilon \in \mathbb{R}_+$:*

†: $$P(|\mathsf{Y}| \geq \varepsilon) \leq \frac{\mathsf{E}(|\mathsf{Y}|^n)}{\varepsilon^n}; \quad \text{Markov Inequality.}$$

When n is even,

‡: $$P(|\mathsf{Y}| \geq \varepsilon) \leq \frac{\mathsf{E}(\mathsf{Y}^n)}{\varepsilon^n} .$$

In particular, if Y has finite mean $\boldsymbol{\mu} := \mathsf{E}(\mathsf{Y})$, then

$$P(|\mathsf{Y} - \boldsymbol{\mu}| \geq \varepsilon) \leq \frac{\mathrm{Var}(\mathsf{Y})}{\varepsilon^2}; \quad \text{Chebyshev Inequality.}$$

Proof. Exercise. ♢

13b: Weak Law of Large Numbers (WLLN). *Consider an identically-distributed pairwise-independent sequence $\mathsf{X}_1, \mathsf{X}_2, \ldots$ where both mean $\boldsymbol{\mu} := \mathsf{E}(\mathsf{X})$ and variance $\mathbf{v} := \mathrm{Var}(\mathsf{X}) \stackrel{\text{def}}{=} \mathsf{E}([\mathsf{X} - \boldsymbol{\mu}]^2)$ are finite. Then*

$$\lim_{N\to\infty} P\Big(|\overline{\mathsf{X}}_N - \boldsymbol{\mu}| \geq \varepsilon\Big) = 0,$$

where $\overline{\mathsf{X}}_N := \frac{1}{N}\sum_{j=1}^N \mathsf{X}_j$. ♢

Proof. WLOG $\boldsymbol{\mu} = 0$. Then $N^2 \cdot \mathrm{Var}(\overline{\mathsf{X}}_N)$ equals

$$\mathsf{E}\Big(\Big[\sum_{j=1}^N \mathsf{X}_j\Big]^2\Big) = \Big[\sum_{i=1}^N \mathsf{E}(\mathsf{X}_i^{\,2})\Big] + \sum_{j\neq k}^N \mathsf{E}(\mathsf{X}_j \mathsf{X}_k)$$
$$= N\mathbf{v} + \sum_{j\neq k}^N \mathsf{E}(\mathsf{X}_j) \cdot \mathsf{E}(\mathsf{X}_k) = N\mathbf{v},$$

since each $\mathsf{E}(\mathsf{X}_j) = 0$. Thus $\mathrm{Var}(\overline{\mathsf{X}}_N) = \frac{\mathbf{v}}{N}$. Hence

$$P\Big(|\overline{\mathsf{X}}_N| \geq \varepsilon\Big) \leq \frac{\mathrm{Var}(\overline{\mathsf{X}}_N)}{\varepsilon^2} = \frac{1}{N} \cdot \frac{\mathbf{v}}{\varepsilon^2},$$

by the Chebyshev Inequality. ♦

§Index for "JK Codes notes"

This is a test of the pre-note.

Filename: `Problems/NumberTheory/jk-codes.latex`
As of: *Saturday 30Mar2019*. Typeset: *10May2023* at *11:10*.