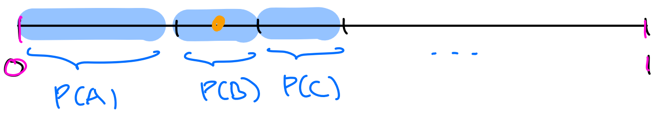
Arithmetic Coding (§6)

Today: Streaming compression algo for explicit probabilistic model $P(x_n | x_1, \dots, x_{n-1})$ ← not necessarily IID !

KEY IDEA:



$\mathcal{A} = \{A, B, C, \dots\}$

* To communicate message, simply send some number in interval (in binary)
* $P(x)$ large ⇒ interval large ⇒ few bits needed

Let's talk about numbers and intervals...

Binary expansions: Any $0 \le f < 1$ can be written as

$$f = 0.b_1 b_2 b_3 \dots = \frac{b_1}{2} + \frac{b_2}{4} + \frac{b_3}{8} + \dots$$

* NOT unique, e.g. $0.1 = 0.011111\dots$
* **Standard binary expansion:** ← avoids this

$$\boxed{\begin{array}{l} \text{for } k = 1, 2, \dots: \\ \quad b_k \leftarrow \begin{cases} 0 & \text{if } f < \frac{1}{2} \\ 1 & \text{otherwise} \end{cases} \\ \quad f \leftarrow 2f - b_k \end{array}}$$

"the" binary expansion

e.g. $\frac{1}{3} = 0.\underbrace{010101}_{\text{periodic!}}\dots$

$\left(\frac{1}{3} \to \frac{2}{3} \to \frac{4}{3} - 1 = \frac{1}{3} \to \dots\right)$

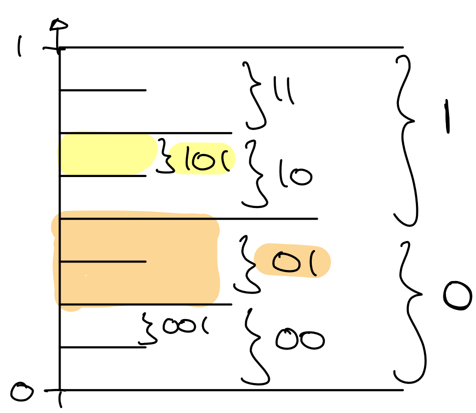
e.g. $\frac{5}{6} = 0.110101\dots$

$\left(\frac{5}{6} \to \frac{5}{3} - 1 = \frac{2}{3} \to \frac{4}{3} - 1 = \frac{1}{3} \to \dots\right)$

Binary ("dyadic") intervals: Given bitstring $b_1 b_2 \dots b_\ell$, define

$$I(b_1 \dots b_\ell) := [0.b_1 b_2 \dots b_\ell,\ 0.b_1 b_2 \dots b_\ell + 2^{-\ell})$$

← general interval of form $\left[\frac{j}{2^\ell}, \frac{j+1}{2^\ell}\right)$



* Smaller intervals ⟺ more bits
* $I(b_1 \dots b_\ell) \ni f \iff f = 0.b_1 \dots b_\ell b_{\ell+1} \dots$
* $I(s) \cap I(\tilde{s}) \neq \emptyset \iff I(s) \subseteq I(\tilde{s})$, or vice versa
  ⟺ s is <u>prefix</u> of $\tilde{s}$, or vice versa

* If $J = [f-r, f+r)$ arbitrary interval with midpoint f & radius r:

$$J \supseteq I(b_1 \cdots b_\ell), \quad \text{where } f = 0.b_1 \cdots b_\ell \cdots, \quad \ell = \lceil \log \tfrac{1}{r} \rceil$$

Pf: $I(b_1 \cdots b_\ell)$ has size $2^{-\ell} \leq r$, contains f

$\Rightarrow I(b_1 \cdots b_\ell) \subseteq [f-r, f+r)$ $\square$

We now use this to construct a simple prefix code, following the above idea:

Let P probability distribution on $\mathcal{A} = \{a_1 < \cdots < a_m\}$ — we order the symbols in some arbitrary way

↳ lower & upper cumulative probabilities:

$$Q(x) := \sum_{y<x} P(y) \quad \& \quad R(x) := \sum_{y \leq x} P(y) = Q(x) + P(x)$$

↳ disjoint intervals $J(x) = [Q(x), R(x))$ with midpoint $F(x) = \frac{Q(x)+R(x)}{2}$ & radius $\frac{P(x)}{2}$

e.g.

x	A <	B
$P(x)$	$\frac{2}{3}$	$\frac{1}{3}$
$Q(x)$	0	$\frac{2}{3}$
$R(x)$	$\frac{2}{3}$	1
$F(x)$	$\frac{1}{3}$	$\frac{5}{6}$
ℓ	2	3
$C(x)$	01	110

Shannon–Fano–Elias code:

$$C(x) = b_1 \cdots b_\ell$$

where $F(x) = 0.b_1 \cdots b_\ell b_{\ell+1} \cdots$

$$\ell = \left\lceil \log \frac{2}{P(x)} \right\rceil = \left\lceil \log \frac{1}{P(x)} \right\rceil + 1$$

* this is a prefix code: $I(C(x)) \subseteq J(x)$ disjoint

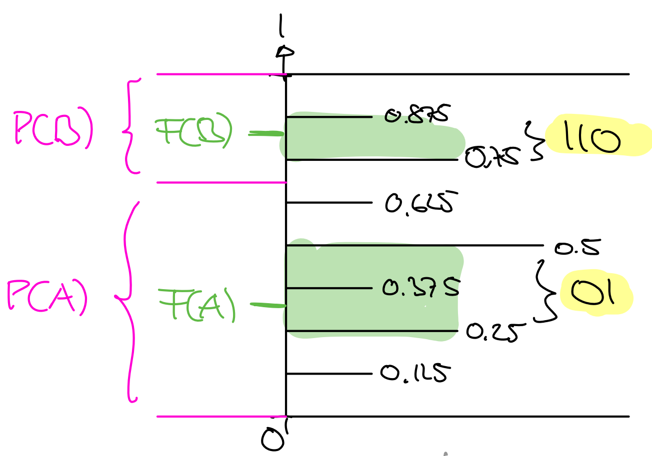
* higher info content $\Leftrightarrow$ more bits

* $H(P) + 1 \leq L(C, P) \leq H(P) + 2$

* when applied to X^N:

average rate $\approx \frac{H(X^N)}{N}$:)

... but no better than block Huffman!



Could even use larger intervals → 0 & 1

How to turn this into a streaming code?   Not possible for Huffman!

Assume we are given conditional probability distributions

$$P(x_n \mid \underbrace{x_1,\dots,x_{n-1}}_{x^{n-1}}) \quad \text{for } n=1,2,\dots$$

← "language model"

* typically only depends on last $k-1$ characters

  $k=1$: IID, $k=2$: "digram", $k=3$: "trigram", ... ⇝ "k-gram model"

* $P(x^n) = P(x_1)\,P(x_2 \mid x_1)\,P(x_3 \mid x_1 x_2) \cdots P(x_n \mid \underbrace{x_1 \cdots x_{n-1}}_{x^{n-1}})$ arbitrary joint distribution!

key ideas:

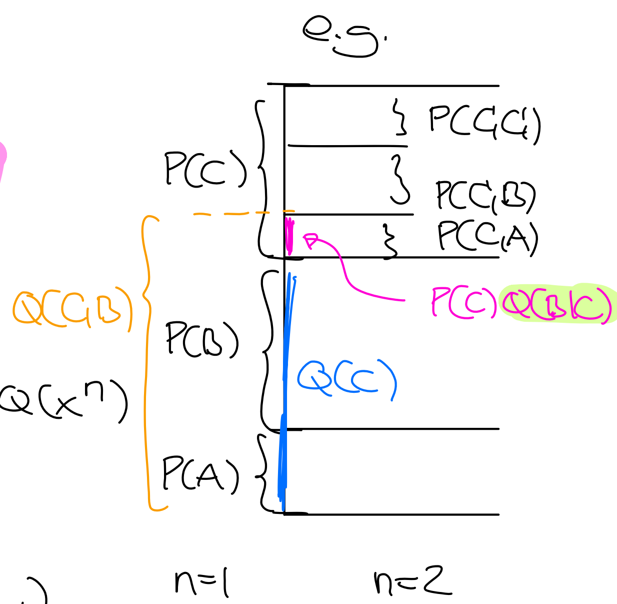
① Can compute P, Q, R recursively:

$$Q(x^n) = Q(x^{n-1}) + P(x^{n-1})\,Q(x_n \mid x^{n-1})$$

$$\underbrace{R(x^n)}_{\text{blocksize } n} = \underbrace{Q(x^{n-1})}_{\text{blocksize } n-1} + P(x^{n-1})\,R(x_n \mid x^{n-1})$$

$$P(x^n) = P(x^{n-1}) \cdot P(x_n \mid x^{n-1}) = R(x^n) - Q(x^n)$$

e.g.



in terms of cumulative conditional probabilities:

$$Q(x_n \mid x_1,\dots,x_{n-1}) := \sum_{y < x_n} P(y \mid x_1,\dots,x_{n-1})$$

$$R(x_n \mid x_1,\dots,x_{n-1}) := \sum_{y \le x_n} P(y \mid x_1,\dots,x_{n-1})$$

lexicographic order

② Start sending bits as soon as possible

Since intervals become smaller & smaller, more & more bits are fixed!

...this leads to the following algorithm...

Arithmetic coding:

Input: $x^N \in \mathcal{A}^N$ to compress

Algo.

* $q \leftarrow 0,\ r \leftarrow 1,\ p \leftarrow 1$
* For $n = 1, 2, \ldots, N$:

  ① $r \leftarrow q + p\,R(x_n | x_1, \ldots, x_{n-1})$

  $q \leftarrow q + p\,Q(x_n | x_1, \ldots, x_{n-1})$

  ② While $r \leq \frac{1}{2}$ or $q \geq \frac{1}{2}$:

  $$b \leftarrow \begin{cases} 0 & \text{if } r \leq \frac{1}{2} \\ 1 & \text{if } q \geq \frac{1}{2} \end{cases}$$

  Write b

  $r \leftarrow 2r - b$

  $q \leftarrow 2q - b$

  Remove b from binary expansion

  In this case ANY number in $[q, r)$ starts with $0.b$, so can write b

  ③ $p \leftarrow r - q$

* Write $\lceil \log \frac{2}{p} \rceil$ bits of binary expansion of $\frac{q+r}{2}$ ← like in Shannon-Fano-Elias

* Average rate: $\approx \frac{H(X^N)}{N}$ if compressing $X^N \sim P(x_1, \ldots, x_N)$

* Without step ②, algorithm reduces to (block) Shannon–Fano–Elias

  Step ② does NOT change output, but makes it streaming also!

* How to decompress? $\boxed{\text{EX CLASS}}$

* What if we don't know language model? Learn "on the fly" → $\boxed{\text{EX CLASS}}$