Arithmetic Coding (§6)

Last time: LZ stream compression, adaptive, no explicit probabilistic model, asymptotically optimal for IID sources (also for ergodic sources ...)

Today: Streaming compression algo for explicit probabilistic model
$P(x_{N+1} | x_1 \cdots x_N)$ ← not necessarily IID !

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

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

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

for $k = 1, 2, \ldots$:

$$b_k \leftarrow \begin{cases} 0 & \text{if } f < \frac{1}{2} \\ 1 & \text{otherwise} \end{cases}$$

$f \leftarrow 2f - b_1$

"the" binary expansion

e.g. $\frac{1}{3} = 0.010101\ldots$ periodic !

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

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

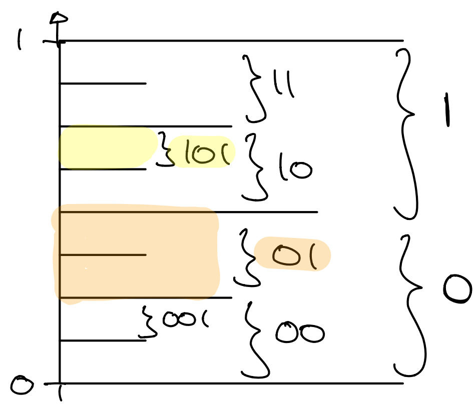
$\left(\frac{5}{6} \to \frac{5}{3} - 1 = \frac{2}{3} \to \frac{4}{3} - 1 = \frac{1}{3} \to \ldots\right)$

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

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

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

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



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

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

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

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

Why useful? As warmup, let's use this to construct a simple prefix code...

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:

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

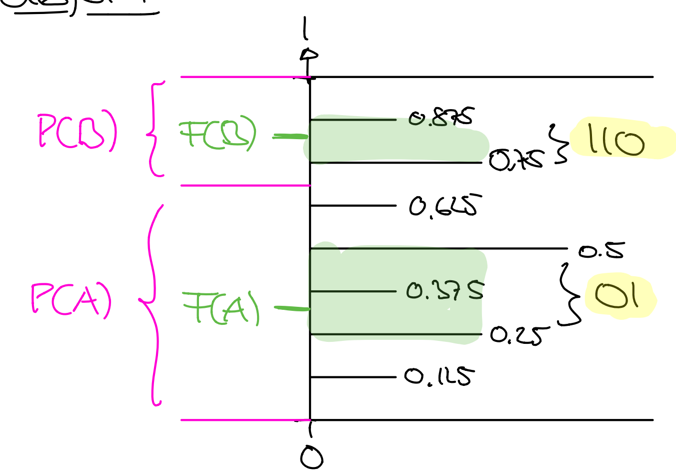
where $F(x) = 0.b_1 \cdots b_\ell b_{\ell+1} \cdots$

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

* 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 & 11

We now discuss how to turn this into a streaming code!

Not possible for Huffman?

For Simplicity: Assume $\boxed{X^N \overset{\text{iid}}{\sim} P}$ ← Not necessary, see below!

Key ideas:

① Can compute P, Q & R recursively:

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

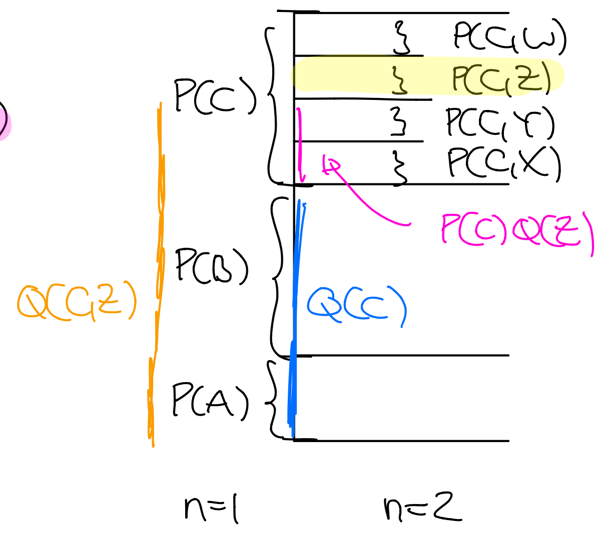
$$\underbrace{R(x^n)}_{\text{blocksize } n} = \underbrace{Q(x^{n-1})}_{\text{blocksize } n-1} + P(x^{n-1})\,R(x_n)$$

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

② Start sending bits as soon as possible

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

e.g.



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 $R \leftarrow Q + p\,R(x_n)$,
  $\quad Q \leftarrow Q + p\,Q(x_n)$
  ② While $R \leq \frac{1}{2}$ or $Q \geq \frac{1}{2}$: ← In this case ANY number in $[R, Q)$ starts with $0.b$, so can write b
  $\quad b \leftarrow \begin{cases} 0 & R \leq \frac{1}{2} \\ 1 & Q \geq \frac{1}{2} \end{cases}$
  $\quad$ Write b
  $\quad \left.\begin{array}{l} R \leftarrow 2R - b \\ Q \leftarrow 2Q - b \end{array}\right\}$ Remove b from binary expansion
  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

* Without Step ②, algorithm reduces to (block) Shannon-Fano-Elias
* Step ② does NOT change output, but makes it streaming also!

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

! Arithmetic coding generalizes directly to non-IID situations:

Assume we are given conditional probability distributions

$P(x_n | x_1, \ldots, x_{n-1})$ for $n = 1, 2, \ldots$ ← "language model"

* typically only depends on $x_{n-k+1}, \ldots, x_{n-1}$
* $k=1$: IID, $k=2$: "digram", $k=3$: "trigram", ... ⇝ "k-gram model"
* Can even learn language model "on the fly" → $\boxed{\text{EX CLASS}}$

Joint distribution: $P(x^N) = P(x_1)\, P(x_2|x_1)\, P(x_3|x_1 x_2) \cdots P(x_N | x^{N-1})$

How to adapt Arithmetic Coding also? Only need to adjust step (i) by

$$\boxed{\begin{aligned} R &\leftarrow Q + p\, R(x_n | x_1, \ldots, x_{n-1}) \\ Q &\leftarrow Q + p\, Q(x_n | x_1, \ldots, x_{n-1}) \end{aligned}}$$

where

$$\left.\begin{aligned} Q(x_n | x_1, \ldots, x_{n-1}) &:= \sum_{y < x_n} P(y | x_1, \ldots, x_n) \\ R(x_n | x_1, \ldots, x_{n-1}) &:= \sum_{y \leq x_n} P(y | x_1, \ldots, x_n) \end{aligned}\right\} \text{cumulative conditional probabilities}$$

↳ $\boxed{\text{average rate: } \approx \frac{H(X^N)}{N}}$ if compressing $X^N \sim P(x_1, \ldots, x_N)$