ILLC Project Course in Information Theory

Crash course

13 January – 17 January 2014
12:00 to 14:00

Student presentations

27 January – 31 January 2014
12:00 to 14:00

Location

ILLC, room F1.15,
Science Park 107, Amsterdam

Materials

informationtheory.weebly.com

Contact

Mathias Winther Madsen
mathias.winther@gmail.com

Monday

Probability theory
Uncertainty and coding

Tuesday

The weak law of large numbers
The source coding theorem

Wednesday

Random processes
Arithmetic coding

Thursday

Divergence
Kelly Gambling

Friday

Kolmogorov Complexity
The limits of statistics

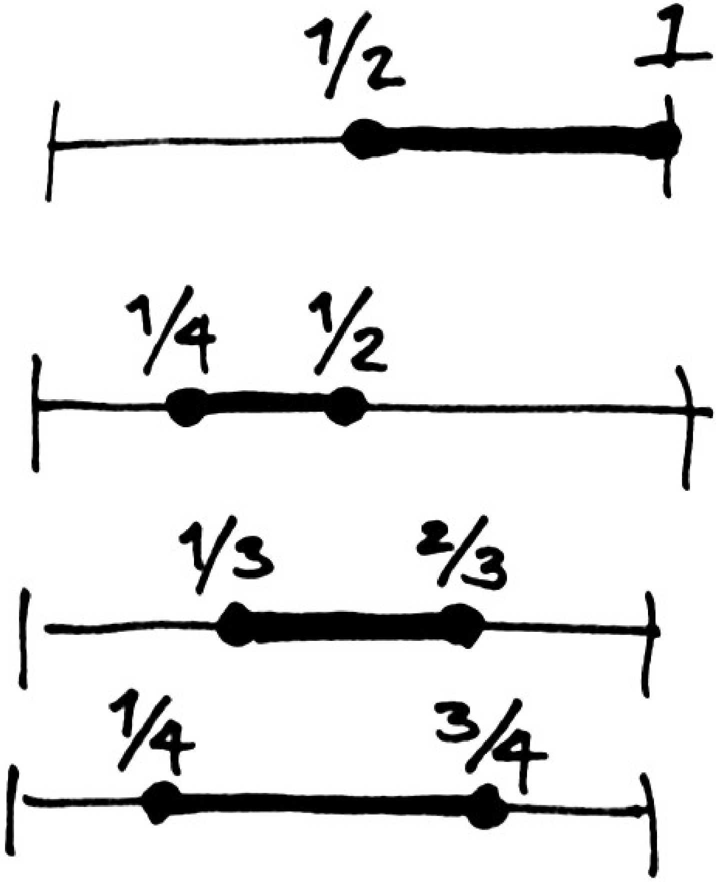
PLAN

- Binary intervals
- Codes for binary intervals
- Inner binary intervals
- The chain rule for intervals
- Arithmetic coding

A binary interval is an interval of the form

$$[\, s\, 2^{-k}, (s + 1)\, 2^{-k}],$$

where *s* and *k* are integers.



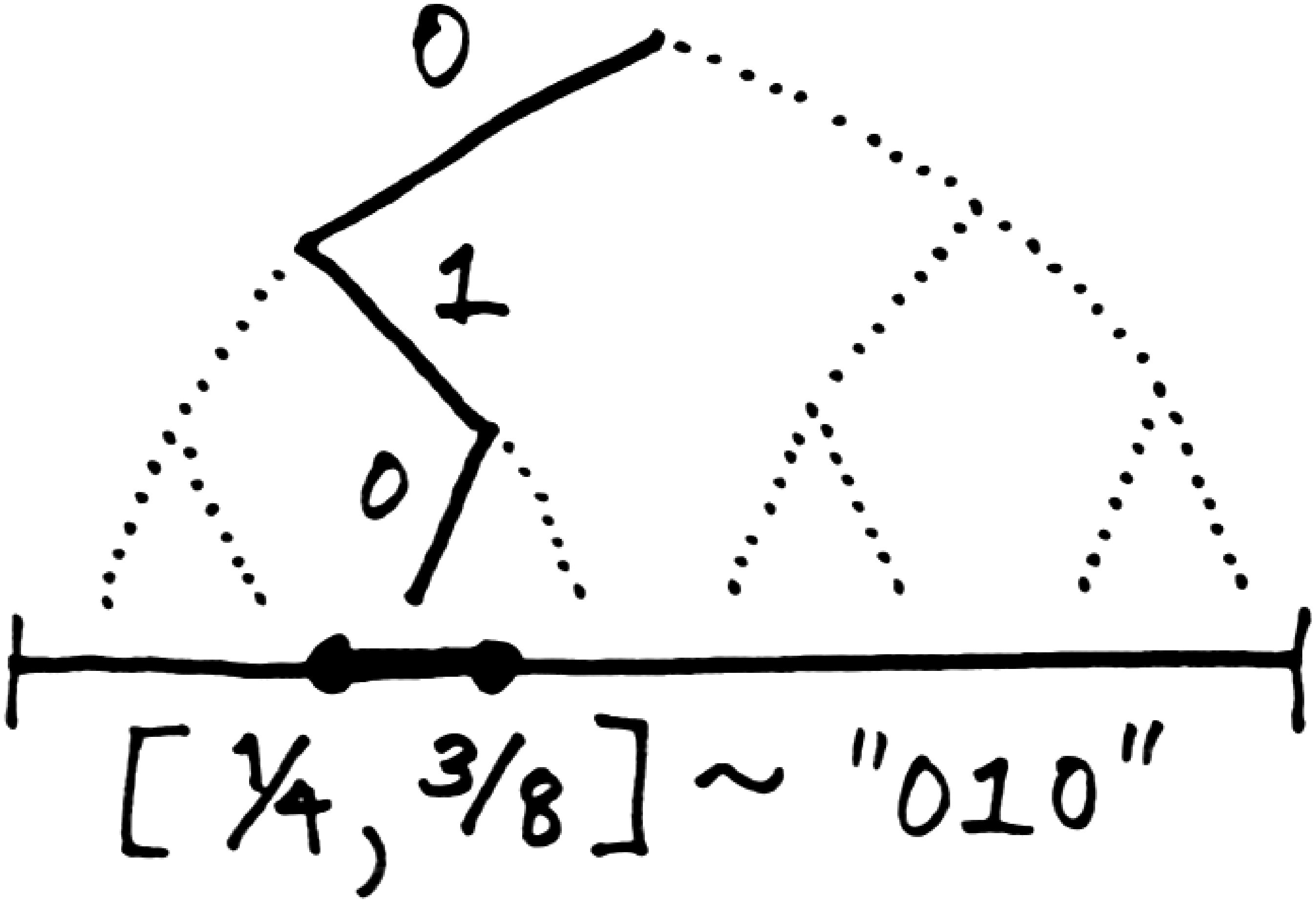
Yes!

Yes!

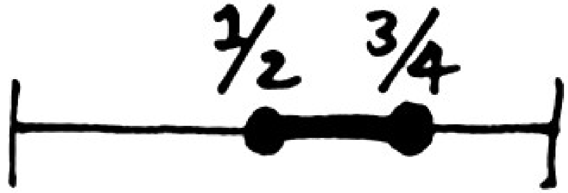
No!!

No!!

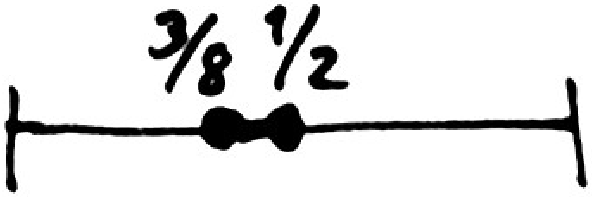
Binary intervals have names.



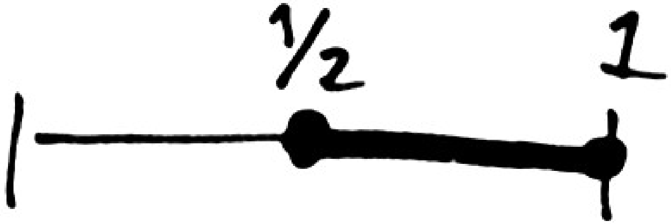
Binary intervals have names.



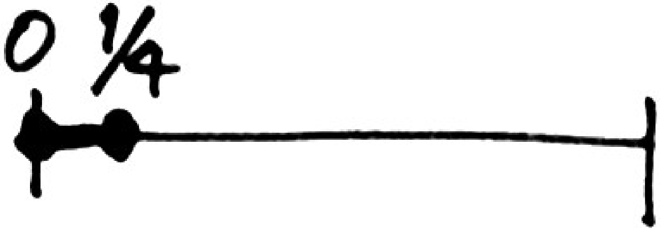
~ 10



~ 011

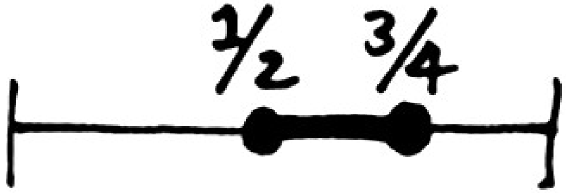


~ ?

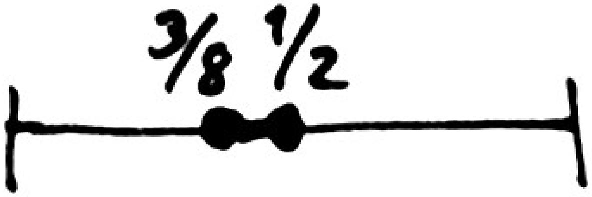


~ ?

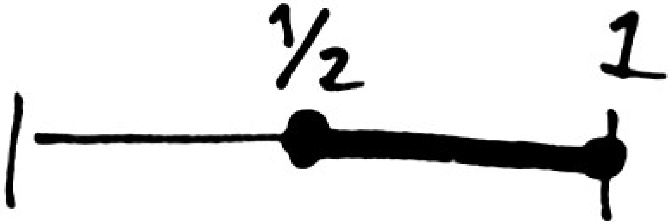
Binary intervals have names.



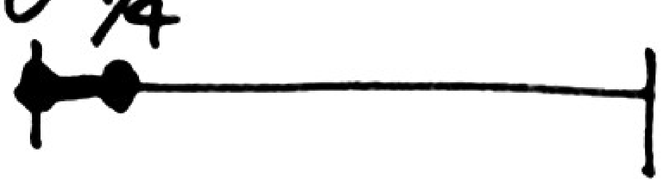
~ 10



~ 011

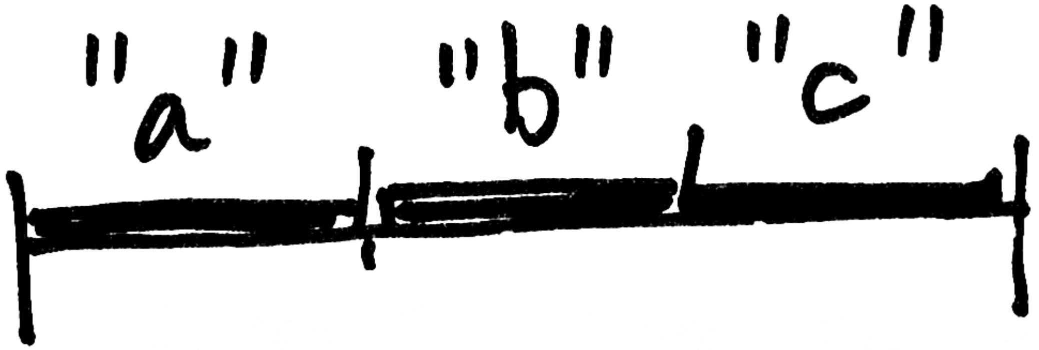


~ 1

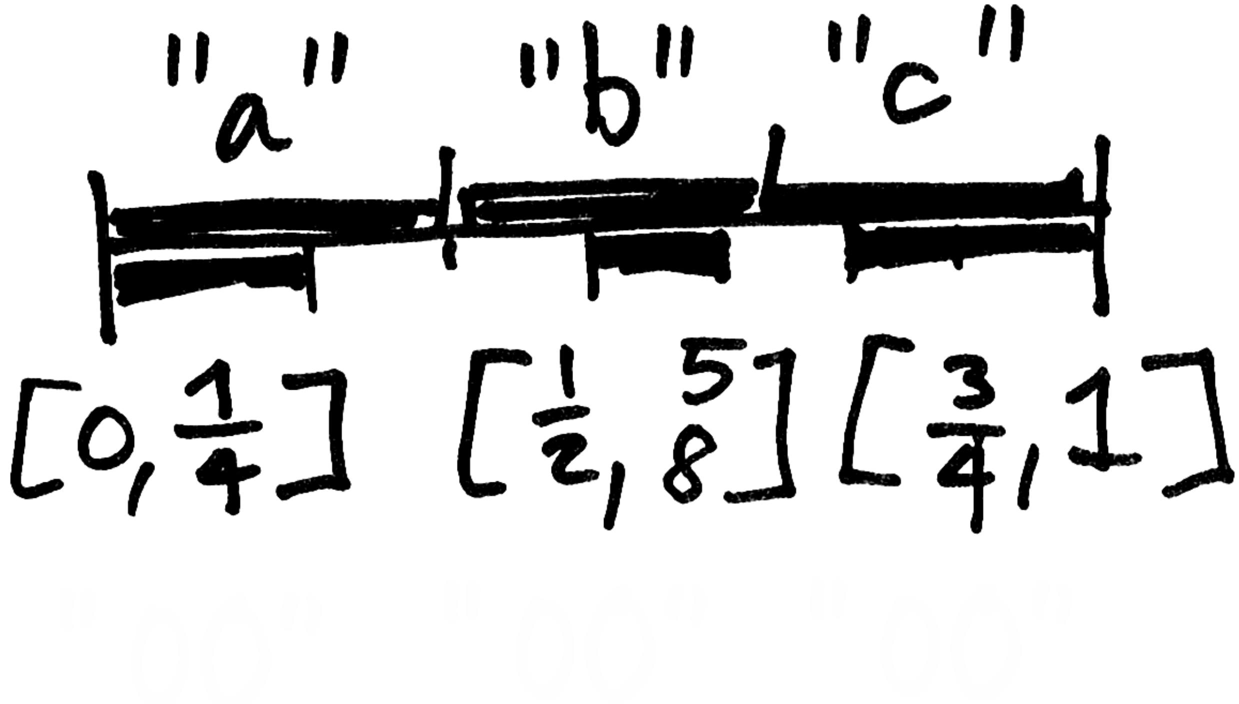


~ 00

A distribution is a set of intervals

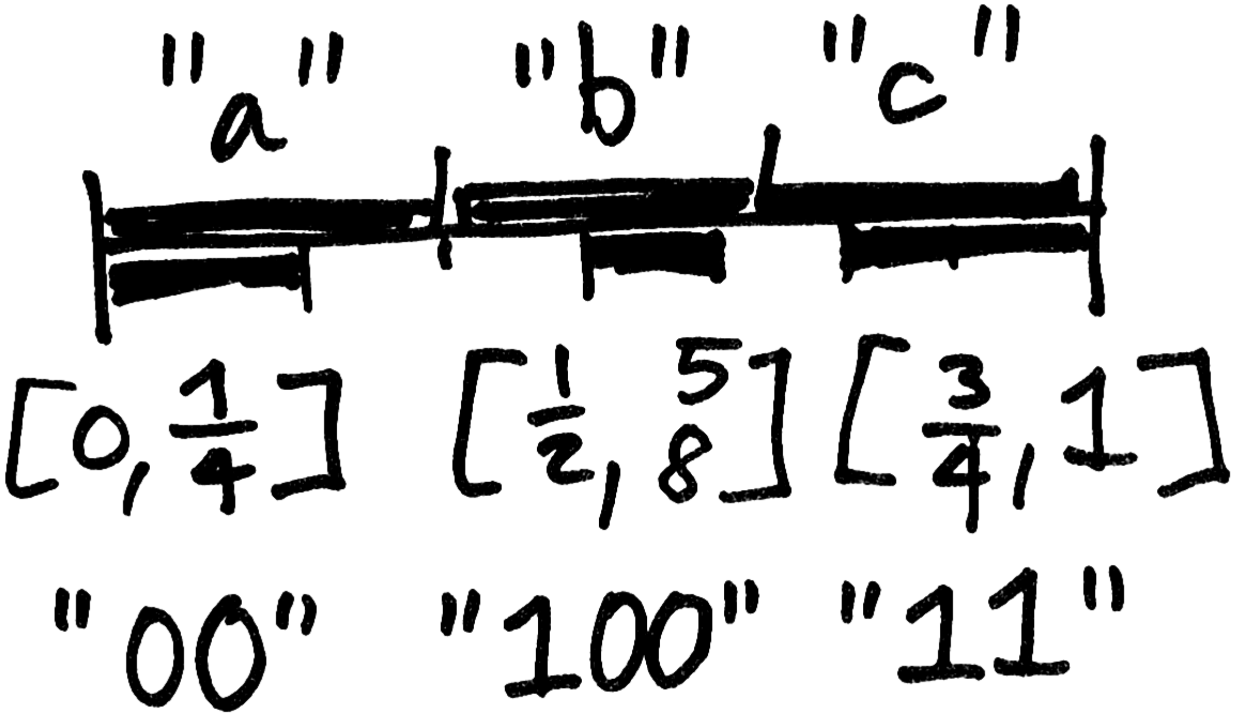


A distribution is a set of intervals



which can be approximated by binary intervals.

A distribution is a set of intervals



which can be approximated by binary intervals.

The binary intervals have names.

[0, .4]	**[.4, 1]**
[0, 1/4]	**[1/2, 1]**
00	1

[0, .3]	**[.3, .6]**	**[.6, 1]**
[0, 1/4]	**[3/8, 1/2]**	**[3/4, 1]**
00	011	11

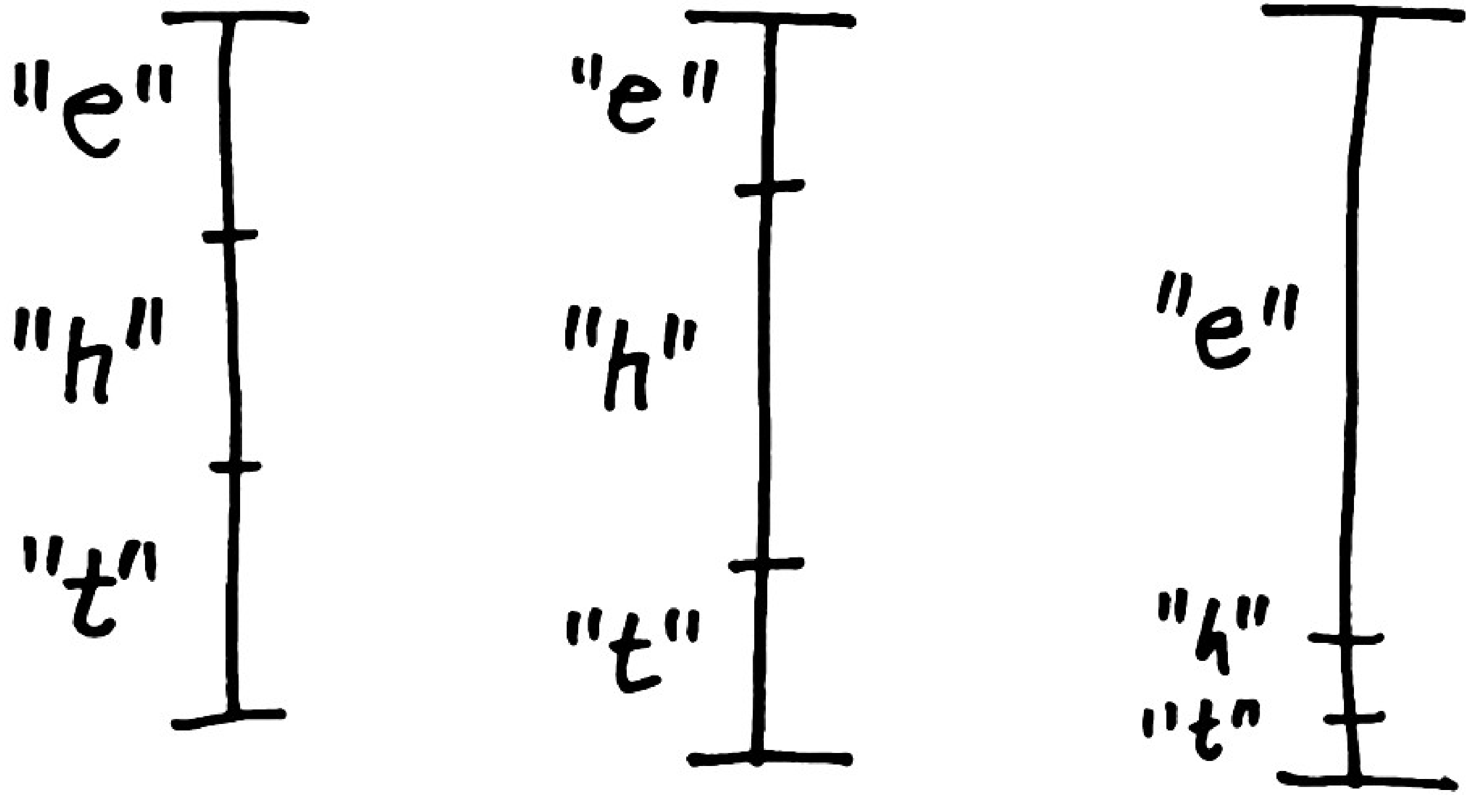
[0, .2]	**[.2, .4]**	**[.4, 1]**
?	?	?
?	?	?

[0, .4]	**[.4, 1]**
[0, 1/4]	**[1/2, 1]**
00	1

[0, .3]	**[.3, .6]**	**[.6, 1]**
[0, 1/4]	**[3/8, 1/2]**	**[3/4, 1]**
00	011	11

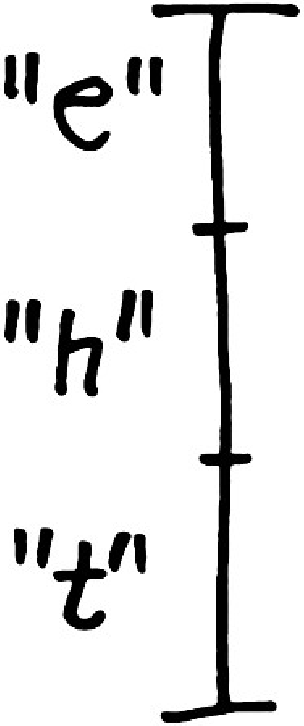
[0, .2]	**[.2, .4]**	**[.4, 1]**
[0, 1/8]	**[1/4, 3/8]**	**[1/2, 1]**
000	010	1

Conditional distributions are also distributions

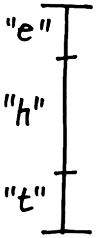


$\mathrm{Pr}(x_1)$ $\mathrm{Pr}(x_2 \mid x_1)$ $\mathrm{Pr}(x_3 \mid x_1, x_2)$

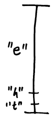
But by the chain rule, they should be downscaled.



$\Pr(x_1)$

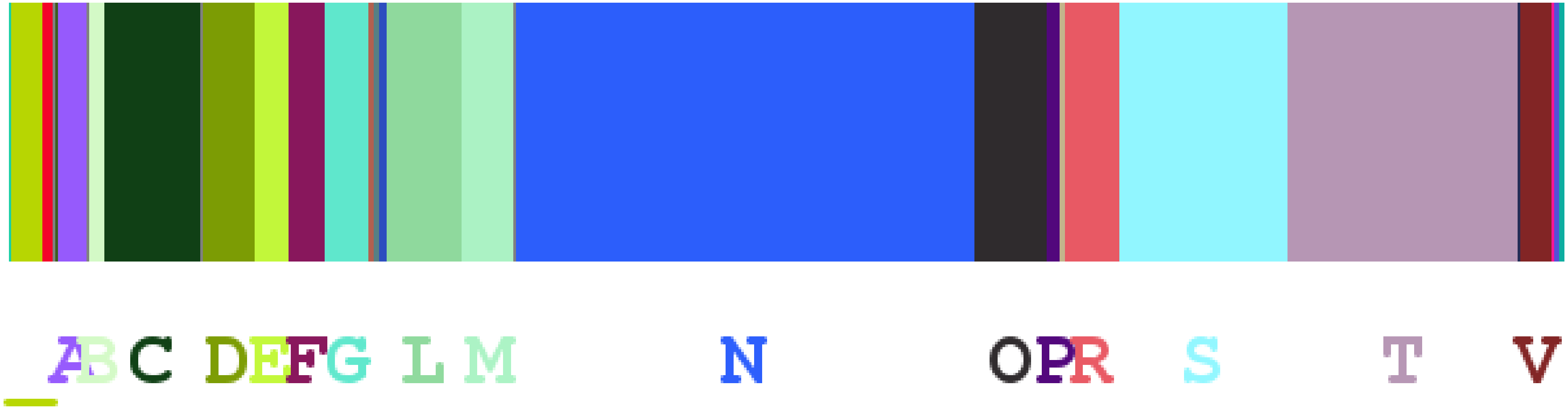


$\Pr(x_2 \mid x_1) \Pr(x_1)$



$\Pr(x_3 \mid x_1, x_2) \Pr(x_2 \mid x_1) \Pr(x_1)$

LOGIC IS BUT ELECTRICAL ENGINEERI



Arithmetic coding:

1. Distribute the unit line into segments according to your probability estimates.

2. Whenever a new observation arrives, subdivide the interval corresponding to that observation according to your conditional probability estimates.

3. Keep dividing and subdividing, and always use your best probability estimate of what the next character is going to be.

4. When the text ends, find the inner binary interval and output its codeword.

Straws (McKay, Exercise 15.4) How can you use a fair coin to draw lots among three people? Come up with at least two different alterantives and compare them in terms of (1) fairness, and (2) expected number of coin flips.

Arithmetic coding for a bent coin Suppose we are going to do $n = 2$ flips of a bent coin with bias $p = 1/4$.

1. Construct the arithmetic code for the outcomes of this experiment.
2. If k_i is the length of the ith codeword, what is $\sum_i 2^{-k_i}$?
3. How does that compare to the same sum for $n = 1$?

Spaced-out language A language consists of all binary strings with no consecutive `1`s. Find a code for the set of messages from this language, assuming that the length of the message is known in advance.

Palindrome machine A function picks an $L = 1, 2, 3, 4, \ldots$ with probabilities $1/2, 1/4, 1/8, 1/16, \ldots$ and then returns a binary palindrome of length L. Possible return values are, e.g., `1`, `00`, and `1001`, but not `1010` or `10`.

A machine repeatedly calls this function and prints the outputs. An output stream from this machine is thus a series of palindromes like `00 101 11 1001...`, but without the spaces.

You start this machine and observe the output

```
1100...
```

What is the probability that the next character is a `0`?

```
repeat indefinitely:

   S = ""

   while flip():

      if flip():
         print "0"
         append "0" to S

      else:
         print "1"
         append "1" to S

   print the last bit of S
   delete the last bit from S
```