Unit 3 Homework: Information Theory and Computability

Algorithmic Information Dynamics: A Computational Approach to Causality and Living Systems From Networks to Cells

Hector Zenil, Narsis A. Kiani, Alyssa Adams

Course available at the Santa Fe Institute MOOC platform the Complexity Explorer at: https://www.complexityexplorer.org/courses/63-algorithmic-information-dynamics-a-computational-approach-to-causality-and-living-systems-from-networks-to-cells

```
Out[•]= Tue 24 Jul 2018 17:00:48
```

Classical Information Theory

Question 1

What happens to the entropy of a random string when increasing the number of symbols, e.g. from binary to trinary? **Would the entropy increase or decrease?**

Answer 1

It increases, because entropy is a function of alphabet (symbol) size. This is because the set of all possible strings grows and thus also the uncertainty if the symbols are uniformly distributed. A numerical experiment confirms it:

```
In[•]:= Table[RandomChoice[{0, 1, 2}, 100] // Entropy // N, 100] // Mean
Out[•]= 1.08842
```

Question 2

What would happen if we used base 2 for our strings made of 0’s and 1’s? What would happen if we used base 3 for our strings made of 0’s, 1’s, and 2’s? **Would the entropy increase or decrease?**

Answer 2

It would increase because the base acts as some sort of normalization factor making the minimum and maximum entropy to be 0 and 1

```
In[•]:= Table[Entropy[2, RandomChoice[{0, 1}, 100]] // N, 100] // Mean

Out[•]= 0.991429
```

```
In[•]:= Table[Entropy[3, RandomChoice[{0, 1, 2}, 100]] // N, 100] // Mean

Out[•]= 0.990607
```

If we equal the base to the alphabet size, the entropy of random strings approximates 1.

Question 3

Lets keep *e* and investigate the behavior of entropy as a function of alphabet base.

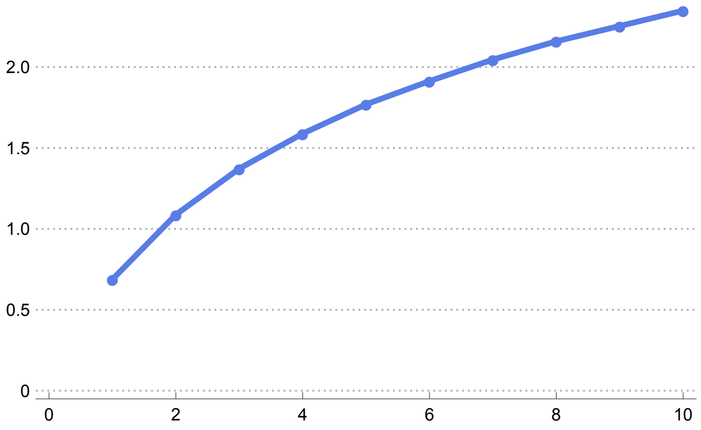
Let’s go all the way up to alphabet size 11.

```
In[8]:= alphabetentropy =
         Table[Table[RandomChoice[Range[0, i], 100] // Entropy // N, 100] // Mean, {i, 1, 10}]

Out[8]= {0.688475, 1.08796, 1.37251, 1.58865,
         1.76957, 1.91329, 2.0461, 2.15903, 2.25369, 2.34842}
```

```
In[9]:= ListPlot[alphabetentropy, PlotTheme → "Business",
         Joined → True, PlotMarkers → Automatic]
```

Out[9]=



Why do you think the curve starts to slow down?

Answer 3

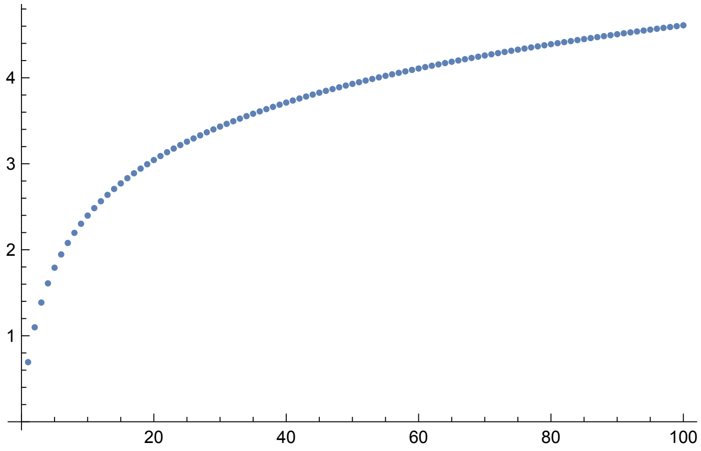
Because we are increasing the number of symbols, but our string length is staying the same. If we did this same plot with longer strings, we'd see something slightly different:

```
In[10]:= alphabetentropy = Table[
           Table[RandomChoice[Range[0, i], 10 000] // Entropy // N, 100] // Mean, {i, 1, 100}]
```

```
Out[10]= {0.693093, 1.09852, 1.38616, 1.60924, 1.79149, 1.9456, 2.07911, 2.1968, 2.30215,
          2.39739, 2.48439, 2.5643, 2.63841, 2.70733, 2.77187, 2.83242, 2.88952,
          2.94355, 2.99479, 3.04354, 3.08994, 3.13439, 3.17693, 3.21768, 3.25682,
          3.29454, 3.33082, 3.36586, 3.39979, 3.43246, 3.46422, 3.49488, 3.52472,
          3.5537, 3.58183, 3.60921, 3.63577, 3.66163, 3.68697, 3.71159, 3.73562,
          3.75916, 3.7821, 3.8045, 3.82644, 3.84791, 3.86883, 3.88945, 3.90957, 3.92929,
          3.94866, 3.96767, 3.98623, 4.00471, 4.02264, 4.04023, 4.05755, 4.0747,
          4.09146, 4.10794, 4.1242, 4.14011, 4.15574, 4.17118, 4.18639, 4.2014, 4.21624,
          4.23076, 4.24511, 4.25931, 4.27314, 4.28683, 4.30039, 4.3139, 4.32701,
          4.33999, 4.35297, 4.36545, 4.37811, 4.39056, 4.40257, 4.41465, 4.42665,
          4.43843, 4.45003, 4.46156, 4.47293, 4.48419, 4.4953, 4.50641, 4.5174, 4.52803,
          4.53863, 4.54919, 4.55966, 4.56994, 4.58019, 4.59012, 4.60017, 4.61022}
```

```
In[11]:= ListPlot[alphabetentropy]
```

Out[11]=



Question 4

Let's say we tossed a coin 1000 times and we got about 50% heads and 50% tails. If we measured the entropy of the heads and tails outcome, we would expect the entropy to be very close to 1, since it is a random process. However, say we found that the coin was weighted, with the tails side being heavier. How would this change the way we see our result?

Answer 4

We would be very surprised that the heads and tails distribution is close to 50-50. Given that we know the tails side is heavier, we would expect to see more heads than tails.

Question 5

Say you find a mysterious DNA sequence. It has lots of interesting patterns and redundancy. For fun, you look at the first half of the sequence and given what you see, you are able to accurately predict the second half. Good job! Would this sequence have a higher or lower Shannon entropy? Does this mean the sequence is noisy?

Answer 5

This means the sequence has low Shannon entropy and is not very noisy.

Question 6

What would you expect the conditional entropy to be between two randomly-generated strings of the same length (made of 1’s and 0’s)?

Answer 6

Will be 1 because knowing one tells you nothing about the other one, so the uncertainty remains the highest:

```
In[12]:= string1 = RandomChoice[{0, 1}, 100]
         string2 = RandomChoice[{0, 1}, 100]

Out[12]= {0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1,
          1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
          1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1,
          0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0}

Out[13]= {0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1,
          1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
          0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0,
          0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0}

In[14]:= Statistics`Library`NConditionalEntropy[string1, string2]

Out[14]= 0.951918
```

Question 6

What about for the following strings?

```
In[•]:= string1 = Table[1, {100}]
        string2 = Table[0, {100}]

Out[•]= {1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}

Out[•]= {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

Answer 6

Is zero, because H (Y | X) = 0 if and only if the value of Y is completely determined by the value of X. There are no surprised with Y just as there are no with X (and viceversa).

```
In[15]:= Statistics`Library`NConditionalEntropy[string1, string2]

Out[15]= 0.951918
```

Question 7

What is the Mutual Information between the following strings?

```
In[16]:= string1 = {0, 1, 0, 1, 0, 1, 0, 1, 0, 1};

In[17]:= string2 = {1, 0, 1, 0, 1, 0, 1, 0, 1, 0};

In[18]:= string3 = RandomInteger[1, 10]

Out[18]= {0, 1, 1, 0, 0, 1, 0, 1, 1, 1}
```

Answer 7

Removed from zero between string1 and string2 because one is the complement of the other and thus knowing about one tells us a lot about the other, but closer to zero between string1 and string3 and thus between string2 and string3 because string3 was ‘randomly’ generated.

```
In[19]:= N[Entropy[string1] + Entropy[string2] -
           Statistics`Library`NConditionalEntropy[string1, string2]]

Out[19]= 1.38629

In[20]:= N[Entropy[string1] + Entropy[string3] -
           Statistics`Library`NConditionalEntropy[string1, string3]]

Out[20]= 0.51972
```

```
In[21]:= N[Entropy[string2] + Entropy[string3] -
           Statistics`Library`NConditionalEntropy[string2, string3]]

Out[21]= 0.51972
```

Question 8

What about the Mutual Information between string2 and string1?

Answer 8

It would be the same as between string1 and string2 because Mutual Information is symmetric.

```
In[22]:= N[(Entropy[string2] + Entropy[string1]) -
           Statistics`Library`NConditionalEntropy[string2, string1]]

Out[22]= 1.38629
```

Computability

Question 1

Is a universal Turing machine with 1 tape computationally more powerful than a universal Turing machine with 3 tapes? What about an infinite number of tapes?

Answer 1

No, once a Turing machine is universal it reaches the maximal computational power. Adding tapes may make it faster but it is not computationally more powerful, even with an infinite number of tapes.

Question 2

Write a transition table of rules for a Turing machine with only 2 states that always halts for any input.

Answer 2

Here is one:
X, S -> H
where X is any symbol, S is any state, and H is the halting state.

Question 3

Describe what the following Turing machine will do, based on what you can see from its following rule transition table:

symbol,	state	->	new symbol,	new state,	head movement (1 is right, -1 is left)
0	1	->	1	1	1

1	1	-> 0	1	1

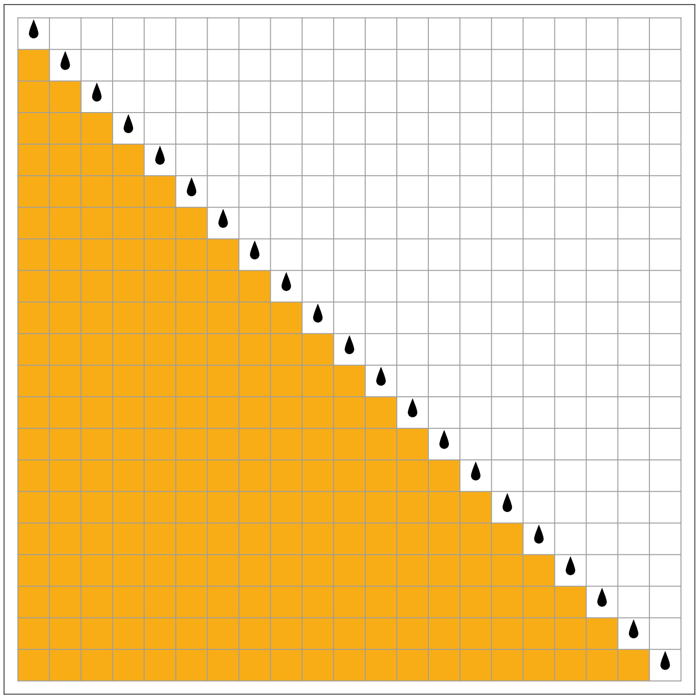
Answer 3

It will reverse the input. If the input is all 1s, it will change it to all 0s. Also the head always moves to the right.

```
In[5]:= rules = {{1, 0} → {1, 1, 1}, {1, 1} → {1, 0, 1}};

In[6]:= RulePlot[TuringMachine[rules], {1, {{}, 0}}, 20, Mesh → All]
```

Out[6]=



Question 4

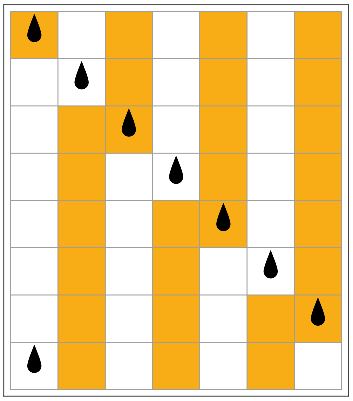
What the above Turing machine would do to the string “1010101”?

Answer 4

It would reverse it:

```
In[7]:= RulePlot[TuringMachine[rules],
         {1, {1, 0, 1, 0, 1, 0, 1}}, 7, Mesh → All, ImageSize → Small]
```

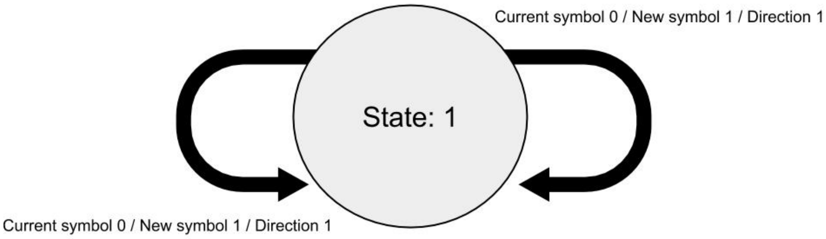
Out[7]=



Question 5

Write the previous Turing machine as a transition diagram.

Answer 5


Current symbol 0 / New symbol 1 / Direction 1
State: 1
Current symbol 0 / New symbol 1 / Direction 1

Question 6

Is the above Turing machine universal?

Answer 6

No, it only runs in one direction, it has only 1 state, and it has no halting state. These are many reasons why it cannot be universal.