

Wordle and Entropy (Preliminary)

Tom Davis

tomrdavis@earthlink.net

<http://www.geometer.org/mathcircles>

November 8, 2025

Abstract

In this article we will consider how to use information theory and entropy to find optimal strategies for playing the game **Wordle** that appears daily in the New York Times.

1 What is Wordle?

Wordle is a word game played on the New York Times (NYT) website.

1.1 Rules of the Game

Wordle is a word game that uses five-letter English words. The goal is to figure out a secret word by making a series of guesses. After each guess, the player learns three things:

1. Which letters are exactly right — both the letter and its position.
2. Which letters are in the secret word but in the wrong position.
3. Which letters don't appear in the secret word at all.

To show this information, each letter in the guess is colored:

1. Green means the letter is in the correct position.
2. Yellow means the letter is in the word but in a different position.
3. Black (or gray) means the letter isn't in the word.

If the guess is completely correct (all five letters are green), the game ends. Otherwise, the player keeps guessing — up to six times in total. The goal is to find the hidden word in as few guesses as possible. Here's an example game where the secret word is **BAGEL**:

T	R	E	N	D
S	L	I	M	E
B	E	L	L	Y
B	O	W	E	L
B	A	G	E	L

The first guess was **TREND**. None of its letters are in the same position as in **BAGEL**, but it does contain an **E**. Since the **E** is not in position 4, it is colored yellow. The other four letters (**T**, **R**, **N**, and **D**) do not appear in **BAGEL**, so they are colored black. This sequence of black-black-yellow-black-black will be listed here as **BBYBB**. The pattern for **BELLY**, below, would be **GYBB**.

The next guess, **SLIME**, tries four new letters but puts the **E** in a different spot. Again, none of the letters are in the correct position, but this time we learn that **L** is somewhere in **BAGEL**, but not in position 2.

After the guess **BELLY**, we now know that **B** is in the correct position so it is painted green. Because the **E** and **L** are still misplaced, we can infer that the secret word ends with **EL**. Note that the second **L** is painted black, so that means there is exactly one **L** in the solution. That makes the next guess, **BOWEL**, a logical choice. Finally, the word **BAGEL** solves the puzzle.

The player wins because the secret word was discovered in 5 guesses and up to 6 guesses are allowed.

In this article we will assume that the secret word and guesses come from a fixed set of words and that each time the game is played, the secret word is chosen at random from the set with equal probability.

At any stage as we play the game, there is a list of possible solutions and when we make a guess, the outcome is simply a five-color pattern that tells us how our guess matched with the actual solution. If the outcome was not five greens, then the solution must be another word that produces exactly the same outcome as our guess. All words that would have produced a different outcome are eliminated as possible solutions.

1.2 New York Times Complexity

The daily version of **Wordle** in the New York Times (NYT) is more complicated:

1. Human NYT editors choose the words so the secret words are not completely random.
2. There are about 2300 possible secret words. These tend to be very common English words, with very few plurals (5-letter words ending in **S**) and very few in the past tense (words ending in **D** or **ED**).
3. The NYT has two lists of words. Guesses can be made with words from the second list, which contains many of the omitted plurals, past-tense, foreign words, et cetera. There is some penalty for using a word in the second list, because there is zero chance that it is the solution. On the other hand, sometimes choosing an impossible word yields the most information.
4. The NYT never re-uses a word. The list of possible NYT words is about 2300 words long, and at this writing more than 1500 games have been played so in reality, today there are only about 750 possible words.
5. When the NYT purchased **Wordle** they edited out some “objectionable” words like **SLAVE** and **ABORT**, even though some of them are pretty common in general English usage.

1.3 Strategies

In this article we will try to find a guessing strategy that minimizes the expected number of guesses required to obtain a solution. Here are some ideas that may prove useful:

1. In some sense, the main purpose of each guess is to maximize the amount of information obtained from the computer’s reply. To do this, we need to have a good idea of what “information” means.
2. Especially early in the process, the secret words are selected at random, so likely words will contain more common letters in English. **Wordle**’s list of possible words contains fewer copies of the letters **S** and **D** because of the lack of plurals or past-tense words, so letter frequencies will be slightly different from those in standard English¹.

¹I do not have the current dictionary that the NYT uses for possible **Wordle** secret words, but I have a very close approximation and it yields: EAROTLISNCUYDHPMGBFKWVZXQJ.

3. Usually it is a good idea to choose a possible word as your guess but sometimes that is not a good idea. Consider the following game:

W	O	U	N	D

At first, this looks great! We have 4 of 5 letter slots already filled. Unfortunately, after guessing **WOUND** there are 7 possibilities: **BOUND**, **FOUND**, **HOUND**, **MOUND**, **POUND**, **ROUND**, and **SOUND**. If we insist upon guessing a possible solution, it might take up to 7 guesses (and we have already used one), so basically $2/7$ of the time we will fail, and even if we succeed, we will need 2, 3, 4, 5, or 6 guesses, so an average of 4.0 guesses.

A better strategy is to guess something like **FROSH** and if any of **F**, **R**, **S** or **H** is in it we get the answer in 3 guesses. if there are no hits among those letters, try **BUMPY** which covers the final possibilities, and in this case the answer requires 4 guesses. The average will be $3(4/7) + 4(3/7) = 24/7 \approx 3.42$, and there is no chance of failure.

It turns out that maximizing the amount of information at each stage yields very good results, and we can measure how good a response is by calculating the entropy of the resulting situation after testing every one of the possible guesses.

The next section will provide, without proof, the calculations necessary to determine the entropy. After that we will look at the concept of entropy carefully to convince ourselves that the entropy described is very close to what we need.

2 Similar Games

A lot of games have the following form:

- There are two players. One selects an item chosen from a finite set of possibilities and keeps the choice secret from the other player.
- The other player (the questioner) asks a series of questions whose answers eliminate some of the possible choices.
- The questioner wins if the secret item is determined within a certain number of guesses; otherwise the other person wins.

In **Wordle** the secret item is a 5-letter word and the guesses are all 5-letter words from a particular list. The response to the question is the five-color pattern that shows the relationship of the guess to the secret word. If the answer is five greens (**GGGGG**) within 6 guesses, the questioner wins.

A common children’s game is called “Twenty Questions” where the secret item is an animal and the questioner has to make up to 20 yes-no questions to determine the secret. The “item” is an animal and the goal is to guess the animal’s name in fewer than 20 questions.

The game of **Mastermind** is another example that is quite similar to **Wordle**. Instead of 5-letter English words, the items are lists of four colors chosen from a set of six. Thus there are $6^4 = 1296$ possible color-words, and the response to each color-word guess is an indication of how many colors match the secret perfectly and how many are correct, but in the wrong position. The difference is that the guesser only gets to know how many colors are in each category and not their positions.

For example, if the secret pattern is red-green-red-yellow and the guess is green-green-blue-red the response would be: one color perfectly correct (the second green) and one color correct, but in the wrong position (the final red of the guess).

Let's consider a much simpler version of 20 questions. Suppose there are only 8 possible animals. Since we can only ask yes-no questions and if we can only ask n of them we can only distinguish among 2^n possibilities. To distinguish among 8 animals we need at least 3 questions since $2^3 = 8$.

The only way to guarantee success within 3 guesses is to make each guess divide the set of remaining possibilities exactly in half. Suppose the animals are numbered 1 through 8. The first guess might be, "is it 4 or less?" A "yes" means the animal is 1, 2, 3, or 4 and a "no" means it's 5, 6, 7, or 8. Depending on the answer, the next guess might be "Is it 2 or less?" or "Is it 6 or less?" The final question only has to distinguish 2 possibilities.

If the questions do not split the possibilities exactly in half there is no guarantee of getting the solution in 3 guesses. If it is not split 4-4 then one set of possibilities will have more than 4 items, and with just 2 guesses left, there is no way to distinguish among 5 or more possibilities.

If the questions can have 3 possible answers, then with n questions, we can distinguish 3^n possibilities, with 4 possible answers, 4^n , and so on.

If each question has m possible answers ideally the question should divide the set of possible answers into m equally-sized groups. Intuitively, the closer the division is to equally-sized piles, the faster the answer can be obtained.

2.1 Wordle guess choice

In **Wordle** the secret word and the guesses are just 5-letter words, but the response to each guess is a sequence of 5 colors chosen from green, yellow, and black. In fact, almost any sequence can occur except for combinations of four greens and one yellow, for a total of $3^5 - 5 = 238$ possible responses.

Each guess can be analyzed as if each of the remaining possible words were the correct answer to see how the responses are divided into the 238 possible response piles. Intuitively, we want to make a guess that, in some sense, makes the piles as uniform in size as possible. In other words, good guesses smear out the possible solutions as much as possible. It will turn out that the mathematical concept of entropy is exactly what we need. The choice that makes the entropy as large as possible is the choice that divides the responses into piles that are spread out as much as possible.

3 What is Entropy?

In a situation with various possible outcomes with various probabilities, it is possible to compute a number from those probabilities called the "entropy." It will turn out that in a sense, the entropy is a measure of the amount of information in the situation.

In **Wordle** the original situation has all possible solutions equally-likely. Each time a guess is made, the list of possible solutions is reduced, producing a new situation. A good strategy is to calculate the entropy of situations resulting from every possible guess and choose the guess that yields the most entropy (or information).

To be a bit more precise, the outcome of a guess is a color pattern and that effectively eliminates all words that do not produce that color pattern with that guess.

4 An Entropy Cookbook

Imagine a situation where there is a certain finite set of outcomes and a probability associated with each outcome. When we make a guess, the possible outcomes of that guess are patterns of 5 colors. The outcome we are seeking is to obtain the pattern of five greens. Suppose there are k possible outcomes and the probability of outcome i is p_i . The following is

true since there are exactly k possible outcomes:

$$\sum_{i=1}^k p_i = p_1 + p_2 + \dots + p_k = 1.$$

The entropy $H(p_1, \dots, p_k)$ of the situation is given by the following formula with the understanding that if any of the p_i are zero, then the terms $p_i \log_2 p_i$ will also be zero:

$$H(p_1, \dots, p_k) = -(p_1 \log_2 p_1 + p_2 \log_2 p_2 + \dots + p_k \log_2 p_k) = -\sum_{i=1}^k p_i \log_2 p_i.$$

We will see why the entropy of a situation can be interpreted as the amount of information in the situation. The base of the logarithms can be anything, but if the base is 2 then the entropy measures the amount of information in bits.

As an example, imagine a situation with four equally-likely outcomes that we'll call O_1, O_2, O_3 , and O_4 , and $p_1 = p_2 = p_3 = p_4 = 1/4$. The entropy is given by:

$$H = -(0.25 \log_2(0.25) + 0.25 \log_2(0.25) + 0.25 \log_2(0.25) + 0.25 \log_2(0.25)) = 2.$$

The value $H = 2$ means that 2 additional bits of information are required to determine which of the four outcomes occurred. If we assign the bit pairs 00, 01, 10, and 11 to O_1, O_2, O_3 , and O_4 , respectively. In this situation, all four patterns of 2 bits are used, and they should occur equally likely

Let's consider a different set of probabilities. Suppose $O_1 = 1/2, O_2 = 1/4$, and $O_3 = O_4 = 1/8$. The entropy in this situation is:

$$H = -(0.5 \log_2(0.5) + 0.25 \log_2(0.25) + 2 \cdot 0.125 \log_2(0.125)) = 1.75$$

This means that this situation only requires 1.75 bits of information to determine the outcome. What do we mean by fractions of a bit? Consider the following encoding²:

O_1	0
O_2	10
O_3	110
O_4	111

Is it obvious why this scheme works?

What is the average number of bits required to determine the outcome? Half the time we determine that the outcome is O_1 with just one bit. One quarter of the time two bits are required, and one eighth of the time three bits are required both of O_3 and O_4 . The average number of bits required is:

$$0.5 \cdot 1 + 0.25 \cdot 2 + 0.125 \cdot 3 + 0.125 \cdot 3 = 1.75.$$

4.1 Is it really 1.75 bits?

Imagine sampling from both versions: one with equal probabilities of the four outcomes and the other with the probabilities $\{0.5, 0.25, 0.125, 0.125\}$

It is also interesting to note that if we look at a long stream of these four messages, encoded as 0, 10, 110 and 111, then (assuming the messages were sent at random in accordance with the probabilities above) the stream of 0's and 1's produced would appear to be completely random. If the encodings had been the "obvious" 00, 01, 10, and 11 there would be far too many zeroes.

²This is called a Huffman encoding.

To see this, here is the result of taking the same 200 random samples with the above probabilities but using the two encoding schemes. This first is with the obvious inefficient encoding of exactly two bits per sample (400 total bits):

```
1100000000000100001000101000011110000011100010100
01001000110000000000000001000111000000010001000001
00000100010001010011000001000001000001010001000100
10110000010010010101000001000001000000001001001000
01001101001000011000001100110000101100010101000000
00000010001001000111011100110110110000001000111110
01001111101100000010000001001010010110100000000001
10101111000000000011000001110100001100001000001100
```

and this is the transmission using the efficient encoding (346 total bits):

```
11100000011001001010001111110010111010100100110011
10000000100101110001001000100010010010100111001000
10001010010010011011100100110101010001000100000110
10011001001111001100101100011101110011011101010100
0000011001101001011101110111011101110001100111111
1101001111111011100011000100110110101011011000001
011011011111100000111001011100011100110001110
```

Notice that the first stream has way more zeroes in it and also notice that it uses 400 bits (as it must: two bits per message) but the second stream uses 346 bits which is quite close to the expected number: $1.75 \cdot 200 = 350$.

In the second stream there are 173 of the 0 bits which happens to be exactly half of the total number (346) of bits. In the first stream, there are 277 of the 0 bits and 123 of the 1 bits.

The outcomes above will be typical for very good encodings: a good encoding will generate a stream that seems to be completely random with equal probabilities of both bits and it will be shorter, on average, than streams generated by other encodings.

In fact the encoding for this set of probabilities of messages is “perfect” in the sense that it is impossible to do better in the long run with messages having the given probabilities of occurring.

4.2 Properties of Entropy

It turns out that the entropy is maximized when all the probabilities are the same. The more lopsided the distribution of probabilities, the smaller the entropy. Consider the extreme (most lopsided) distribution where $p_1 = 1$ and all the other probabilities are 0. Before any information about this situation is passed, we *know* the result: O_1 . Thus zero additional information is required to determine the outcome of this situation.

Here are a few values of the entropy of a two-outcome situation for various values of p_1 and p_2 . It’s easy to see that the more lopsided the probabilities, the smaller the entropy:

$$\begin{aligned} H(0.5, 0.5) &= 1.00000 \\ H(0.51, 0.49) &\approx 0.99971 \\ H(0.75, 0.25) &\approx 0.81128 \\ H(0.99, 0.01) &\approx 0.08079 \\ H(0.999, 0.001) &\approx 0.01141 \end{aligned}$$

Here are similar results when we calculate the entropy with three outcomes. Note that the results are given as the number of bits of information. If one is zero, the result is the same as the entropy function with two outcomes.

$$\begin{aligned}H(0.333333, 0.333333, .33334) &\approx 1.58496 \\H(0.3, 0.3, .4) &\approx 1.57095 \\H(0.1, 0.4, .5) &\approx 1.36096 \\H(0.4, 0.5, .1) &\approx 1.36096 \\H(0.05, 0.05, .9) &\approx 0.56900 \\H(0.0, 0.5, 0.5) &= 1.00000\end{aligned}$$

The important properties of the entropy function H can be found in Appendix A.

5 Using Entropy in Wordle

How can entropy be used to try to solve **Wordle** puzzles efficiently?

For now let's consider a simple version of **Wordle** where all of the possible words are in a single list that is available to the questioner³. In the original version of **Wordle** this list contained about 2300 words. We also assume that the secret word is chosen at random from this list with a uniform distribution. In other words, humans do not select words, and just because a word has already been used, it is still equally-likely to be selected as an unused word.

At every stage of a **Wordle** game the human has a list of words that have not yet be eliminated as well as a set of words that can be used as a guess. (It is usually a good idea to choose the guess from the list of possible solutions, but as we see in Section 1.3 it is not always the case.)

Intuitively, a good guess should leave a situation that has the largest value of entropy as possible so for each possible guess we need to calculate the entropy that would result as a result of that guess.

Here is an algorithm to calculate the entropy for a particular word G as a guess. When that guess is compared to any possible solution, the computer would return a list of 5 colors. There are $3^5 = 243$ sequences of 5 colors and We need for a data structure that associates of a list of words for each of the possible color patterns⁴.

1. Empty all the lists in the data structure.
2. For every possible solution word S :
 - (a) Calculate the color pattern obtained if G were the guess and S were the solution.
 - (b) Add S to the data structure list corresponding to that color pattern.
3. At this point every possible solution is found in one of the color pattern lists. Since we assume that all the possible solutions are equally-likely the relative probabilities of a solution having that color pattern is proportional to the number of solutions in that list.
4. If there are k possible solutions and there are m of them that are in a particular color pattern list, then the probability of being in that list is simply m/k .
5. We now have a probability of the solution being in each of the different color pattern lists, and the entropy can be calculated and associated with the guess G .

³In the NYT version **Wordle** has two word lists. One is of more common words that are possible solutions, and the other contains words that can be used as guesses, but are not possible solutions. In fact, the possible **Wordle** solutions contain almost no plural words, so the set of possible guesses includes lots of four-letter words followed by an "s" to make them plural.

⁴Although there are 243 possible color patterns five of them are impossible: those with four greens and one yellow,

The entropy is calculated for each possible guess, and the one with the highest entropy is selected since it will leave us in a situation with the most entropy.

As soon as we see the result, we need to get rid of any possible guesses that were eliminated by that result, and then we can start over with a reduced set of possibilities.

5.1 A Complete Example

Suppose you are playing and your first guess is the word **GUESS**. The computer responds **YBGB**, and there are 9 words that return that result to an initial guess of **GUESS**, and they are **BLUSH**, **CRUST**, **PLUSH**, **JOUST**, **LOUSY**, **BRUSH**, **CRUSH**, **TRUST**, and **FLUSH**. We would like to find the best next guess among these 9 words.

If each word is tested against each word, the results are shown in this grid. In the grid the first column corresponds to the case where **BLUSH** is the guess and the entries in the column are the results of guessing the 9 possibilities against **BLUSH**:

	BLUSH	CRUST	PLUSH	JOUST	LOUSY	BRUSH	CRUSH	TRUST	FLUSH
BLUSH	GGGGG	BBGGB	BGGGG	BBGGB	YBGBB	GBGGG	BBGGG	BBGGB	BGGGG
CRUST	BBGGB	GGGGG	BBGGB	BBGGG	BBGGB	BGGGB	GGGGB	BGGGG	BBGGB
PLUSH	BGGGG	BBGGG	GGGGG	BBGGB	YBGBB	BBGGG	BBGGG	BBGGB	BGGGG
JOUST	BBGGB	BBGGG	BBGGB	GGGGG	BGGGB	BGGGB	BBGGG	BBGGG	BBGGB
LOUSY	BYGGB	BBGGB	BYGGB	BGGGB	GGGGG	BBGGB	BBGGB	BBGGB	BYGGB
BRUSH	GBGGG	BGGGB	BBGGG	BBGGB	BBGGB	GGGGG	BGGGG	BGGGB	BBGGG
CRUSH	BBGGG	GGGGG	BBGGG	BBGGB	BBGGB	BGGGG	GGGGG	BGGGB	BBGGG
TRUST	BBGGB	BGGGG	BBGGB	BBGGG	BBGGB	BGGGB	BGGGB	GGGGG	BBGGB
FLUSH	BGGGG	BBGGB	BGGGG	BBGGB	YBGBB	BBGGG	BBGGG	BBGGB	GGGGG

Each row above are the possible color patterns that **Wordle** would reply in response to each possible guess. Some are duplicates. Here is a grid that shows the number of times each pattern appears:

Pattern	BLUSH	CRUST	PLUSH	JOUST	LOUSY	BRUSH	CRUSH	TRUST	FLUSH
GGGGG	1	1	1	1	1	1	1	1	1
BBGGB	3	4	3	5	4	2	2	4	3
BGGGG	2	1	2	0	0	1	1	1	2
BBGGG	1	1	2	2	0	2	3	1	2
BGGGB	0	1	0	1	1	2	1	2	0
YBGBB	0	0	0	0	3	0	0	0	0
BYGGB	1	0	1	0	0	0	0	0	1
GBGGG	1	0	0	0	0	1	0	0	0
GGGGB	0	1	0	0	0	0	1	0	0

Next calculate the entropy of each column:

Pattern	BLUSH	CRUST	PLUSH	JOUST	LOUSY	BRUSH	CRUSH	TRUST	FLUSH
GGGGG	1/9	1/9	1/9	1/9	1/9	1/9	1/9	1/9	1/9
BBGGB	3/9	4/9	3/9	5/9	4/9	2/9	2/9	4/9	3/9
BGGGG	2/9	1/9	2/9	0	0	1/9	1/9	1/9	2/9
BBGGG	1/9	1/9	2/9	2/9	0	2/9	3/9	1/9	2/9
BGGGB	0	1/9	0	1/9	1/9	2/9	1/9	2/9	0
YBGBB	0	0	0	0	3/9	0	0	0	0
BYGGB	1/9	0	1/9	0	0	0	0	0	1/9
GBGGG	1/9	0	0	0	0	1/9	0	0	0
GGGGB	0	1/9	0	0	0	0	1/9	0	0
Entropy	2.4194	2.2810	2.1972	1.6577	1.7527	2.5033	2.4194	2.0588	2.1972

Our next guess should be the one with the highest entropy; namely, **BRUSH** with an entropy of 2.5033 bits.

6 Information

Consider a situation with two outcomes, O_1 and O_2 with probabilities p_1 and p_2 and without loss of generality, assume $p_1 \geq p_2$. The larger p_1 is, the less information you have about the situation. This is easy to see by looking at extremes: if $p_1 = 1$ then O_1 is certain to occur, so zero additional information is required to know the outcome.

Similarly, low probabilities are associated with more information, so perhaps a useful approximation for the amount of information with probability p is just $1/p$.

This is close, but not quite right. Here's why. Suppose two outcomes are sampled from the same situation, and the outcomes have probabilities p and q (and that all the samples are independent in that none depend on the previous sample). Since the samples are independent events, the probability of getting the two samples is pq (the product of the probabilities). Intuitively the total amount of information received from the two samples should be the sum of the individual amounts of information, so if there is a function $I(p)$ that means, "the amount of information received when a message of probability p arrives", then $I(p)$ should satisfy the following equation:

$$I(pq) = I(p) + I(q).$$

This looks exactly like a logarithm function, and in fact, that's what it is:

$$I(p) = \log(1/p).$$

Notice that:

$$I(pq) = \log(1/(pq)) = \log(1/p \cdot 1/q) = \log(1/p) + \log(1/q) = I(p) + I(q).$$

It also has the nice property that the information associated with an event that is certain to occur (in other words, $p = 1$) is zero, since $\log(1/1) = \log(1) = 0$.

We did not specify the base of the logarithm above, and in fact, logarithms of any base would behave in exactly the same way. For reasons that will become clear later, we will usually use the logarithm base-2, so:

$$I(p) = \log_2(1/p) = -\log_2(p).$$

Now let's ask a *very* interesting question. If an experiment or situation plays out, what is the average expected amount of information we will receive as a result? For each of the possible outcomes, we need to multiply the probability of that outcome by the amount of information that becomes available as a result, and that will be the expected amount of information received. For our experiment with outcomes having probabilities p_1, p_2, \dots, p_k , that value will be:

$$-(p_1 \log_2 p_1 + p_2 \log_2 p_2 + \dots + p_n \log_2 p_k) = -\sum_{i=1}^k p_i \log_2 p_i.$$

And that is exactly the expression we use for entropy.

6.0.1 A Formula for H

We will not prove this, but it turns out that there is basically only one type of function that satisfies all of the conditions above, and the total uncertainty, called the "entropy," has the following form:

$$H(p_1, p_2, \dots, p_k) = -(p_1 \log p_1 + p_2 \log p_2 + p_n \log p_k) = -\sum_{i=1}^k p_i \log p_i.$$

The logarithms above can be taken to any base, but in information theory, it is customary (and we'll see why later) to use logarithms base-2. If we used other bases, the resulting values of H would simply be multiples of each other. The negative sign makes sense because all probabilities are at most one, so the logarithms are all zero or negative⁵.

Notice that this formula for H is identical to the formula near the end of Section 6! So another way to think of entropy is as the average amount of information received per outcome/message/event.

⁵It turns out that if we use base-2 the entropy will tell us the optimal average number of bits required to send an average message. Had we used base-3, it would tell us the average number of symbols in a three-symbol encoding that would be required, et cetera

In the expression above, we will also assume that if some probability p_i is zero, then the term $p_i \log p_i$ is also zero in spite of the fact that the logarithm of 0 is undefined.

It is worth spending a bit of time checking so see that the conditions for an uncertainty measure are satisfied by that formula. Some of the checks, for example, the fact that the entropy is maximized when all the probabilities are equal, requires a bit of calculus, but you can check with some actual numbers to convince yourself, even without calculus, that it is the case. Look at the examples in Section 4.2.

A Properties of the Entropy Function

In fact, the following lists (without proof) most of the important properties of the entropy function H .

1. If there is no uncertainty; namely, if there is only one possible outcome, then the value of H should be 0. The uncertainty of any event should be non-negative, and if situation A is less-predictable than situation B , the uncertainty of A should be larger than that of B .
2. The order of the parameters should make no difference. That is, it shouldn't matter which outcome we name as the first, second, third, et cetera. Mathematically, it means:

$$H(p_1, p_2) = H(p_2, p_1)$$

for two possible outcomes, and

$$\begin{aligned} H(p_1, p_2, p_3) &= H(p_1, p_3, p_2) = H(p_2, p_1, p_3) \\ &= H(p_2, p_3, p_1) = H(p_3, p_1, p_2) = H(p_3, p_2, p_1) \end{aligned}$$

for three outcomes, et cetera. Mathematical functions where the order of the parameters is unimportant are called *symmetric functions*. Another way of stating this idea is to say that $H(p_1, p_2, \dots, p_k)$ is symmetric. We can mathematically state this for the general case as follows: Let $\pi(i)$ be a permutation (a rearrangement) of the set $\{1, 2, 3, \dots, k\}$. Then for any such π we have:

$$H(p_1, p_2, p_3, \dots, p_k) = H(p_{\pi(1)}, p_{\pi(2)}, p_{\pi(3)}, \dots, p_{\pi(k)}).$$

3. When there are k possible outcomes, the situation that is the most uncertain is when all the p_i are equal. If it's more likely that one outcome will occur than another, then there is less uncertainty. If two equally-able football teams are playing, you are quite uncertain about the outcome, but if the San Francisco 49ers are playing against your high-school football team, there is very little uncertainty about what will happen.

We can state this mathematically as follows: For any set of $p_i \geq 0$ such that $p_1 + p_2 + \dots + p_k = 1$ then:

$$H(1/k, 1/k, 1/k, \dots, 1/k) \geq H(p_1, p_2, p_3, \dots, p_k).$$

4. Similarly, if you have more equally-likely outcomes, the uncertainty increases:

$$H(1/k, 1/k, \dots, 1/k) \leq H(1/(k+1), 1/(k+1), \dots, 1/(k+1)),$$

(where the first H has k parameters and the second, $k+1$).

5. Adding an outcome to the list of possible outcomes which has zero probability of occurring will not change the value of H . In other words, it will have no effect on the uncertainty. Mathematically:

$$H(p_1, p_2, \dots, p_k) = H(p_1, p_2, \dots, p_k, 0),$$

(where the first H has k parameters and the second, $k+1$).

6. If there are two completely independent situations, the first with outcomes having probabilities p_1, p_2, \dots, p_k and the other having probabilities q_1, q_2, \dots, q_m then when both situations have occurred, there are km possible outcomes, and since the situations are independent, the probabilities of these km outcomes will be:

$$p_1q_1, p_1q_2, \dots, p_1q_m, p_2q_1, p_2q_2, \dots, p_2q_m, \dots, p_kq_1, \dots, p_kq_m.$$

The total uncertainty should be the sum of the two uncertainties, so:

$$H(p_1q_1, p_1q_2, \dots, p_kq_m) = H(p_1, p_2, \dots, p_k) + H(q_1, q_2, \dots, q_m).$$

7. H should be continuous in its variables. In other words, if one set of probabilities is very close to another set, then the values of H for those two sets should also be close. (The standard calculus definition of continuity applies, if you know what that is.)
8. Here is a final condition that's a little more complicated. First we'll state the result mathematically and then try to describe it.

Suppose $p_i \geq 0, q_i \geq 0, p = p_1 + \dots + p_n, q = q_1 + \dots + q_m$, and $p + q = 1$. Then:

$$\begin{aligned} & H(p_1, \dots, p_n, q_1, \dots, q_m) \\ = & H(p, q) + H(p_1/p, p_2/p, \dots, p_n/p) + H(q_1/q, q_2/q, \dots, q_m/q). \end{aligned}$$

In English, this basically says that if a situation with $n + m$ outcomes is divided into two classes, one having total probability p and the other, total probability q , then the total uncertainty is the sum of three things: the uncertainty of whether the first or the second class occurs, the uncertainty of the result if we were restricted to only the first class and the uncertainty of the result if we were restricted to only the second class. This is a little technical, so don't worry if it is not at first obvious.