cf. earlier document: thoughts about an algorithm for generating all possible words in a language

From all possible words to a reduced subset: restrictions on patterns
In natural languages not all sounds can follow each other. It is not impossible to pronounce the consonant cluster /kkk/, but such a cluster of consonants never occurs in EN.

For Fomo (meaning "sound"), i went with the fonemes C=[m, p, t, k, f, s] and V=[a, e, i, o, u], a total of 11 fonemes. With unrestricted foneme patterns, this results in a huge number of possible worlds. Words with length 1, 11 possibilities; 2, 121; 3, 1331; 4, 14641; 5, 161051. A staggering number.

Supposing that one reduces this to some subset by getting rid of words with consonant clusters that are near impossible to pronunce, say /kptkk/. For a simple to use rule, one can avoid consonant clusters all together. To do this, one can find all consonants in the word representation, and see if they have a consonant on the right or on the left. If they have, remove the word from the list. Many languages have exactly such a rule.
Another rule is to avoid having consonants that have no adjacent vowel. In such a case words such as /kukku/ er possible, which they aren't with the first rule since /kk/ is a consonant cluster.

I wrote the code to remove the words with invalid patterns. Here are the results for *Kuk* and *Fomo*.

Kuk

Length	**Words without invalid patterns out of total possible**
1	2
2	3 of 4
3	5 of 8
4	9 of 16
5	17 of 32

Notice the pattern. It is easy to calculate with this language. $|V| = 1 + \frac{|A|}{2}$, where V is the set of valid possible words, and A is the set of all possible words. |A| is also straightforward, since it's just $2^{L-1}$where L is wordlength.
This is the relation between |V| and |A| tho, not between wordlength and |V| or |A|.

Fomo

Length	**Words without invalid patterns out of total possible**
1	11 of 11
2	85 of 121
3	755 of 1331
4	7225 of 14641
5	74075 of 161051
6	723205 of 1771561

The reason that there are 11 out of 11 in the first case, is that no rules currently designate words consisting only of consonants as invalid.

Given the easy patterns from before, i wonder if the same or similar is true for *Fomo*. I plotted the data series from the language. Then i had the computer guess the function. Two functions yielded a perfect fit (R^2=1): a linear and a polynomial. (Well, perfect according to the program, it probably rounded the result.) Then, i had the program calculate the number of possible words with 6 letters. Meanwhile, i used the functions from before to predict the number of valid possible words. It didn’t fit. So the pattern the computer is finding is just *ad hoc*. Still there remains the possibility that it is mathematically solvable in some clever way.