The formalist point of view

April 3, 2018

Contents

1 The formalist point of view

Very concisely the formalist believes: "Everything is a formal language".
"The limits of my language mean the limits of my world." Ludwig Wittgenstein
There are a lot of "formalist views" of the world/of mathematics/of art/of the mind, but before diving into these views I want to deliver a solid understanding of the concepts **formal language**, **formal systems**, **computation** and **formal logic**.

2 Formal language

Conventionally the theory of formal languages is defined with set-theory, however do not assume that the theory of formal languages is based on set-theory. The theory of formal languages is perfectly independent and, as we will later see, can even describe set-theory. This is why I will talk about collections, but will use a similar notation due to

Alphabet Any finite non-empty collection Σ can be called an alphabet. The elements of Σ are called **symbols** or **characters**.

For example $\Sigma_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and $\Sigma_{lat} = \{a, ..., z\}$. These characters are sometimes surrounded by single quotes: 'a'

String A string or word is a finite sequence / a finite list of symbols over an alphabet. 73 is a string over Σ_{10}. These strings are sometimes surrounded by double quotes: "73". There is also the **empty string** "", which is sometimes denoted as ϵ or λ.

All possible strings Σ^* is the infinite collection of all possible strings over an alphabet (including the empty string).

Formal language A formal language L consists of an alphabet Σ and a notion of "belonging to the language". It **divides** Σ^* into two collections: The strings that belong to the language L and the strings that do not belong to the language L.

For example I can define L to be the language of all words over the alphabet $\Sigma = \{0, 1\}$, which contain an equal amount of zeros and ones.

Entscheidungsproblem Given a string x and a formal language L, the problem of deciding whether x belongs to L (or not) is called the Entscheidungsproblem.

For example "0101" would belong to the previously mentioned language L.
Instead of saying x belongs to L you can also state:
x is a **well-formed formula** of L.
x is **syntactically valid formula** of L.
$x \in L$
x is **matched by** L
For natural languages: x is **grammatically valid formula** of L.

2.1 Some languages

Now I hope to give some intuition on my radical statement that everything is formal, by giving examples of (somewhat) formal languages.

- $L =$ the base-2 representation of all prime numbers over $\Sigma = \{0, 1\}$.
- $L =$ the binary encoding of all JPG images, which Pewdiepie would consider to be a meme over $\Sigma = \{0, 1\}$.
- $L =$ the language of all valid propositional statements (of all tautologies) over $\Sigma = \{A, B, \vee, \wedge, \neg, \rightarrow\}$.
- $L =$ the coordinates of all elementary particles relative to me, which are encoded in the following way: Their relative x, y, z coordinates in meters is rounded to the precision of $10^{-100}m$. The binary integer representation of these rounded coordinates in meters multiplied by 10^{-100} is then stored as "x:y:z" over $\Sigma = \{:, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ (this ignores relativistic effects and doesn't define the direction of the axis, but you get the point).
- $L =$ all sentences, which Alice and Bob would both consider to be grammatically correct English over $\Sigma =$ computer-representable characters.
- $L =$ Every sentence Alice has ever muttered, which Bob would consider to be grammatically correct English over $\Sigma =$ computer-representable characters.

What makes a language "formal" is its exclusive membership. Either $x \in L$ or $x \notin L$, but not both. In set-theory you would define $L \subseteq \Sigma^*$, which would entail binary membership.

For example in English ":) walks to Schaffhausen" might by some be considered grammatically correct English, since they view the ":)" as a smiley, which they interpret as a valid subject. On the other hand some other people would reject the smiley as a subject. In this sense English is not a formal language. However in my view you can always find some common subset of English, for which two parties both handle the Entscheidungsproblem (and therefor the grammar) unambiguously or we might just learn a language like Lojban.

2.2 Meaning is syntax

Take the language L of all propositional formulas over the alphabet $\Sigma = \{A, B, \wedge, \vee, \neg, \rightarrow\}$. Now you might not only be interested in whether a string over Σ^* is a syntactically valid propositional formula or not, you might also be interested in their propositional validity (whether they are a tautology or not). Instead of reasoning over L, we can simply change the language L to be all propositional formulas over the alphabet $\Sigma = \{A, B, \wedge, \vee, \neg, \rightarrow\}$, which are tautologies. Suddenly propositional truth shifts to another instance of the Entscheidungsproblem.

In a similar vain you might define L to be all binary encodings of JPG images over $\Sigma = \{0, 1\}$. Now you might want to rank these images according to your own metrics of beauty. Even here you can define L' to be the language consisting out of words of the form x, y, z, where x is the binary encoding of the JPG image and y, z encode binary numbers, which measure how beautiful you find x, so $\frac{y}{z} \in [0, 1]$.

The formalist now believes in a very precise sense, that everything is a formal language. The formalist believes that there is no problem outside of the Entscheidungsproblem and when we want to get some results from a formal language outside of this Entscheidungsproblem, what we really are doing is solving the Entscheidungsproblem for a different formal language.

In the last example I tried to define L', however this language does not really solve the Entscheidungsproblem in a binary manner (my tastes might change over time).

If you are a formalist and believe everything needs to be formal you can either attempt to formalise the notion of L' so that the Entscheidungsproblem becomes objective or state like the early Wittgenstein: "Whereof one cannot speak, thereof one must be silent." Ludwig Wittgenstein

So ultimately we shift from meaning to definability, which leads us to the next question: What formal languages can be defined and how?

2.3 Two problems of communication

- How can I describe a new formal language L to you?
- How can I know that you understand and therefore are able to apply the Entscheidungsproblem to this language L?

2.3.1 Translation

If we had already solved these problems for some other formal language M and M were expressive enough to describe L, I could then go ahead and describe the language L with M. Since I know you perfectly understand every sentence I utter in M, I know you then must understand every sentence I utter in L.

This is precisely what I attempted to do in the previous section, I tried to describe another language L using English.

So how did we learn English in the first place? Well one might say, that we learnt English from translating the language of our physical world into the language of our sensory organs and from there translated the language of our sensory organs into English. This rests on a lot of assumptions, which we will discuss in-depth later, however now I want to tackle the problem of translation. How do you define a formal language, with which you can define other formal languages?

2.3.2 Formal language 1: Greibach normal form (aka GNF)

L_{GNF} is a formal language over the alphabet

$\Sigma = \{$a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,0,1,2,3,4,5,6,7,8,9,',:,=,;,&$\}$.
A word belongs to L_{GNF} if it consists out of a sequence of one or more statements. A statement is of the form "LHS::=RHS;", where LHS (left-hand side) is a string of one or more lower-case latin characters and RHS (right-hand side) is a character embraced by single-quotes such as '0' optionally followed-by zero or more lower-case latin strings (which are separated by & symbol).
Additionally the first statement needs to have a LHS that's named l.

With this syntactic definition of L_{GNF} it should be enough for you to decide whether a string belongs to L_{GNF} or not, so you can solve the Entscheidungsproblem of L_{GNF}.

If you ignore line-breaks the following string is a word of GNF for example.

```
l::='1';l::='2';l::='3';l::='4';l::='5';l::='6';l::='7';l::='8';l::='9';
l::='1'&d;l::='2'&d;l::='3'&d;l::='4'&d;l::='5'&d;
l::='6'&d;l::='7'&d;l::='8'&d;l::='9'&d;
d::='0';d::='1';d::='2';d::='3';d::='4';
d::='5';d::='6';d::='7';d::='8';d::='9';
d::='0'&d;d::='1'&d;d::='2'&d;d::='3'&d;d::='4'&d;
d::='5'&d;d::='6'&d;d::='7'&d;d::='8'&d;d::='9'&d;
```

We already solved the Entscheidungsproblem for this language, so what else is there to say? Again we are talking about the meaning of these words in L_{GNF} and will later show how to reduce this to the Entscheidungsproblem.

So without further a do let me explain the meaning of a word $x \in L_{\text{GNF}}$ and therefor define how you can interpret x as a new formal language.

Every statement (LHS::=RHS) of x defines its own sub-language in the following way:
Statements of the form "LHS::='c';" match only the embraced character.
Statements of the form "LHS::='c'x&y&...&z;" match only words which start with c. The next few characters of that word need to be matched by any statement, which has x as a left-hand side. The next characters need to be matched by any statement, which has y as a left-hand side and so on. Lastly the remaining characters of the word need to be matched by a statement, which has z as a left-hand side. If no statement exists with such a left-hand side, it will match the word '0'.

We say that x defines the language L_l, which accepts all words that are accepted by any statement with left-hand side l in x.

Notice that every syntactically valid L_{GNF} formula corresponds to a formal language.

Now if you think about the previous example you will see that it accepts all positive integers. For example "13" and "10" are accepted, however "0" and "007" are not accepted.

L_{GNF} is even expressive enough to describe the syntax of itself:

```
l::='l'def&charlist;l::='l'def&charlist&stmt
x::='a';...;x::='z';
x::='0';...;x::='9';
x::=''';x::=':';x::='=';x::=';';x::='&';
```

```
lw::='a';...;lw::='z';
lw::='a'&lw;...;lw::='z'&lw;
equ::='=';col::=':';def::=':'col&equ;
semi::=';';quot::=''';
charlist::='''x&quot&list&semicol;
list::='&'&lw;list::='&'&lw&list;
stmt::='a'stmt;...;stmt::='z'stmt;
stmt::='a'def&charlist;...;stmt::='z'def&charlist;
stmt::='a'def&charlist&stmt;...;stmt::='z'def&charlist&stmt;
```

Notice that while this language defines the exact same formal language L_{GNF}, there is no way of describing the semantics of GNF using GNF. So where is the magic trick?

Let us define a new language $L_{\text{IGNF}} = x, y$, where x is a syntactically valid word of L_{GNF} and y is a word that is acceptable by the description of x (the way we reasoned about previously).

So while the syntax is the same, one of it also has a semantics defined and the other does not. So GNF is a formal language expressive enough to describe other formal languages. English on the other hand was somehow able to express both the syntax and the semantics of GNF.

GNF is fairly expressive. Notice that it can define all finite languages (that do not contain the empty string) by simply listing all words. For example the language $L = \{"\text{world}","\text{hi}"\}$ can be defined using:

```
l::='w'&o&r&d;
l::='h'&i;
a::='a';...;z::='z';
zero::='0';...;nine::='9';
```

3 Formal systems

In this section it should get more clear how mathematics can be thought of just another formal language.

3.1 Formal language 2: Peano axioms

$L =$ All valid statements that can What about proofs? Term rewriting systems can also be

3.2 Syntax = Semantics Nani?!?

Wittgenstein: Meaning is use.

Consequence: Language are the limits of my world.

We can also describe BNF in BNF. Notice that we can only describe the syntax with BNF in BNF, since BNF is not powerful enough to describe the semantics of BNF.

If you understand the intended meaning behind every Lojban sentence Karl utters, Lojban is powerful enough to also describe semantics.

It also implicitly defines Σ^*, which is part of the language L.

Usually a Backus-Naur Form consists of multiple lines. This is an example Backus-Naur program to describe

In fact we can even define the language of all valid Backus-Naur forms using Backus-Naur:

3.3 Reduce the space of syntax and the space of semantics

Previously I described L using English.

Usually in formal language theory, L is defined as a set and therefore needs to be described using some mathematical language of set-theory, so: $L = \{x \in \{0,1\}^* : \mathit{NumberFromBinaryRepresentation}(x)\%2 = 0\}$ or $L = \{x \in \{0,1\}^* : \mathit{FirstElement}(x) = 1\} \cap \{x \in \{0,1\}^* : \mathit{LastElement}(x) = 0\}$

Notice that this set-theoretic notation is not at all a well-defined way of writing down (although you might be able to define L using only axioms of ZFC, but ZFC does not even hold the necessary ... for ...). a formal the syntax of the language of all

Backus-naur-form can itself be a set.

Set theory is not unique, so we could've defined it as $L = \{x \in \{0,1\}^* : \mathit{FirstElement}(x) = 1\} \cap \{x \in \{0,1\}^* : \mathit{LastElement}(x) = 0\}$ Realise that we could have also expressed this language as (set-theory is): $L = \{x \in \{0,1\}^* : \mathit{FirstElement}(x) = 1 \wedge \mathit{LastElement}(x) = 0\}$

the language of all binary strings, such that the string starts with a one and ends on a zero.

3.4 In what sense is the Church-Turing thesis wrong?

You cannot know everything about the world...

3.5 Syntax is semantics

Translating

If we had already solved these problems for some other formal language M and M were expressive enough to describe L, I could then go ahead and describe the language L with M. Since I know you perfectly understand every sentence I utter in M, I know you then must understand every sentence I utter in L.

In the section before I tried to do precisely this; I tried to define a language L using English. Now you might think it is a bit of a stretch to say that English is a formal language. For example ":) walks to abcdefg" might by some be considered English and by some not. However we can always refer to some common subset of English, for which we both handle the Entscheidungsproblem unambiguously or we might just learn to use a language like Lojban.

If you are a scientist and do empiric modelling of the world, you assume there exists some kind of formal structure in the world, which you try to model using a formal language like mathematics. But if the world can to some degree be described by formal languages, it is not a stretch to say that we learned English by translating it from the language of reality via our senses.

You don't need to be a scientist to create a model of the world,

3.6 Knowing

So if I'm able to explain English to you using a translation from the language of the physical world, how can I know that you understood my explanation of English? In the same vain, how do I know whether I myself understood the rules of physics?

Problem of induction = **Turing test** . Can a fact be true independent of a subject?

The scientists way of gathering truth (using testable hypothesis).

Your thought is either true or false independent of the observer. Think of a world where the universe explodes as soon as you understood all of its rules.

There can always be a world independent of truth

Finding an all-encompassing theory of everything is also **Hilbert's sixth problem**. However you will always face Humes **problem of induction**: Just because all made measurements confirm the theory, needs not to imply that the theory is ultimately correct. A nice way of putting it is: Even if the sun raises every day, you cannot know for certain it will tomorrow.

So if I'm able to explain English to you using a translation from the language of the physical world, how can I know that you understood English?

One way of doing it is through testing **Turing**

We will later explore how machines can be built

3.7 Problem of Semantics

What does it mean for a word to be part of a language. What does it mean say for "$A \vee \neg A$" to be in the language of valid propositional statements?

For a formalist all words are meaningless beside the fact that they either syntactically belong to a formal language or not.

4 Formal systems

In order to give the statement "Everything is a formal language" more weight, I will now introduce formal systems and how they reduce to formal languages (and vice-versa if you want).

4.1 Other terminology

belongs to x belongs to the language L. $x \in L$ x is a **well-formed formula** of L x is a **syntactically valid formula** of L.

Issues: - Brain is not perfect and does mistakes, FSM do not! - English is not a formal language - Physics might be a formal language??

You might think it is a bit of a stretch to say that English is a formal language, since everyone has a different picture in mind

I can also take a small subset of English

and it is a lot harder

The issue of having such an underlying language

But how did we learn English in the first place? While there are a lot of opinions on this matter the formalist simply

Senses are window to the world.

Aristotelian sense

Now the previous two paragraphs rest on a lot of assumptions, is English really a formal language and what does it mean for reality to be a language?

Aristotelian sense How did we learn English?

You might argue that English is not a formal language or that physics is not a formal language and we will discuss this matter more in-depth in a later section. For now please give me the benefit of the doubt on this one.

While English might not be formal (more a term used by many people that describe their own idea of what the language English is), it is quite a complicated language (we will later see what makes a language complicated) and it is a bit of a stretch to say that a anyone completely formalised English.

In English one could define L to be the

4.2 Mathematical induction vs. induction

Example: $L = \{01, 0011, 000111, 00001111, ...\}$

5 Explanation with the help of set theory

I want to state explicitly that formal languages are not a construct resting on set-theory, in fact you can define set-theory as a formal language (which we will do later).

Formal Language $\mathcal{L}$ is a formal language if and only if $\mathcal{L} \subseteq \mathcal{P}(\Sigma)$.

Alphabet An alphabet is denoted as Σ, where Σ is a finite non-empty set.

Symbol/Character x is a symbol/character of $\Sigma \Leftrightarrow x \in \Sigma$

String/Word x is a string over $\Sigma \Leftrightarrow x \in \mathcal{P}(\Sigma)$

Set of all strings $\Sigma^* := \mathcal{P}(\Sigma)$

Entscheidungsproblem Given $x \in Sigma^*, \mathcal{L} \subseteq \Sigma^*$, figure out whether $x \in \mathcal{L}$.

Definition of inclusivity Since set-theory already comes with a few unstated axioms it should be clear that either $x \in \mathcal{L}$ or $x \notin \mathcal{L}$, but not both.

Alphabet $:= \Sigma$, which is of the type finite non-empty set. x is a symbol/character $\Leftrightarrow x \in \Sigma$ x is a string $\Leftrightarrow x \in \mathcal{P}(\Sigma)$ $\Sigma^* := \mathcal{P}(\Sigma)$