Decidability by Frequency Propositions

by Sven Nilsen, 2020

In this paper I show that decidability is a constructive property of frequency propositions. I also prove that there exists generalized forms of propositional logic and that useful proofs require interpretation.

Propositional logic occurs in many contexts. It was discovered because humans found constructs in natural language that formed a pattern. Yet, it appears that some intuition of how to use logic is required. Why is it that logic works how it does? Is it possible to generalize logic in new ways?

A proof \`f\` in propositional logic returns \`true\` for all inputs:

f <=> \true

f : $\text{bool}^N \rightarrow \text{bool}$

For example, here is a proof:

\(a : bool) = (a ∧ ¬a) == false

All proofs have the same function identity. What makes propositional logic interesting is the various expressions that represents ideas that are possible to examine using proofs as a building block. In everyday language, we say that an idea is "true" or "false", but this is somewhat ambiguous.

- An idea is "true" when the function returns \`true\` for all inputs.
- An idea is "false" when the function returns \`false\` for some input.

Propositional logic is used to talk about other stuff, so it is not easy to explain what it means itself. This paper is about finding a technique to prove things about propositional logic as a language.

In a brute force theorem prover, one can generate all possible inputs by using binary counting.

Here is an example of binary counting of 3 bits:

000
001
010
011
100
101
110
111

I gave the arguments different colors to help the reader understanding the next step.

By extracting the bit values vertically, one can form 3 sets:

00001111
00110011
01010101

A proof can also be thought of as a set computed from these extracted sets such that it fills all bits:

11111111

When the number of arguments goes to infinity, the extracted sets become infinite:

00001111…	frequency 1/4
00110011…	frequency 1/2
01010101…	frequency 1/1
…	

For the rest of this paper, I call these sets "frequency propositions".

Although the order of the arguments is irrelevant,
the frequency of a frequency proposition might be defined from the argument index:

frequency(i : nat) = 2^{-i}

The argument index starts at \`0\`, such that \`frequency(0) = 1\`.

This definition is arbitrary and is determined by the implementation of automatic theorem provers.

Think about the frequency as each input lasting for one second.
The frequency describes how many times per second the bit value of the argument changes.
In physics, this unit is measured in "Hertz".
The intuition about the frequency comes from thinking about propositional arguments as square waves.

For example, a frequency 1/4 means the bit flips its value every 4^{th} second.

Such square waves might be infinite, making it easier to reason about proofs of infinite arguments.

It is not just the sets of the arguments that becomes infinite.
Correspondingly, the set that fills all bits becomes infinite:

11111111… frequency 0

There are only two frequency propositions of frequency 0:

00000000…	frequency 0	(not constructible using the \`frequency\` function)
11111111…	frequency 0	(not constructible using the \`frequency\` function)

So far I have discussed how to think about the arguments of proofs as square waves when the number of arguments goes to infinity. Now it is time to take a step back and explain my motivation.

A proof in propositional logic at the syntactical level consists of:

1. A list of symbols, representing the arguments
2. A list of Boolean functions (e.g. logical AND, OR, NOT etc.)
3. Expressions constructed from Boolean functions operating on the symbols

When talking about what a symbol means in propositional logic, one can use the semantics of frequency propositions. Instead of reasoning about the symbol abstractly, one pretends that the semantics of the symbol is a square wave of a given frequency. This makes it possible to reason about symbols without being confused by the interpretation of any specific proof:

All constructive properties that square waves have, are shared with propositions.

Since propositional logic is the basis of many languages for theorem proving,
it also becomes important for the theory about whether a proof can be constructed.

In computer science, decidability means that a problem can be solved using an algorithm.
All problems where you can return `true` or `false` for every input are decidable.
Using the intuition about square waves, one can reason about *decidability* as a kind of “universe”.
The decidable universe is part of a larger universe which contains other signals, not possible to construct using the square waves of frequencies determined from the argument indices.

The semantics of frequency is chosen such that it is easy to generalize to any kind of sets:

1. All arguments of decidable propositional logic have a non-overlapping “frequency”
2. All proofs of decidable propositional logic are constructible from frequency propositions

This means that one does not have to worry about the implementation in larger universes.
As long as these two properties hold, the propositional logic is internally consistent.

The frequency chosen for an argument can be arbitrary, as long it is non-overlapping with others.
It is possible to construct any proof of propositional logic from arbitrary frequency propositions.

Now, I will use this to construct a subtly generalized form of propositional logic.

When a proof holds for `N` arguments, it also holds for `N+1` arguments.
This makes propositional logic useful, since the proof holds no matter how complex the context is.
It would be nice to not have to specify the complexity of the context, while showing proofs will hold.

Imagine this proof of possibly infinite arguments, but which depends only two arguments:

$$\backslash(x : [len] (>= 2)) = (x_0 \land x_1) = (x_1 \land x_0)$$

$$len : [bool] \rightarrow nat$$

Instead of checking this proof for infinite arguments, it is sufficient to check it for the smallest possible case. Since the proof only depends on two arguments, one can limit inputs to `$2^2 = 4$` tests.

At first sight, it might seem this language is just ordinary propositional logic, but this feeling is wrong.

If the proof consists of any expression using a black box function of type:

g : [bool] → bool

Then it is impossible to determine the number of arguments which the proof depends on:

\(x : [bool]) = g(x)   ERROR: Proof is undecidable

This is because the theorem prover can not look into the definition of `g`.

For any fixed number of arguments, propositional logic is decidable.
Propositional logic is also decidable for a bounded range of arguments.
However, when the range of arguments is not bounded, propositional logic becomes undecidable.
This does not mean that is impossible to check all proofs.
As shown above, it is possible to check such proofs when the dependent arguments are known.

This might seem irrelevant to the technique of using frequencies for arguments.
Yet, it turns out that frequency propositions can be used here as well!

When frequency propositions can not be constructed, it is a proof of undecidability.
Since no algorithm can look into `g` to tell the number of arguments it depends on,
it is also impossible to assign a frequency to the argument propositions.
As a consequence, the proof is undecidable.

One can also use this theory to prove more generalized facts about propositional logic as a language.

For example, since frequency propositions are constructed independently of the interpretation of the argument, any proof in logic requires an interpretation to be useful.

Proof: For `N` arguments, there exists at least `N!` interpretations. E.g. a proof of 3 arguments might refer to a list `[“Alice”, “Bob”, “Carl”]` which every permutation of order is a possible interpretation.

A proof is mostly useful when one can use it to refer to some external objects. If the reference is ambiguous, the proof is less useful. The extremely fast growing number of ways a proof without interpretation can be used, makes the proof useless because an interpretation is lacking.

Notice that `N!` interpretations is a lower bound, not an upper bound on possible interpretations.

As the reader might have guessed already (or not?), the theory of frequency propositions gets more powerful when using continuous sets. Those are sets of the type:

real → bool

However, the semantics of this is a lot harder and requires further research.