Motivation of Homotopy Levels From Multiple PSQs

by Sven Nilsen, 2022

In this paper I show that multiple PSQs motivates Martin-Löf identity types and therefore homotopy levels as avatar extension. This implies that multiple PSQs – not pure PSQ – is the foundation of mathematics.

Path Semantical Quantum Propositional Logic (PSQ)[1] extends Propositional Calculus[2] with a single operator `~` called a "qubit"[3]. From this operator alone, one can model many-valued logic[4] and Path Semantical Quality[5] that is necessary for Path Semantics[6].

Ongoing research on PSQ shows, which I will not go into details about here, that there are multiple possible PSQs in constructive logic with reasonable axioms, of the qubit operator, that are exclusive to each other.

For example, assume that PSQ-A and PSQ-B are two possible PSQs. These two logical languages might be incompatible, such that if one assumes both sets of axioms for the qubit operator, one can prove `false`.

To distinguish common theorems of PSQs from specific versions of PSQ, one talks about a "pure PSQ" which does not assume any axioms, except that symbolic identity is a congruence[7]:

```
x : ~a
y : ~a
-------
x == y
```

Notice! This notation is not meant to be interpreted literally.
It roughly translates to dependent types (e.g. Coq), or traits in Rust.

In most mathematical languages, this axiom is unexpressible and assumed implicitly.

What makes the qubit operator in PSQ special, is that,
while symbolic identity is a congruence, the same does not hold for equality:

```
x : ~a
y : ~b
a == b
-------
x ¬= y
```

Equality is not congruent because qubits do not
assume substitution in the argument by equality.

In the classical implementation of PSQ[8], the above holds probabilistically.
This means that it is true in the limit when `x ¬= y` is not expressed directly,
but assumed implicitly and by running proofs infinitely times.

In constructive implementations of PSQ[9], the above axiom is often not assumed, since theorem proving with symbolic distinction requires adding an assumption `(a == b) => (a ~~ b)`.
It is more common to express such cases explicitly, while otherwise simply lacking such proofs.

The reason for strange behaviour is that symbolic identity is very subtle in mathematics.

Symbolic identity is biased by Seshatism[10]. This means that it has a different basis of existence than mathematical objects biased by Platonism[11]. One can imagine mathematical objects in Platonism as having some topology[12], such that topological equivalent[13] objects are equal. In Seshatism, one is able to "probe" the underlying geometry of the topology, such that the abstract mathematical object gets manifested in a "physical form", also called an "avatar"[14].

In most mathematical languages, it is not possible to reason about symbolic identity, because it makes reasoning unsound[15]. However, in Path Semantics[6], it is exploited that symbolic distinction is possible to reason about without causing unsoundness, as long symbolic identity is never manifested directly.

The difficulty of reasoning about symbolic identity, has lead to development of mathematical languages where equality is identical to identity up to some homotopy[16]. The homotopy can be thought of as the definition of topological equivalence.

Basically, by introducing an identity type `Id`[17], one can restore congruence of equality:

x : ~a	
y : ~b	*Congruence is restored by providing additional proof.*
p : Id(a, b)	*However, since this proof requires more axioms to work as*
a == b	*symbolic identity should work, it is non-trivial to implement.*
-------------	*Correct implementation depends on chosen mathematical language.*
x == y	

Now, since `a` and `b` are thought of as identical, it is desirable to not needing mentioning equality, since it should follow from `p : Id(a, b)`. However, to solve this problem, one must develop homotopy levels[18] properly. I will not go into this in detail here.

In summary, by introducing an identity type `Id`, one can reason "as if" symbols were identical.

However, what is the motivation of introducing `Id` in the foundation of mathematics, besides higher levels motivation such as topology?

The answer is that quality in multiple PSQs has the property that different versions of the qubit operator gives rise to different versions of quality, which motivates reasoning relative to pure PSQ in the precise way `Id` is introduced.

The quality in a specific PSQ has the interesting mathematical property that it implies equality while having a notion of a stronger equivalence. This means that the following axiom can hold within a specific PSQ:

x : ~a	
y : ~a	*A stronger notion of equivalence by quality encodes*
v : ~x	*relations that can be used to prove more powerful theorems.*
w : ~y	*In one interpretation, the qubit operator behaves like time.*
-------	*Time passes when the future is constructed partially.*
x ~~ y	

The notion of quality depends on the qubit operator, so each specific PSQ can use a "self-stronger" sense of equivalence that distinguishes PSQs from each other, while sharing pure PSQ through equality. `x ~~ y` above is defined as `(x == y) ∧ ~x ∧ ~y`, which explains why `v` and `w` are needed. To make such reasoning more powerful, one requires introducing `Id` and homotopy levels.

References:

[1] "PSQ – Path Semantical Quantum Propositional Logic"
AdvancedResearch – Summary page on Path Semantical Quality
https://advancedresearch.github.io/quality/summary.html#psq---path-semantical-quantum-propositional-logic

[2] "Propositional calculus"
Wikipedia
https://en.wikipedia.org/wiki/Propositional_calculus

[3] "Path Semantical Qubit"
Wikipedia
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip2/path-semantical-qubit.pdf

[4] "Many-valued logic"
Wikipedia
https://en.wikipedia.org/wiki/Many-valued_logic

[5] "Path Semantical Quality"
Sven Nilsen, 2021
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip2/path-semantical-quality.pdf

[6] "Path Semantics"
Sven Nilsen, 2016-2021
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip/path-semantics.pdf

[7] "Congruence relation"
Wikipedia
https://en.wikipedia.org/wiki/Congruence_relation

[8] "Pocket-Prover"
AdvancedResearch
https://github.com/advancedresearch/pocket_prover

[9] "Prop – Propositional logic with types in Rust"
AdvancedResearch
https://github.com/advancedresearch/prop

[10] "Seshatism"
Sven Nilsen, 2021-2022
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip2/seshatism.pdf

[11] "Seshatism vs Platonism"
AdvancedResearch – Summary page on Avatar Extensions
https://advancedresearch.github.io/avatar-extensions/summary.html#seshatism-vs-platonism

[12] "Topology"
Wikipedia
https://en.wikipedia.org/wiki/Topology

[13] “Topological equivalence”
Encyclopedia of Mathematics
https://encyclopediaofmath.org/wiki/Topological_equivalence

[14] “Avatar Extensions”
AdvancedResearch – Summary page on Avatar Extensions
https://advancedresearch.github.io/avatar-extensions/summary.html

[15] “Symbolic Distinction”
Sven Nilsen, 2021
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip2/symbolic-distinction.pdf

[16] “Homotopy Type Theory”
Home page for Homotopy Type Theory
https://homotopytypetheory.org/

[17] “identity type”
nLab
https://ncatlab.org/nlab/show/identity+type

[18] “homotopy level”
nLab
https://ncatlab.org/nlab/show/homotopy+level