# Counter-Example to Leibniz's First Principle

by Sven Nilsen, 2023

*In this paper I show that Leibniz's First Principle does not hold in Path Semantics.*

Leibniz's First Principle[1] is stated in the logical form:

$$(x = y) \Rightarrow \forall f \{ f(x) == f(y) \}$$

Here, a single `=` means some stronger notion of equality, which in Path Semantics is stated:

$$(x == y)\hat{}true \Rightarrow \forall f \{ f(x) == f(y) \}$$

The reason Path Semantics uses tautological propositional equality[2] instead of `x = y`, is because reasoning with symbolic indistinction[3] is not sound[4]. Here is why:

Assume that `x = y` means that `x` and `y` are symbolic indistinct. Evaluate the following:

$$sd(a, b) \wedge (a == b)\hat{}true \Rightarrow (a = b)$$

On the left side, `sd(a, b)` means `a` and `b` are symbolic distinct[4]. If `a` and `b` are symbolic indistinct, then this operator does not terminate. However, assume that `a` and `b` actually are symbolic distinct, so we can continue evaluation. The statement `(a == b)^true` is perfectly plausible because if `a` and `b` can be proved to be propositionally equal, then this holds.

However, on the right side, `a = b` means that `a` and `b` are symbolic indistinct. Yet, this is a contradiction, because if `sd(a, b)` on the left side, then `a` and `b` are not symbolic indistinct:

$$sd(a, b) \Rightarrow \neg(a = b)$$

Since we can prove both `a = b` and `¬(a = b)`, it is possible to prove `false`.

This means that `=` in `a = b` can not be a predicate, as this would violate Leibniz's First Principle. However, if `a = b` is not a predicate, then one can not formulate Leibniz's First Principle for tautological propositional equality. This property is desired in Path Semantics, so the theory designs around this problem. To be more useful in theorem proving, function application is relaxed:

$$(x == y) \Rightarrow \forall f \{ f(x) == f(y) \}$$

On the other hand `~x == ~y` requires tautological propositional equality:

$$(x == y)\hat{}true \Rightarrow (\sim x == \sim y)$$

When one applies `~` to some argument, it is not considered the same as function application. The qubit operator `~` lives outside the universe of functions in Path Semantics, like interval types in Cubical Type Theory[5]. Furthermore, symbolic indistinction is excluded from the language. Therefore, Leibniz's First Principle does not hold, since there is no symbolic indistinction.

## References:


[1] “Identity of indiscernibles”
Wikipedia
https://en.wikipedia.org/wiki/Identity_of_indiscernibles

[2] “HOOO Exponential Propositions”
Sven Nilsen, 2022-2023
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip2/hooo-exponential-propositions.pdf

[3] “Symbolic Indistinction as Propositional Infinity”
Sven Nilsen, 2023
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip2/symbolic-indistinction-as-propositional-infinity.pdf

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

[5] “cubical type theory”
nLab
https://ncatlab.org/nlab/show/cubical+type+theory