Semiconjuctions as Satisfied Models of Total Normal Paths

by Sven Nilsen, 2021

In this paper I discuss semiconjuctions as satisfied models of total normal paths.

The following proof was provided by Eric Weiser,
implemented for the community edition of Lean 3 theorem prover[1][2]:

```
import logic.function.conjugate
import tactic

open function

variables (α β γ : Type*)

notation g `[` f:(foldl `][` (h t, function.comp h t) id `]`) ` <=> ` g' := semiconj₂ f g g'

example (g₁ : α → β) (g₂ : β → γ) (f : α → α → α) (h₁ : β → β → β) (h₂ : γ → γ → γ) :
(f[g₁] <=> h₁) → (h₁[g₂] <=> h₂ ) → (f[g₁][g₂] <=> h₂) :=
λ a b, semiconj₂.comp b a
```

This proof uses semiconjuctions from Mathlib[3] and shows that dependent types can prove composition in path space, given satisfied models of symmetric normal paths on total[4] binary[5] functions. The most notable about this proof is that Eric found a way to create the syntax of path semantics, so one does not need to translate between two different notations. This demonstrates the possibility of using path semantical notation in Lean. The remaining part of the proof was simply calling a library function in Mathlib that composes semiconjuctions.

Total normal paths are equivalent to introducing an imaginary inverse[6]. With current knowledge, total normal paths can still not be modelled fully by dependent types, because total normal paths can be composed when imaginary (having no solution), yielding another normal path that might have a solution.

However, it might be possible to apply the Axiom of Choice[7] in Lean 3,
to construct some analogue of the imaginary inverse using `inv_fun_on`[8]:

```
noncomputable def inv_fun_on (f : α → β) (s : set α) (b : β) : α :=
if h : ∃a, a ∈ s ∧ f a = b then classical.some h else classical.choice n
```

This brings us one step closer to understanding normal paths formally in dependent types.

References:

[1] “Lean Theorem Prover”
Microsoft Research
https://leanprover.github.io/

[2] “Lean Community”
Lean Community Website
https://leanprover-community.github.io/

[3] “Semiconjugate and commuting maps”
mathlib documentation
https://leanprover-community.github.io/mathlib_docs/logic/function/conjugate.html

[4] “Partial function”
Wikipedia
https://en.wikipedia.org/wiki/Partial_function

[5] “Binary function”
Wikipedia
https://en.wikipedia.org/wiki/Binary_function

[6] “Imaginary Inverse”
Sven Nilsen, 2020
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip/imaginary-inverse.pdf

[7] “Axiom of Choice”
Wikipedia
https://en.wikipedia.org/wiki/Axiom_of_choice

[8] “inv_fun_on”
Mathlib
https://github.com/leanprover-community/mathlib/blob/26e4f15f67fee3883aabe217790f8bbdf7460566/src/logic/function/basic.lean#L208-L209