Seshatic Inequality Overloading

by Sven Nilsen, 2021-2023

In this paper I introduce a technique of overloading inequality with a Seshatic relation.

Assume one has a member of some type:

a : T

Now, pick one value of some second type, that is symbolic distinct[1] from the first:

b : U

Seshatic Inequality Overloading is the idea that members of two symbolic distinct types are inequal, by propagating inquality[2] to inequality:

¬(a ~~ b) => eq((a, b)) = false

With the extended definition of the type of `eq`:

¬(T == U) => eq{T : type, U : type} : T × U -> bool

This form of overloading is only valid when two symbolic distinct types can not be treated as equal.

Seshatic Inequality Overloading works, because because one can prove in
Path Semantical Intuitionistic Logic[3] (PSI):

(a : T) ∧ (b : U) ∧ ¬(T == U) => ¬(a ~~ b)

Notice that one can not prove `¬(a == b)`.
Seshatic Inequality Overloading requires path semantical quality[2].

For an implementation, see the `quality_traits::SeshNeq` trait in the Prop library[4].

References:

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

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

[3] “Path Semantical Logic”
AdvancedResearch – Reading sequences on Path Semantics
https://github.com/advancedresearch/path_semantics/blob/master/sequences.md#path-semantical-logic

[4] “Prop”
AdvancedResearch – Propositional logic with types in Rust
https://github.com/advancedresearch/prop