`outline`

1. Structures and Theories

1.1. Language and Structures

1.1.1. Definition

a language $\mathcal{L}$ is defined by the following data

1. a set of functions $\mathcal{F}$ and positive integers n_f foreach $f \in \mathcal{F}$ which is the arity of each function
2. a set of relations $\mathcal{R}$ and positive integers n_R foreach $R \in \mathcal{R}$ which is the arity of each relation
3. a set of constant symbols $\mathcal{C}$

all sets can be empty

1.1.2. Examples

1. the language of rings $\mathcal{L}_r = \{+, -, \cdot, 0, 1\}$
2. the language of ordered rings $\mathcal{L}_{or} = \mathcal{L}_r \cup \{<\}$
3. the language of pure sets $\mathcal{L} = \emptyset$
4. the language of graphs $\mathcal{L} = \{R\}$ where R is a binary relation

Remark: *Seems to me that it resembles a lot the conçept of an abstract interface in programming languages like: Interfaces in Java, Traits in Rust, Typeclasses in Haskell, Conçepts in C++*

1.1.3. Definition

An $\mathcal{L}$-structure $\mathcal{M}$ is given by the following data:

1. a non empty set M called universe/domain/underlying set of $\mathcal{M}$
2. a function $f^{\mathcal{M}} : M^{n_f} \to M$ for each $f \in \mathcal{F}$
3. a set $R^{\mathcal{M}} \subseteq M^{n_R}$ for each $R \in \mathcal{R}$
4. an element $c^{\mathcal{M}} \in M$ for each $c \in \mathcal{C}$

the last 3 are called interpretations of their counterpart

$\mathcal{M}$ is generaly given by a tuple of its components

1.1.4. Examples

for example we have the language of groups being $\mathcal{L}_g = \{\cdot, e\}$. an $\mathcal{L}$-structure $G = (\mathbb{R}, \cdot, 1)$ or even $G = (\mathbb{Z}, +, 0)$

Remark: *This in turn looks to me like the implementation of the structure in programming languages*

1.1.5. Definition

suppose $\mathcal{M}, \mathcal{N}$ are structures of $\mathcal{L}$ with universes M, N, an $\mathcal{L}$-embedding $\eta : \mathcal{M} \to \mathcal{N}$ is a one-to-one embedding map $\eta : M \to N$ that preserves structure as follows:

1. $\eta\left(f^{\mathcal{M}}(a_1, ..., a_n)\right) = f^{\mathcal{N}}(\eta(n_1), ..., \eta(n_n))$
2. $(a_1, ..., a_m) \in R^{\mathcal{M}}$ iff $(\eta(a_1), ..., \eta(a_m)) \in R^{\mathcal{N}}$
3. $\eta\left(n^{\mathcal{M}}\right) = c^{\eta(N)}$

some special cases:

- a bijective $\mathcal{L}$-embedding is callend an $\mathcal{L}$-isomorphism.
- if $M \subseteq N$ and inclusion map is an $\mathcal{L}$-embedding then $\mathcal{M}$ is a substructure of $\mathcal{N}$ and $\mathcal{N}$ is an extension of $\mathcal{M}$

1.1.6. Examples

if $\eta : \mathbb{Z} \to \mathbb{R}$ is $\eta(x) = e^x$ then it is $\mathcal{L}$-embedding of $(\mathbb{Z}, +, 0)$ into $(R, \cdot, 1)$

1.1.7. Definition

$\mathcal{L}$-terms is the smallest set $\mathcal{T}$ such that

1. $\forall c \in \mathcal{C}, c \in \mathcal{T}$
2. each variable symbol $v_i \in \mathcal{T}$ for $i \in \mathbb{N}^*$
3. if $t_1, ..., t_n \in \mathcal{T}$ and $f \in \mathcal{F}$ then $f(t_1, ..., t_n) \in \mathcal{T}$

1.1.8. Examples

for $\mathcal{L}_r$ (ring language) we have $\cdot\,(v_1, -(v_3, 1))$ and $+(1, +(1, +(1, 1)))$ this example is written in S-expressions like lisp but it works with infix

Remark: *this also is exactly the same thing as defining what is a valid term in programming languages*