Time Hybrids

Lamda Days 2024

Luc Duponcheel

Time Hybrids

nova science publishers

A New Generic Theory of Reality

Fred Van Oystaeyen

Generic Theory

Generic Theory

- theory of theories

Generic Theory

- theory of theories
- partially unifying framework theory where theories fit into

Generic Theory of Reality

Generic Theory of Reality

- partially unifying framework theory where theories of reality fit into

Generic Theory of Reality

- partially unifying framework theory where theories of reality fit into
    - relativity theory

Generic Theory of Reality

- partially unifying framework theory where theories of reality fit into
  - relativity theory
  - quantum theory

Generic Theory of Reality

- partially unifying framework theory where theories of reality fit into
    - relativity theory
    - quantum theory
- until now no unifying theory for them has been agreed upon

Specification

Specification

- *declares* features

Specification

- *declares* features
- *defines* laws that come with those declared features

Specification

- *declares* features
- *defines* laws that come with those declared features
- also *defines* features in terms of declared and defined features

Implementation

Implementation

- *defines* declared features of a specification

Implementation

- *defines* declared features of a specification
- provides proofs of the laws that come with those declared features

Description
(as a painting)

Description
(as a painting)

- below is an informal description of a pipe

Description
(as a painting)

- below is an informal description of a pipe

- well, not really, does it?

Description
(as a computational simulation)

Description
(as a computational simulation)

- this TED talk of Stephen Wolfram is about a, mathematically well-founded, description of reality that, maybe, can, somehow, be seen as an implementation of the specification of the generic theory of Fred Van Oystaeyen

Continuous versus Discrete

Continuous versus Discrete

- relativity theory resp. quantum theory is a macro theory resp. micro theory, but where do micro and macro end?

Continuous versus Discrete

- relativity theory resp. quantum theory is a macro theory resp. micro theory, but where do micro and macro end?
- traditionally, mathematics is used to deal with that in a continous, analytic way, using shrinking limits of time intervals

Continuous versus Discrete

- relativity theory resp. quantum theory is a macro theory resp. micro theory, but where do micro and macro end?
- traditionally, mathematics is used to deal with that in a continous, analytic way, using shrinking limits of time intervals
- recently, mathematics is used to deal with that in a discrete, algebraic way (if only because of the observational minimal Planck time unit!)

Discrete

Discrete

- Stephen Wolfram's mathematics uses an expanding limit of a time interval starting at some time moment (think of a big bang of graph rewriting without specific rules), recall that it is an implementation

Discrete

- Stephen Wolfram's mathematics uses an expanding limit of a time interval starting at some time moment (think of a big bang of graph rewriting without specific rules), recall that it is an implementation
- Fred Van Oystaeyen's approach also goes for a discrete, algebraic way but makes no concrete choices, recall that it is a specification

Generic Theory of Mathematics as a foundation

Generic Theory of Mathematics as a foundation

- Category Theory is a partially unifying framework theory where theories of mathematics fit into

Generic Theory of Mathematics as a foundation

- Category Theory is a partially unifying framework theory where theories of mathematics fit into
  - probability

Generic Theory of Mathematics as a foundation

- Category Theory is a partially unifying framework theory where theories of mathematics fit into
  - probability
  - geometry (topology)

Generic Theory of Mathematics as a foundation

- Category Theory is a partially unifying framework theory where theories of mathematics fit into
  - probability
  - geometry (topology)
  - others

Generic Theory of Mathematics as a foundation

- Category Theory is a partially unifying framework theory where theories of mathematics fit into
  - probability
  - geometry (topology)
  - others
- Category theory is, just like reality, compositional

Virtual Topology

Virtual Topology

- Fred Van Oystayen's Virtual Topology is, as far as I know, the most abstract (read: simplest) geometry as far as being useful for modeling reality

Virtual Topology

- Fred Van Oystayen's Virtual Topology is, as far as I know, the most abstract (read: simplest) geometry as far as being useful for modeling reality
- Virtual Topology is pointfree (cfr Category Theory)

Program language notation

Program language notation

- Mathematical notation can overwhelm you

$$R_{\mu\nu} - \frac{1}{2} R \, g_{\mu\nu} + \Lambda \, g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$$

Program language notation

- Mathematical notation can overwhelm you

$$R_{\mu\nu} - \frac{1}{2} R\, g_{\mu\nu} + \Lambda\, g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$$

- Natural language notation can confuse you

Program language notation

- Mathematical notation can overwhelm you

$$R_{\mu\nu} - \frac{1}{2} R \, g_{\mu\nu} + \Lambda \, g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$$

- Natural language notation can confuse you
  - many words used for one concept without explanantion

Program language notation

- Mathematical notation can overwhelm you

$$R_{\mu\nu} - \frac{1}{2} R \, g_{\mu\nu} + \Lambda \, g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$$

- Natural language notation can confuse you
  - many words used for one concept without explanantion
  - one word used for many concepts without defining context

Program language notation

- Mathematical notation can overwhelm you

$$R_{\mu\nu} - \frac{1}{2}R\,g_{\mu\nu} + \Lambda\,g_{\mu\nu} = \frac{8\pi G}{c^4}T_{\mu\nu}$$

- Natural language notation can confuse you
  - many words used for one concept without explanantion
  - one word used for many concepts without defining context
- Program language notation is required to be precise and, therefore may also overwhelm you, but the benefit is that it is checked by a type system that, by the way, is not overwhelmed by it

Category

```
trait Category[BTC[_, _]]
  extends BtcComposition[BTC], BtcUnit[BTC]:
```

CategoryLaws

```
trait CategoryLaws[L[_]: Law]:

  def leftIdentity[Z, Y]: BTC[Z, Y] => L[BTC[Z, Y]] =
    z2y => { i `o` z2y } `=` { z2y }

  def rightIdentity[Z, Y]: BTC[Z, Y] => L[BTC[Z, Y]] =
    z2y => { z2y `o` i } `=` { z2y }
```

BtcComposition

```scala
trait BtcComposition[BTC[_, _]]:

  extension [Z, Y, X](y2x: BTC[Y, X])
    def `o`(z2y: BTC[Z, Y]): BTC[Z, X]
```

BtcCompositionLaws

```
trait CompositionLaws[L[_]: Law]:

  def associativity[Z, Y, X, W]
      : BTC[X, W] => BTC[Y, X] => BTC[Z, Y] => L[BTC[Z, W]] =
    x2w => y2x => z2y =>
      { (x2w ‘o‘ y2x) ‘o‘ z2y } ‘=‘ { x2w ‘o‘ (y2x ‘o‘ z2y) }
```

BtcUnit

```
trait BtcUnit[BTC[_, _]]:
  def i[Z]: BTC[Z, Z]
```

Time

```
trait Time[Moment: Arbitrary: Ordered]
```

Time

```
trait Time[Moment: Arbitrary: Ordered]
```

- this allows us to

Time

```
trait Time[Moment: Arbitrary: Ordered]
```

- this allows us to
  - write statements involving arbitrary time moments

Time

```
trait Time[Moment: Arbitrary: Ordered]
```

- this allows us to
  - write statements involving arbitrary time moments
  - state that one time moment is before another one

Universe

```
trait Universe[
  Set[_]: Sets,
  Morphism[_, _]: Category: ActingUponFunction,
  Moment: Time,
  State: [_] =>> VirtualTopology[
    Set,
    State
  ]: [_] =>> Functor[
    [_, _] =>> MomentMorphism,
    Morphism,
    [_] =>> State
  ]
]
```

Universe continued

Universe continued

- this allows us to

Universe continued

- this allows us to
  - write statements using topological features of universe states

Universe continued

- this allows us to
  - write statements using topological features of universe states
  - write statements relating time moment transitions, to universe state morphisms (think of an dynamic, expanding universe)

An example: immobility

An example: immobility

- immobility can be expressed without mentioning real numbers for time and space

An example: immobility

- immobility can be expressed without mentioning real numbers for time and space
- some history of geometry

An example: immobility

- immobility can be expressed without mentioning real numbers for time and space
- some history of geometry
    - Euclides did not use numbers to measure with

An example: immobility

- immobility can be expressed without mentioning real numbers for time and space
- some history of geometry
  - Euclides did not use numbers to measure with
  - Pythagoras showed that rational numbers were not enough

immobileAfter

```
val immobileAfter
  : MomentMorphism => L[Function[PreThingsSet, State]] =
mm =>
  {
    pts2s `o` mm2ptsf(mm)
  } `=` {
    mm2sm(mm) `a` pts2s
  }
```

immobileAfter

```
val immobileOnInterval
  : MomentMorphism => PreThingsSet => L[State] =
  case (bm, em) =>
    val mi: Interval[Moment] = interval apply ((bm, em))
    pts =>
      all apply {
        for {
          m <- mi
        } yield {
          {
            (pts2s `o` mm2ptsf((bm, m)))(pts)
          } `=` {
            (mm2sm((bm, m)) `a` pts2s)(pts)
          }
        }
      }
```

Reality goes beyond Imagination

Reality goes beyond Imagination

- the ‘a‘ in the formula is not morphism composition ‘o‘

Reality goes beyond Imagination

- the `‘a‘` in the formula is not morphism composition `‘o‘`
- the `‘a‘` in the formula is an action of morphism `mm2sm(mm)` on the place function `pts2s`

Reality goes beyond Imagination

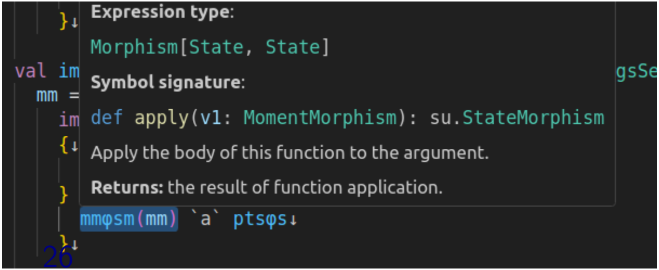
- the ‘a‘ in the formula is not morphism composition ‘o‘
- the ‘a‘ in the formula is an action of morphism mm2sm(mm) on the place function pts2s
- cfr a monoid acting upon a set

Reality goes beyond Imagination

- the `‘a‘` in the formula is not morphism composition `‘o‘`
- the `‘a‘` in the formula is an action of morphism `mm2sm(mm)` on the place function `pts2s`
- cfr a monoid acting upon a set
- cfr rotations acting upon a Rubic Cube

Type System


val immobileAfter: M
mm =>↓
import su.{mmφsm
{↓
ptsφs `o` mmφp
} `=` {↓
mmφsm(mm) `a` ptsφs↓
}↓
ate]
Expression type:
Set[Composition2[Set, PreObject]] => State
Symbol signature:
val ptsφs: PreThingsSet => State


}↓
val im
mm =
im
{↓
}
mmφsm(mm) `a` ptsφs↓
}↓
gsSe
Expression type:
Morphism[State, State]
Symbol signature:
def apply(v1: MomentMorphism): su.StateMorphism
Apply the body of this function to the argument.
Returns: the result of function application.

More Information

https://github.com/LucDuponcheelAtGitHub/timeHybrids

THANKS FOR ATTENDING