Towards verified hardware-aware Topological Quantum Programming

Urs Schreiber @ CQTS CENTER FOR QUANTUM & TOPOLOGICAL SYSTEMS

presenting at:

CQTS and TII Workshop

NYU Abu Dhabi, 24 Feb 2023

slides and references at: ncatlab.org/schreiber/show/TQP#TalkForTII

Real quantum computation

$\underbrace{\text{scalable, universal, reliable}}$

Real quantum computation

scalable, universal, reliable

Real quantum computation will require:

scalable, universal, reliable

Real quantum computation will require

scalable, universal, reliable

Real quantum computation will require

stabilization

$\underbrace{\text{scalable, universal, reliable}}$

Real quantum computation will require

stabilization

compilation

$\underbrace{\text{Real}}_{}$

scalable, universal, reliable

Real quantum computation will require

stabilization

compilation

verification

$\overbrace{\text{Real}}^{\text{scalable, universal, reliable}}$ quantum computation will require

stabilization

compilation

verification

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation

verification

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

> “*small* [NISQ] *machines are unlikely to uncover truly macroscopic quantum phenomena, which have no classical analogs.*
>
> ”

J. Sau: *Roadmap for Scalable Topological Quantum Computers*
Physics **10** (2017) 68
`[physics.aps.org/articles/v10/68]`

$\underbrace{\text{scalable, universal, reliable}}$

Real quantum computation will require

stabilization ↔ *topological* gates

> “*small* [NISQ] *machines are unlikely to uncover truly macroscopic quantum phenomena, which have no classical analogs. This will likely require a scalable approach to quantum computation* [...] *based on* [...] *topological quantum computation (TQC)*
>
> ”

J. Sau: *Roadmap for Scalable Topological Quantum Computers*
Physics **10** (2017) 68
`[physics.aps.org/articles/v10/68]`

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

> “*small* [NISQ] *machines are unlikely to uncover truly macroscopic quantum phenomena, which have no classical analogs. This will likely require a scalable approach to quantum computation* [...] *based on* [...] *topological quantum computation (TQC)* [...] *The central idea of TQC is to encode qubits into states of topological phases of matter.*
>
> ”

J. Sau: *Roadmap for Scalable Topological Quantum Computers*
Physics **10** (2017) 68
`[physics.aps.org/articles/v10/68]`

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

> “*small* [NISQ] *machines are unlikely to uncover truly macroscopic quantum phenomena, which have no classical analogs. This will likely require a scalable approach to quantum computation* [...] *based on* [...] *topological quantum computation (TQC)* [...] *The central idea of TQC is to encode qubits into states of topological phases of matter. Qubits encoded in such states are expected to be topologically protected, or robust, against the ’prying eyes’ of the environment, which are believed to be the bane of conventional quantum computation.*”

J. Sau: *Roadmap for Scalable Topological Quantum Computers*
Physics **10** (2017) 68
`[physics.aps.org/articles/v10/68]`

$\overbrace{\text{Real}}^{\text{scalable, universal, reliable}}$ quantum computation will require

stabilization ↔ *topological* gates

compilation

verification

$\underbrace{\text{scalable, universal, reliable}}$

Real quantum computation will require

stabilization ↔ *topological* gates

> *"The qubit systems we have today are a tremendous scientific achievement,*

S. Das Sarma: *Quantum computing has a hype problem*
MIT Technology Review (March 2022)
`[www.technologyreview.com/2022/03/28/1048355/quantum-computing-has-a-hype-problem]`

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

> *“The qubit systems we have today are a tremendous scientific achievement, but they take us no closer to having a quantum computer that can solve a problem that anybody cares about.*

S. Das Sarma: *Quantum computing has a hype problem*
MIT Technology Review (March 2022)
`[www.technologyreview.com/2022/03/28/1048355/quantum-computing-has-a-hype-problem]`

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

> *“The qubit systems we have today are a tremendous scientific achievement, but they take us no closer to having a quantum computer that can solve a problem that anybody cares about.* [...] *What is missing is the breakthrough* [...] *bypassing quantum error correction by using far-more-stable qubits, in an approach called topological quantum computing.”*

S. Das Sarma: *Quantum computing has a hype problem*
MIT Technology Review (March 2022)
`[www.technologyreview.com/2022/03/28/1048355/quantum-computing-has-a-hype-problem]`

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation

verification

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification

$\overbrace{\text{Real}}^{\text{scalable, universal, reliable}}$ quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

“Why did discovering quantum teleportation take 60 years?

Bob Coecke: *Kindergarten QM* `[arXiv:quant-ph/0510032]`
Quantum Picturalism `[arXiv:0908.1787]`

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

“Why did discovering quantum teleportation take 60 years?
We claim that this is due to a ‘bad quantum formalism’

Bob Coecke: *Kindergarten QM* `[arXiv:quant-ph/0510032]`
Quantum Picturalism `[arXiv:0908.1787]`

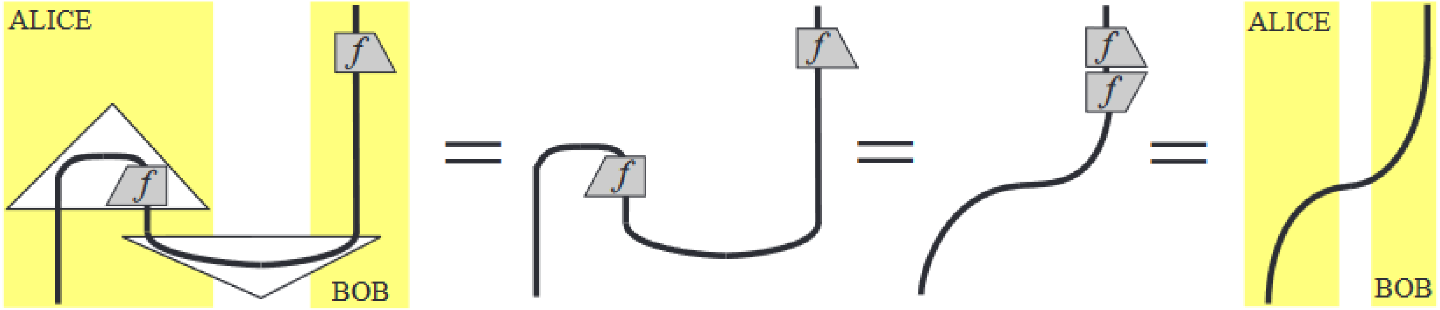
scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

> *“Why did discovering quantum teleportation take 60 years?*
> *We claim that this is due to a ‘bad quantum formalism’*
> *I claim that a good formalism exists:* [linear circuit logic]



Bob Coecke: *Kindergarten QM* `[arXiv:quant-ph/0510032]`
Quantum Picturalism `[arXiv:0908.1787]`

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

ve

even with good linear circuit logic:

$\overbrace{\hphantom{\text{Real quantum computation}}}^{\text{scalable, universal, reliable}}$

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

ve

even with good linear circuit logic:

topological gate set is highly constrained,

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

ve

even with good linear circuit logic:

topological gate set is highly constrained,
& movement of topological qbits is costly

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

ve

even with good linear circuit logic:

topological gate set is highly constrained,
& movement of topological qbits is costly
⇒ topological quantum compilation intricate

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

to declare the **data type** of all data

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

to declare the **data type** of all data
formally specifying the admissible
data construction and behaviour

$\overbrace{\text{Real quantum computation}}^{\text{scalable, universal, reliable}}$ will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

to declare the **data type** of all data
formally specifying the admissible
data construction and behaviour
(aka: “formal methods”)

$\overbrace{\text{Real}}^{\text{scalable, universal, reliable}}$ quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

> *“We argue that quantum programs demand machine-checkable proofs of correctness. We justify this on the basis of*

Robert Rand: *Formally Verified Quantum Programming*
UPenn (2018) [repository.upenn.edu/edissertations/3175]

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

> *“We argue that quantum programs demand machine-checkable proofs of correctness. We justify this on the basis of the complexity of programs manipulating quantum states,*

Robert Rand: *Formally Verified Quantum Programming*
UPenn (2018) [repository.upenn.edu/edissertations/3175]

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

> *"We argue that quantum programs demand machine-checkable proofs of correctness. We justify this on the basis of the complexity of programs manipulating quantum states, the expense of running quantum programs, and*

Robert Rand: *Formally Verified Quantum Programming*
UPenn (2018) [repository.upenn.edu/edissertations/3175]

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

> *“We argue that quantum programs demand machine-checkable proofs of correctness. We justify this on the basis of the complexity of programs manipulating quantum states, the expense of running quantum programs, and the inapplicability of traditional debugging techniques to programs whose states cannot be examined.*

Robert Rand: *Formally Verified Quantum Programming*
UPenn (2018) [repository.upenn.edu/edissertations/3175]

$\overbrace{\text{Real}}^{\text{scalable, universal, reliable}}$ quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

> *“We argue that quantum programs demand machine-checkable proofs of correctness. We justify this on the basis of the complexity of programs manipulating quantum states, the expense of running quantum programs, and the inapplicability of traditional debugging techniques to programs whose states cannot be examined.*
>
> **Thesis Statement**:
>
> Quantum programming is not only amenable to formal verification: it demands it.”

Robert Rand: *Formally Verified Quantum Programming*
UPenn (2018) [repository.upenn.edu/edissertations/3175]

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

all the more for topological quantum computing:
due to exotic gates in complex & unituitive circuits

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

all the more for topological quantum computing:
due to exotic gates in complex & unituitive circuits

but existing quantum circuit verification languages
such as `QWIRE` or `Quipper`
lack support for topological gates

$\overbrace{\text{Real}}^{\text{scalable, universal, reliable}}$ quantum computation will require

stabilization	↔	*topological* gates
compilation	↔	*linear* circuit logic
verification	↔	*typed* quantum data

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

Real quantum programming is

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

Real quantum programming is *homotopy*

scalable, universal, reliable

Real quantum computation will require

stabilization	↔	*topological* gates
compilation	↔	*linear* circuit logic
verification	↔	*typed* quantum data

Real quantum programming is *linear homotopy*

scalable, universal, reliable

Real quantum computation will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

Real quantum programming is *linear homotopy typed*

$\overbrace{\text{Real quantum computation}}^{\text{scalable, universal, reliable}}$ will require

stabilization ↔ *topological* gates

compilation ↔ *linear* circuit logic

verification ↔ *typed* quantum data

Real quantum programming is *linear homotopy typed* programming.

$\overbrace{\text{Real}}^{\text{scalable, universal, reliable}}$ quantum computation will require

stabilization	↔	*topological* gates
compilation	↔	*linear* circuit logic
verification	↔	*typed* quantum data

Real quantum programming is *linear homotopy typed* programming.

Foundations Project
@ CQTS

scalable, universal, reliable

Part I

Verifying realistic topological quantum gates

Real quantum programming is *linear homotopy typed* programming.

Foundations Project
@ CQTS

scalable, universal, reliable

Part I

Verifying realistic topological quantum gates

Part II

Verifying their compilation into quantum circuits

Real quantum programming is *linear homotopy typed* programming.

Foundations Project
@ CQTS

Part I
Verifying realistic topological quantum gates

Part II
Verifying their compilation into quantum circuits

Towards verifying realistic topological quantum gates

Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates

seems a formidable task and
grossly inefficient even if possible.

Towards verifying realistic topological quantum gates

Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates
aka: anyon braiding in topological materials,

seems a formidable task and
grossly inefficient even if possible.

Towards verifying realistic topological quantum gates

Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates
aka: anyon braiding in topological materials,
aka: monodromy of WZW-conformal blocks,

seems a formidable task and
grossly inefficient even if possible.

Towards verifying realistic topological quantum gates

Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates
aka: anyon braiding in topological materials,
aka: monodromy of WZW-conformal blocks,
aka: Wilson lines of Chern-Simons QFT
seems a formidable task and
grossly inefficient even if possible.

Towards verifying realistic topological quantum gates

Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates

aka: anyon braiding in topological materials,
aka: monodromy of WZW-conformal blocks,
aka: Wilson lines of Chern-Simons QFT

seems a formidable task and
grossly inefficient even if possible.

Nobody had touched this, even remotely.

Towards verifying realistic topological quantum gates

Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates
aka: anyon braiding in topological materials,
aka: monodromy of WZW-conformal blocks,
aka: Wilson lines of Chern-Simons QFT
seems a formidable task and
grossly inefficient even if possible.

Nobody had touched this, even remotely.
But recently we showed that:

Towards verifying realistic topological quantum gates

Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates
    aka: anyon braiding in topological materials,
    aka: monodromy of WZW-conformal blocks,
    aka: Wilson lines of Chern-Simons QFT
seems a formidable task and
grossly inefficient even if possible.

Nobody had touched this, even remotely.
But recently we showed that:

Realistic topological quantum gates secretly
are natives of parameterized homotopy theory

Towards verifying realistic topological quantum gates

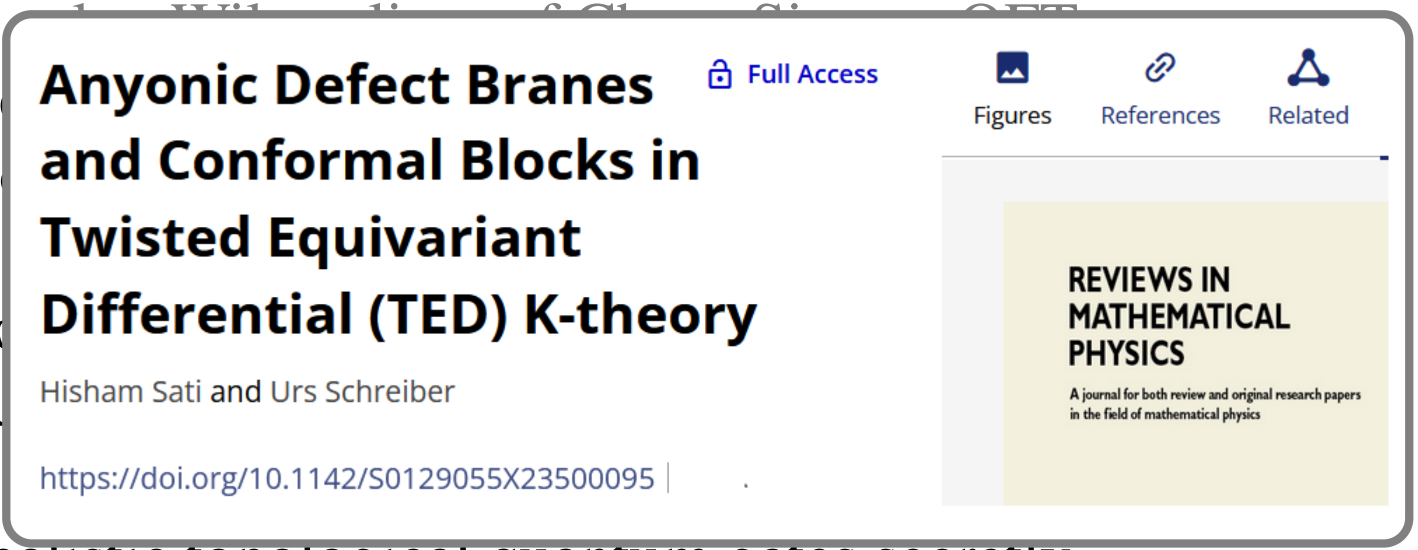
Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates
    aka: anyon braiding in topological materials,
    aka: monodromy of WZW-conformal blocks,
    aka: Wilson lines of Chern-Simons QFT

se

gr

No

Bu



Anyonic Defect Branes and Conformal Blocks in Twisted Equivariant Differential (TED) K-theory

Hisham Sati and Urs Schreiber

https://doi.org/10.1142/S0129055X23500095

REVIEWS IN MATHEMATICAL PHYSICS

A journal for both review and original research papers in the field of mathematical physics

Realistic topological quantum gates secretly
are natives of parameterized homotopy theory

Towards verifying realistic topological quantum gates

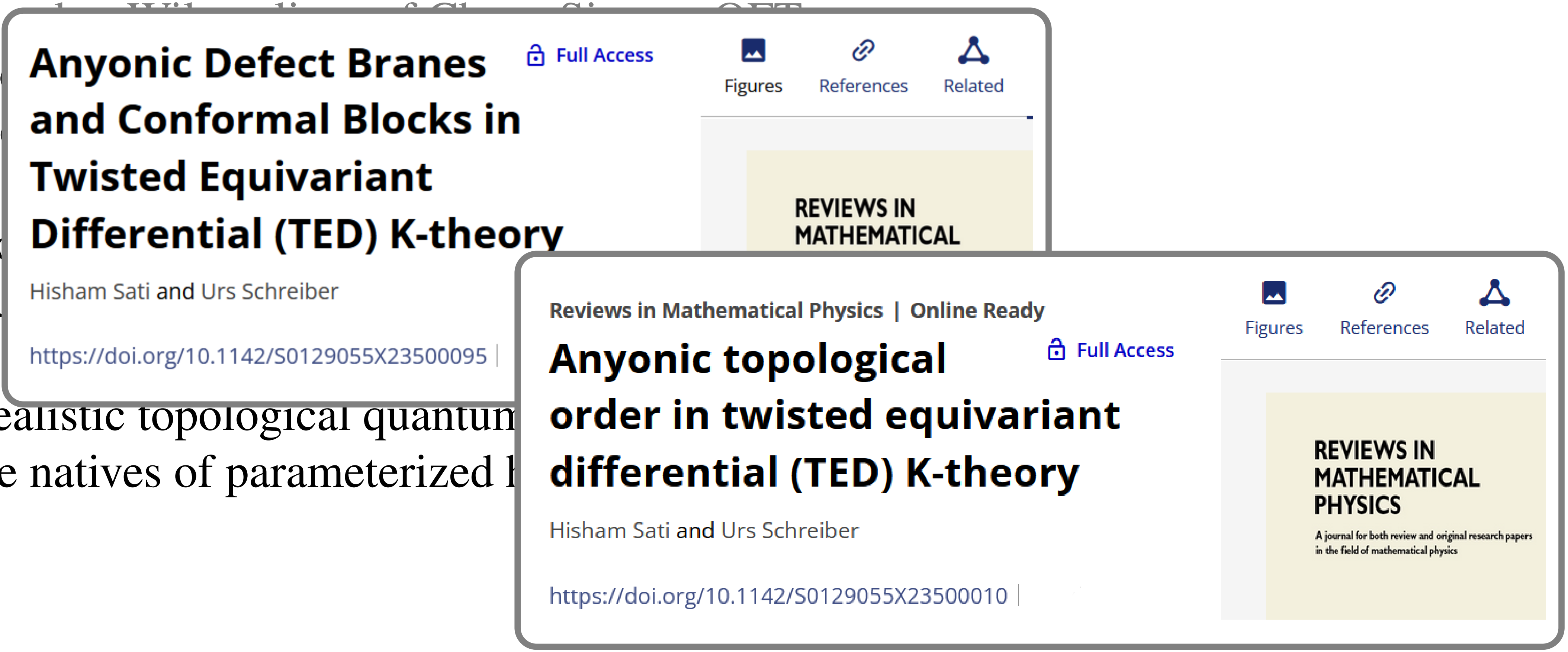
Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates

aka: anyon braiding in topological materials,
aka: monodromy of WZW-conformal blocks,

se
gr

N
Bu

Realistic topological quantum
are natives of parameterized



Anyonic Defect Branes and Conformal Blocks in Twisted Equivariant Differential (TED) K-theory

Hisham Sati and Urs Schreiber

https://doi.org/10.1142/S0129055X23500095

Reviews in Mathematical Physics | Online Ready

Anyonic topological order in twisted equivariant differential (TED) K-theory

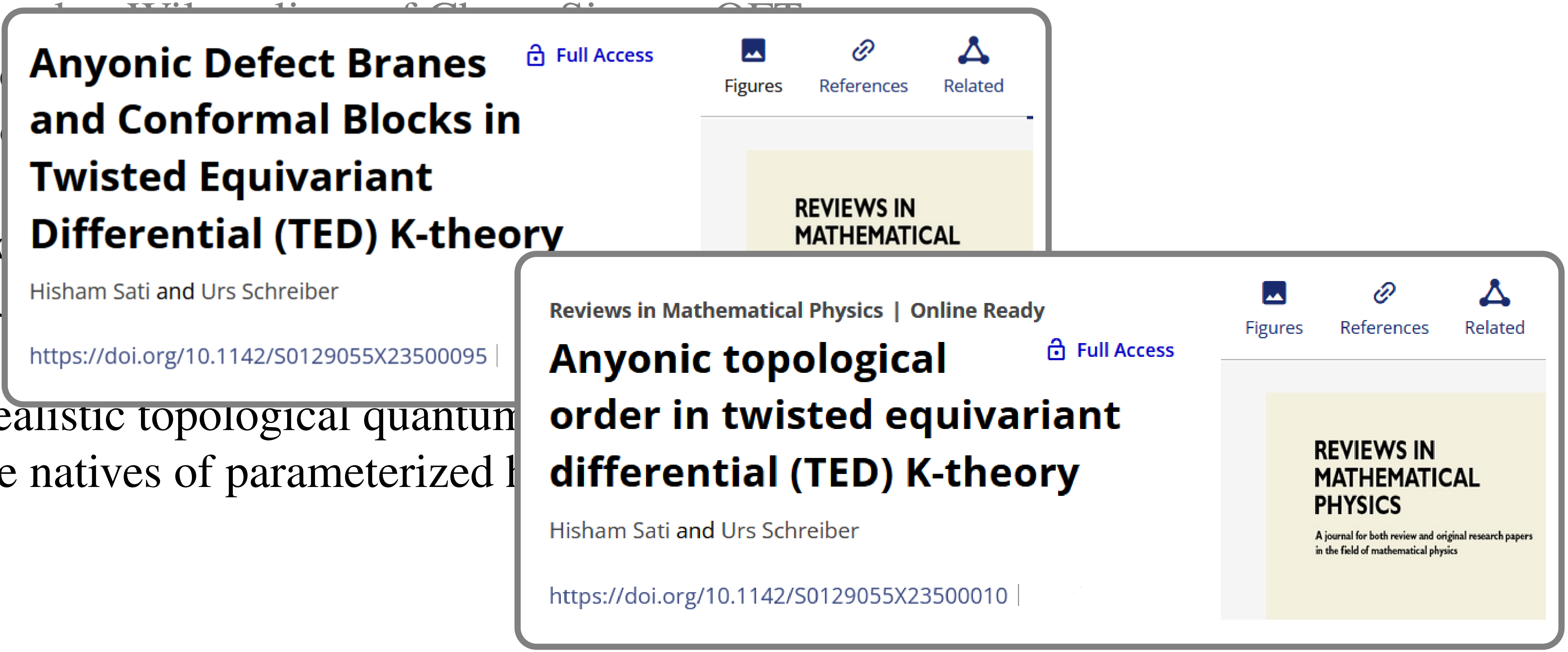
Hisham Sati and Urs Schreiber

https://doi.org/10.1142/S0129055X23500010

Towards verifying realistic topological quantum gates

Taken at traditional face value,
formally specifying & certifying
realistic topological quantum gates

aka: anyon braiding in topological materials,
aka: monodromy of WZW-conformal blocks,
aka: Wilson lines of Chern-Simons QFT



Anyonic Defect Branes and Conformal Blocks in Twisted Equivariant Differential (TED) K-theory

Hisham Sati and Urs Schreiber

https://doi.org/10.1142/S0129055X23500095

Reviews in Mathematical Physics | Online Ready

Anyonic topological order in twisted equivariant differential (TED) K-theory

Hisham Sati and Urs Schreiber

https://doi.org/10.1142/S0129055X23500010

se
gr

No
Bu

Realistic topological quantum
are natives of parameterized

with a slick formal specification in *homotopy-typed* programming languages

Towards verifying realistic topological quantum gates

Taken at tradition

formally specifyi

realistic topologi

aka: anyon br

aka: monodro

se

gr

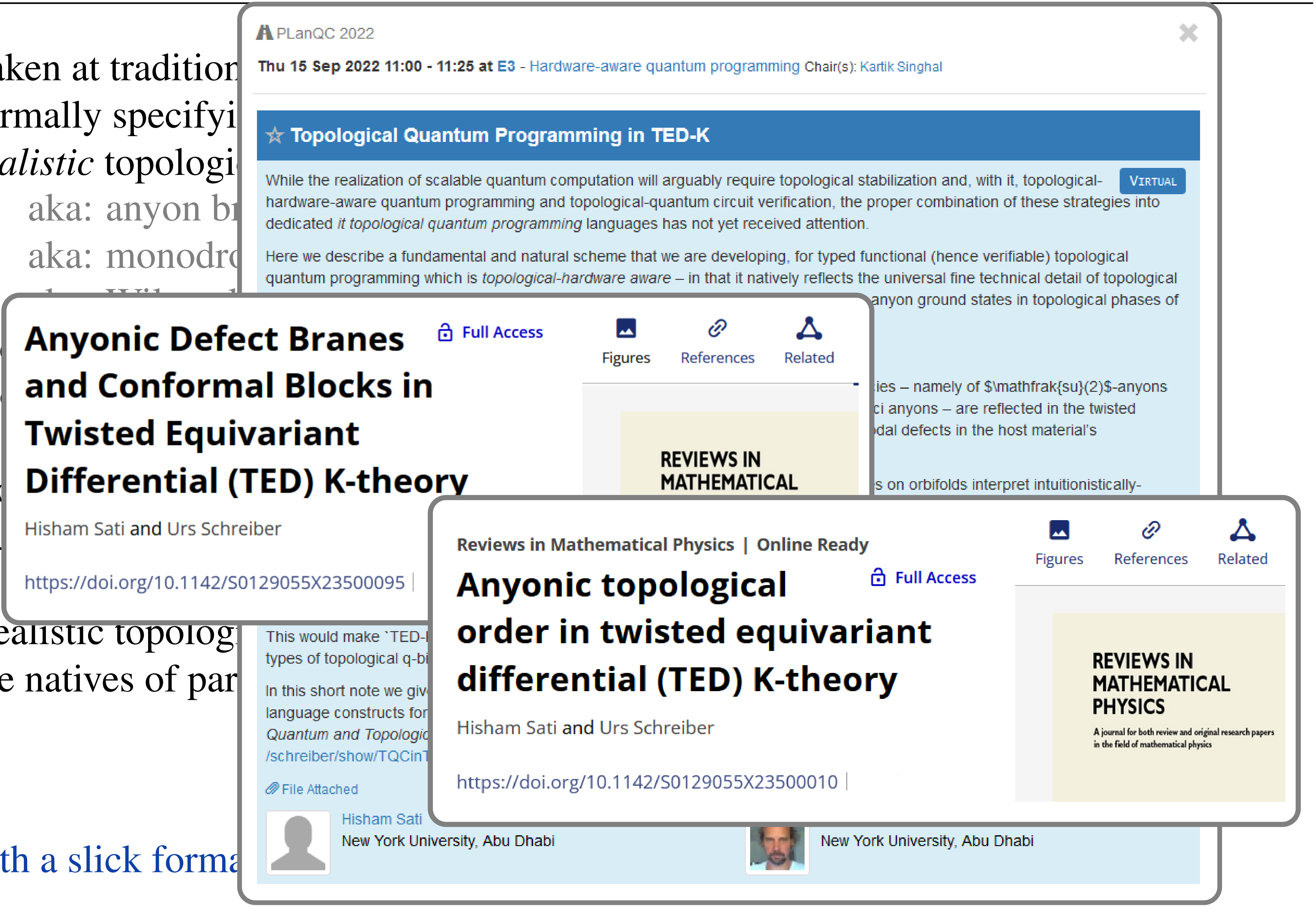
N

Bu

Realistic topolog

are natives of par

with a slick forma



PLanQC 2022

Thu 15 Sep 2022 11:00 - 11:25 at E3 - Hardware-aware quantum programming Chair(s): Kartik Singhal

☆ Topological Quantum Programming in TED-K

VIRTUAL

While the realization of scalable quantum computation will arguably require topological stabilization and, with it, topological-hardware-aware quantum programming and topological-quantum circuit verification, the proper combination of these strategies into dedicated *it topological quantum programming* languages has not yet received attention.

Here we describe a fundamental and natural scheme that we are developing, for typed functional (hence verifiable) topological quantum programming which is *topological-hardware aware* – in that it natively reflects the universal fine technical detail of topological

anyon ground states in topological phases of

ies – namely of $\mathfrak{su}(2)$-anyons

ci anyons – are reflected in the twisted

dal defects in the host material's

s on orbifolds interpret intuitionistically-

This would make `TED-

types of topological q-bi

In this short note we giv

language constructs for

Quantum and Topologic

/schreiber/show/TQCin

File Attached

Hisham Sati

New York University, Abu Dhabi

New York University, Abu Dhabi

Anyonic Defect Branes and Conformal Blocks in Twisted Equivariant Differential (TED) K-theory

Full Access

Figures References Related

REVIEWS IN MATHEMATICAL

Hisham Sati and Urs Schreiber

https://doi.org/10.1142/S0129055X23500095 |

Reviews in Mathematical Physics | Online Ready

Anyonic topological order in twisted equivariant differential (TED) K-theory

Full Access

Figures References Related

Hisham Sati and Urs Schreiber

https://doi.org/10.1142/S0129055X23500010 |

REVIEWS IN MATHEMATICAL PHYSICS

A journal for both review and original research papers in the field of mathematical physics

Towards verifying realistic topological quantum gates

Taken at tradition

formally specifyi

realistic topologi

aka: anyon br

aka: monodro

se

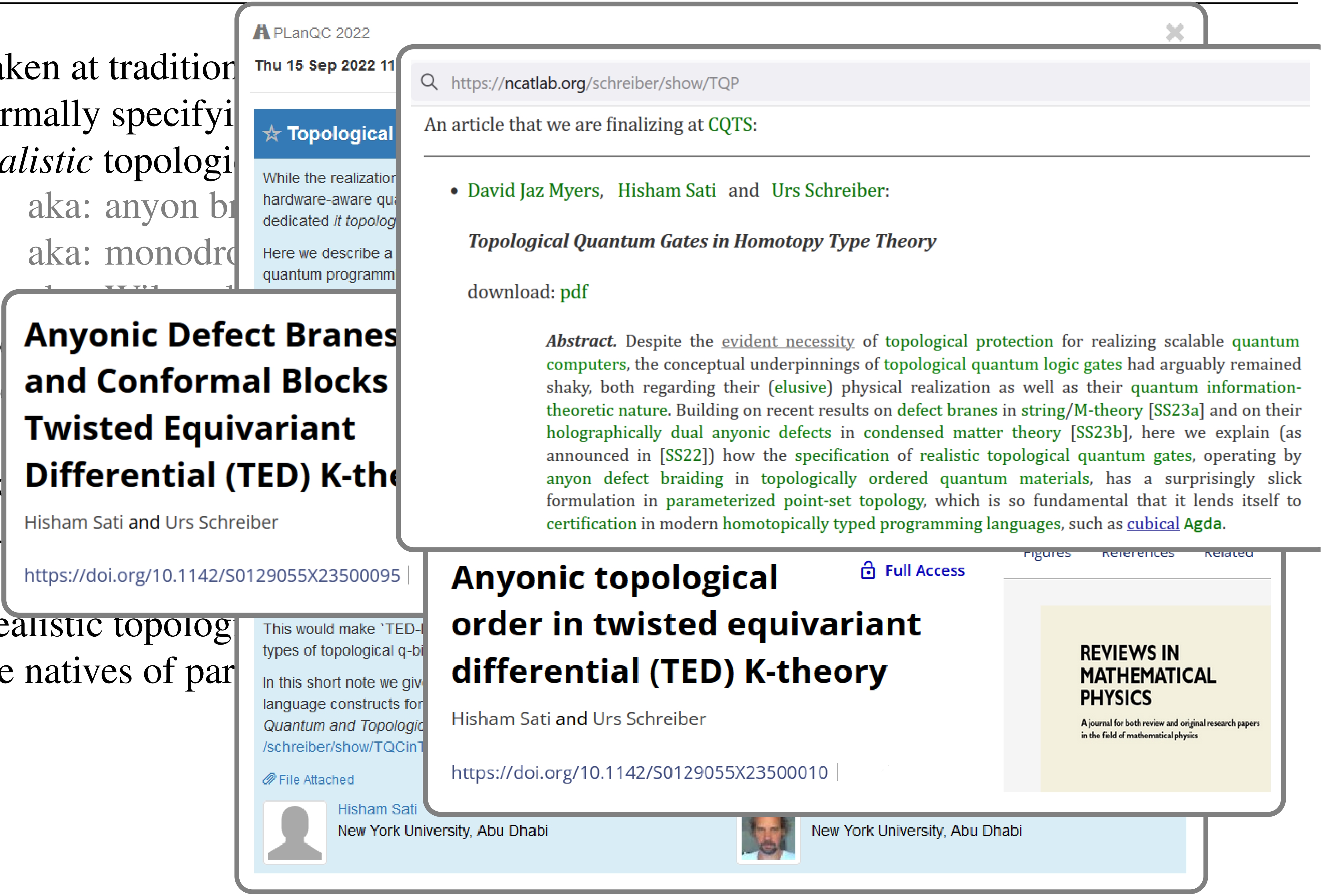
gr

N

Bu

Realistic topolog

are natives of par



PLanQC 2022

Thu 15 Sep 2022 11

☆ Topological

While the realization
hardware-aware qu
dedicated *it topolog*

Here we describe a
quantum programm

This would make `TED-K
types of topological q-bi

In this short note we giv
language constructs for
Quantum and Topologic
/schreiber/show/TQCinT

File Attached

Hisham Sati
New York University, Abu Dhabi

New York University, Abu Dhabi

https://ncatlab.org/schreiber/show/TQP

An article that we are finalizing at CQTS:

- David Jaz Myers, Hisham Sati and Urs Schreiber:

  Topological Quantum Gates in Homotopy Type Theory

  download: pdf

  Abstract. Despite the evident necessity of topological protection for realizing scalable quantum computers, the conceptual underpinnings of topological quantum logic gates had arguably remained shaky, both regarding their (elusive) physical realization as well as their quantum information-theoretic nature. Building on recent results on defect branes in string/M-theory [SS23a] and on their holographically dual anyonic defects in condensed matter theory [SS23b], here we explain (as announced in [SS22]) how the specification of realistic topological quantum gates, operating by anyon defect braiding in topologically ordered quantum materials, has a surprisingly slick formulation in parameterized point-set topology, which is so fundamental that it lends itself to certification in modern homotopically typed programming languages, such as cubical Agda.

Anyonic Defect Branes and Conformal Blocks Twisted Equivariant Differential (TED) K-the

Hisham Sati and Urs Schreiber

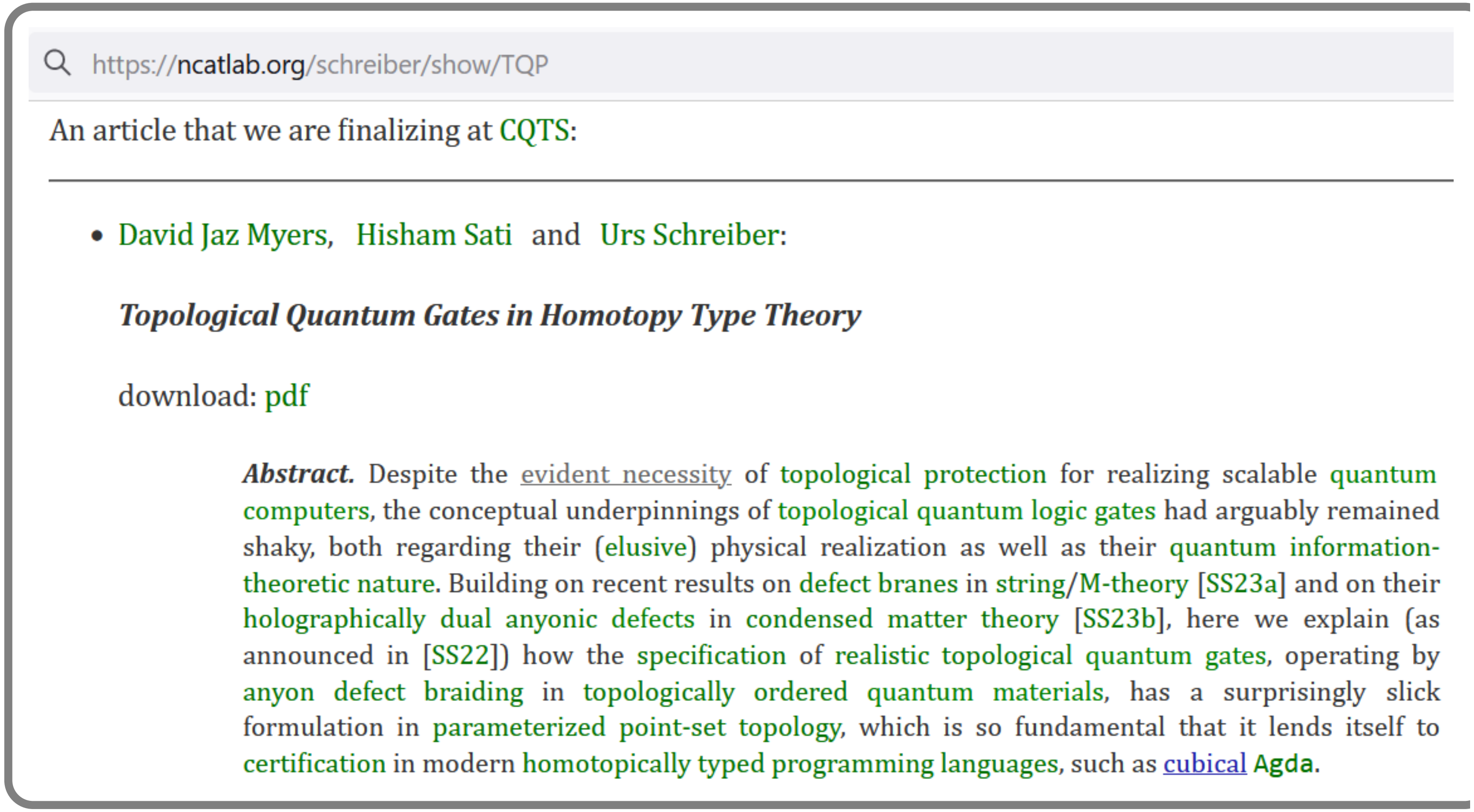
https://doi.org/10.1142/S0129055X23500095

Anyonic topological order in twisted equivariant differential (TED) K-theory

Full Access

Hisham Sati and Urs Schreiber

https://doi.org/10.1142/S0129055X23500010



https://ncatlab.org/schreiber/show/TQP

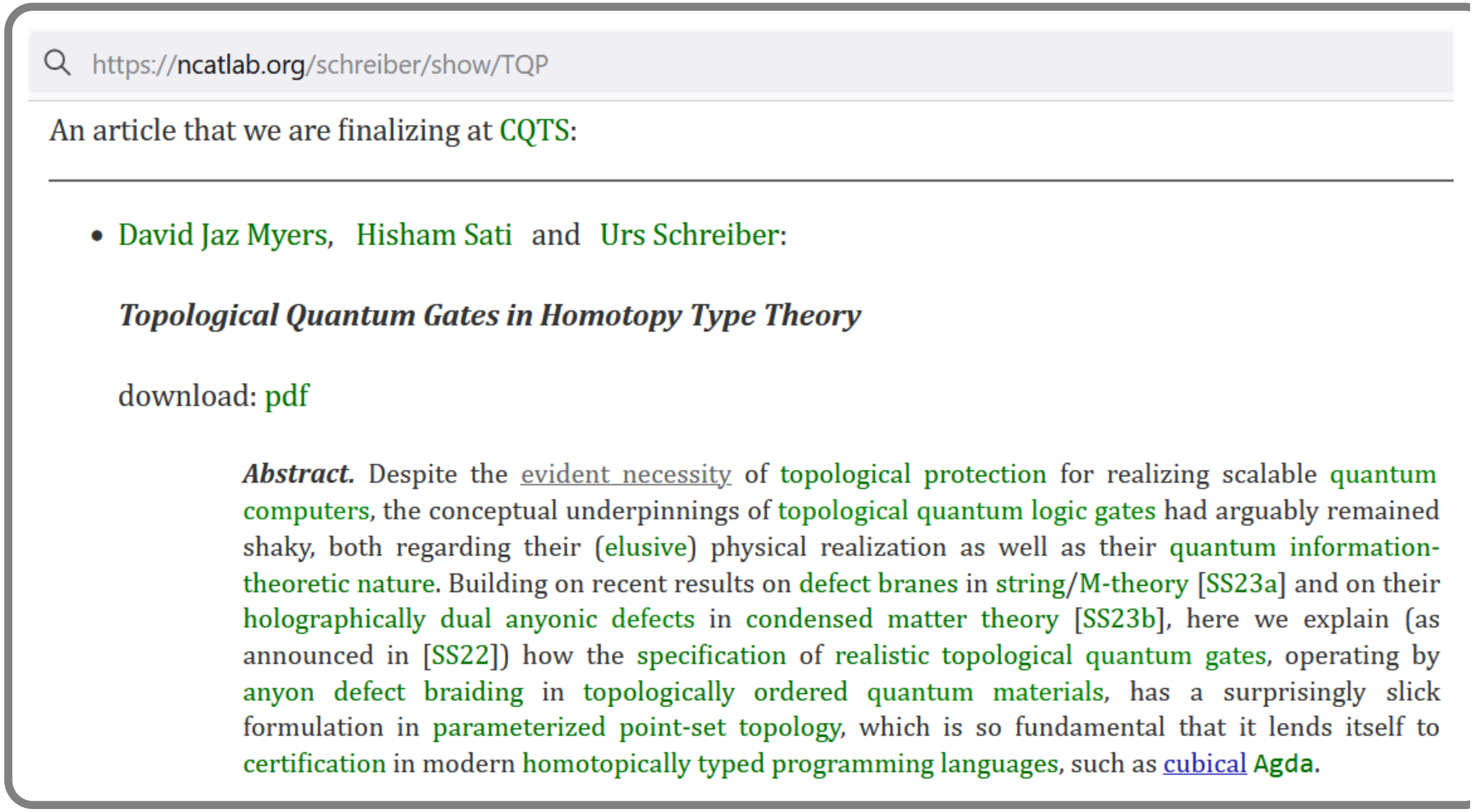
An article that we are finalizing at CQTS:

- David Jaz Myers, Hisham Sati and Urs Schreiber:

  Topological Quantum Gates in Homotopy Type Theory

  download: pdf

  > ***Abstract.*** Despite the evident necessity of topological protection for realizing scalable quantum computers, the conceptual underpinnings of topological quantum logic gates had arguably remained shaky, both regarding their (elusive) physical realization as well as their quantum information-theoretic nature. Building on recent results on defect branes in string/M-theory [SS23a] and on their holographically dual anyonic defects in condensed matter theory [SS23b], here we explain (as announced in [SS22]) how the specification of realistic topological quantum gates, operating by anyon defect braiding in topologically ordered quantum materials, has a surprisingly slick formulation in parameterized point-set topology, which is so fundamental that it lends itself to certification in modern homotopically typed programming languages, such as cubical `Agda`.



https://ncatlab.org/schreiber/show/TQP

An article that we are finalizing at CQTS:

- David Jaz Myers, Hisham Sati and Urs Schreiber:

  Topological Quantum Gates in Homotopy Type Theory

  download: pdf

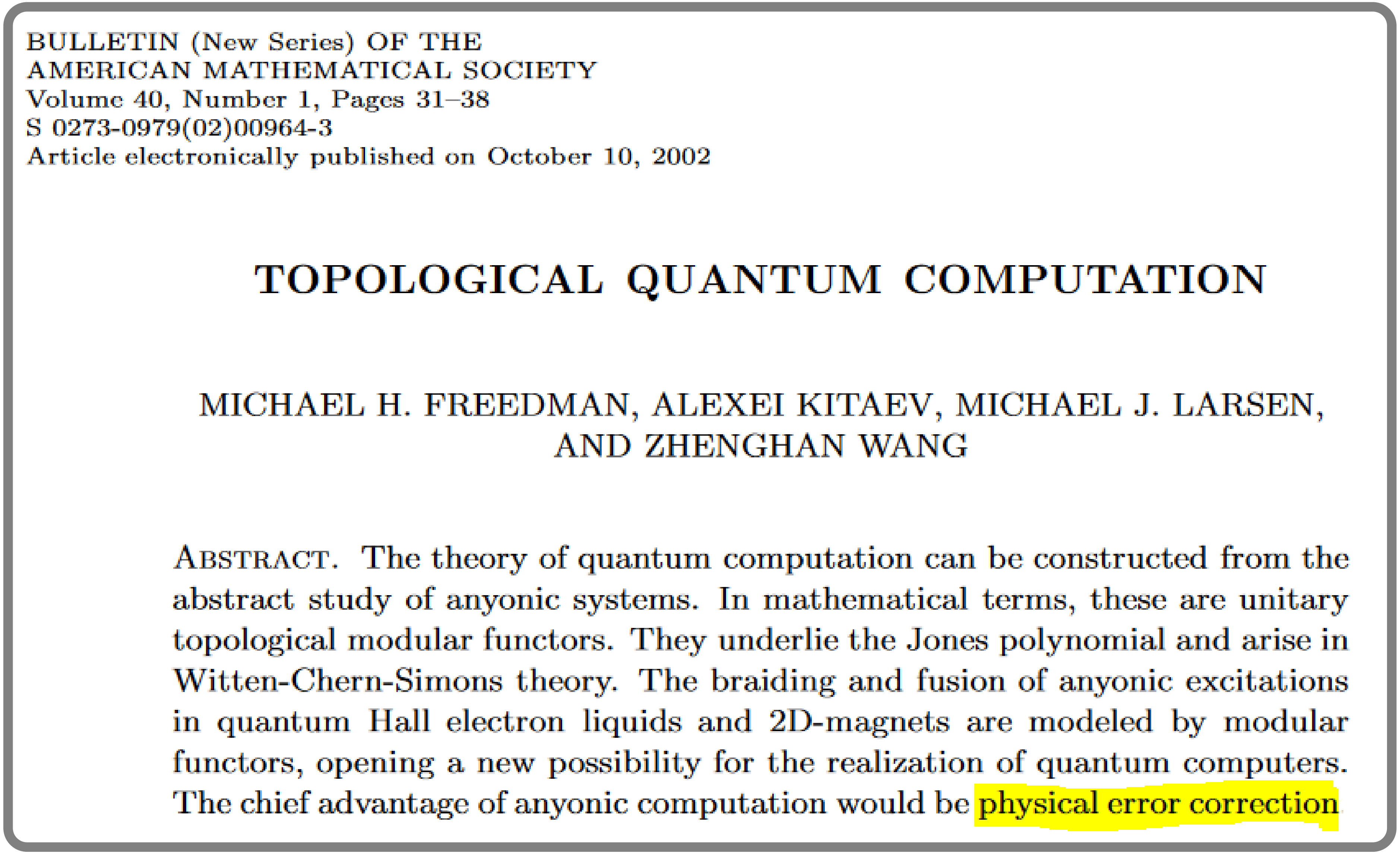
  Abstract. Despite the evident necessity of topological protection for realizing scalable quantum computers, the conceptual underpinnings of topological quantum logic gates had arguably remained shaky, both regarding their (elusive) physical realization as well as their quantum information-theoretic nature. Building on recent results on defect branes in string/M-theory [SS23a] and on their holographically dual anyonic defects in condensed matter theory [SS23b], here we explain (as announced in [SS22]) how the specification of realistic topological quantum gates, operating by anyon defect braiding in topologically ordered quantum materials, has a surprisingly slick formulation in parameterized point-set topology, which is so fundamental that it lends itself to certification in modern homotopically typed programming languages, such as cubical `Agda`.

Now to say all this in more detail ⟶

There are good arguments that
if Quantum Computation is to be a practical reality

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation



BULLETIN (New Series) OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 40, Number 1, Pages 31–38
S 0273-0979(02)00964-3
Article electronically published on October 10, 2002

TOPOLOGICAL QUANTUM COMPUTATION

MICHAEL H. FREEDMAN, ALEXEI KITAEV, MICHAEL J. LARSEN,
AND ZHENGHAN WANG

Abstract. The theory of quantum computation can be constructed from the abstract study of anyonic systems. In mathematical terms, these are unitary topological modular functors. They underlie the Jones polynomial and arise in Witten-Chern-Simons theory. The braiding and fusion of anyonic excitations in quantum Hall electron liquids and 2D-magnets are modeled by modular functors, opening a new possibility for the realization of quantum computers. The chief advantage of anyonic computation would be physical error correction

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation

Das Sarma, MIT Tech Rev (2022):

“*The quantum-bit systems we have today are a tremendous scientific achievement,*

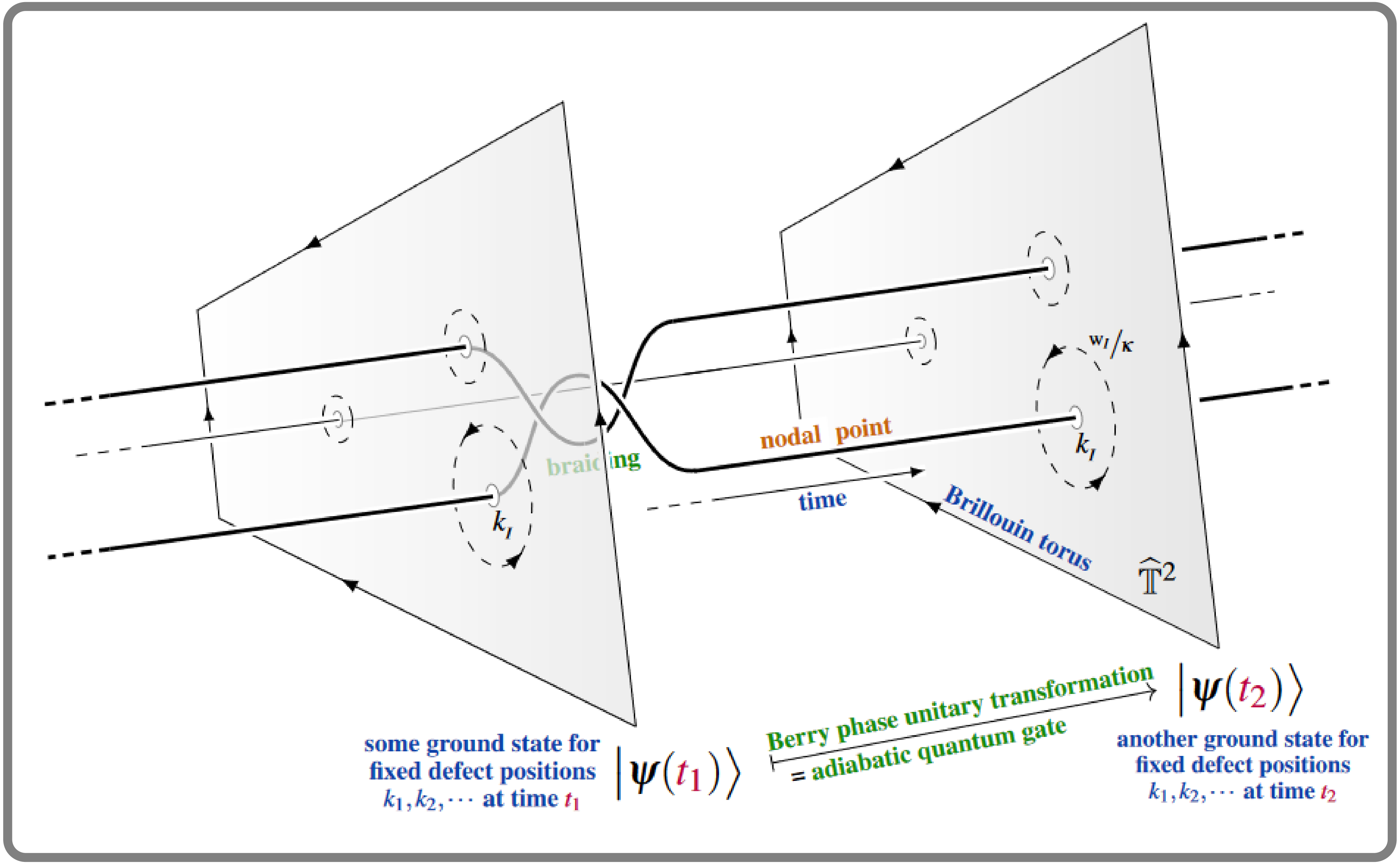
but they take us no closer to having a quantum computer that can solve a problem that anybody cares about.

What is missing is the breakthrough bypassing quantum error correction by using far-more-stable quantum-bits, in an approach called topological quantum computing.”

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,



some ground state for fixed defect positions $k_1, k_2, \cdots$ at time t_1 $\left|\psi(t_1)\right\rangle$

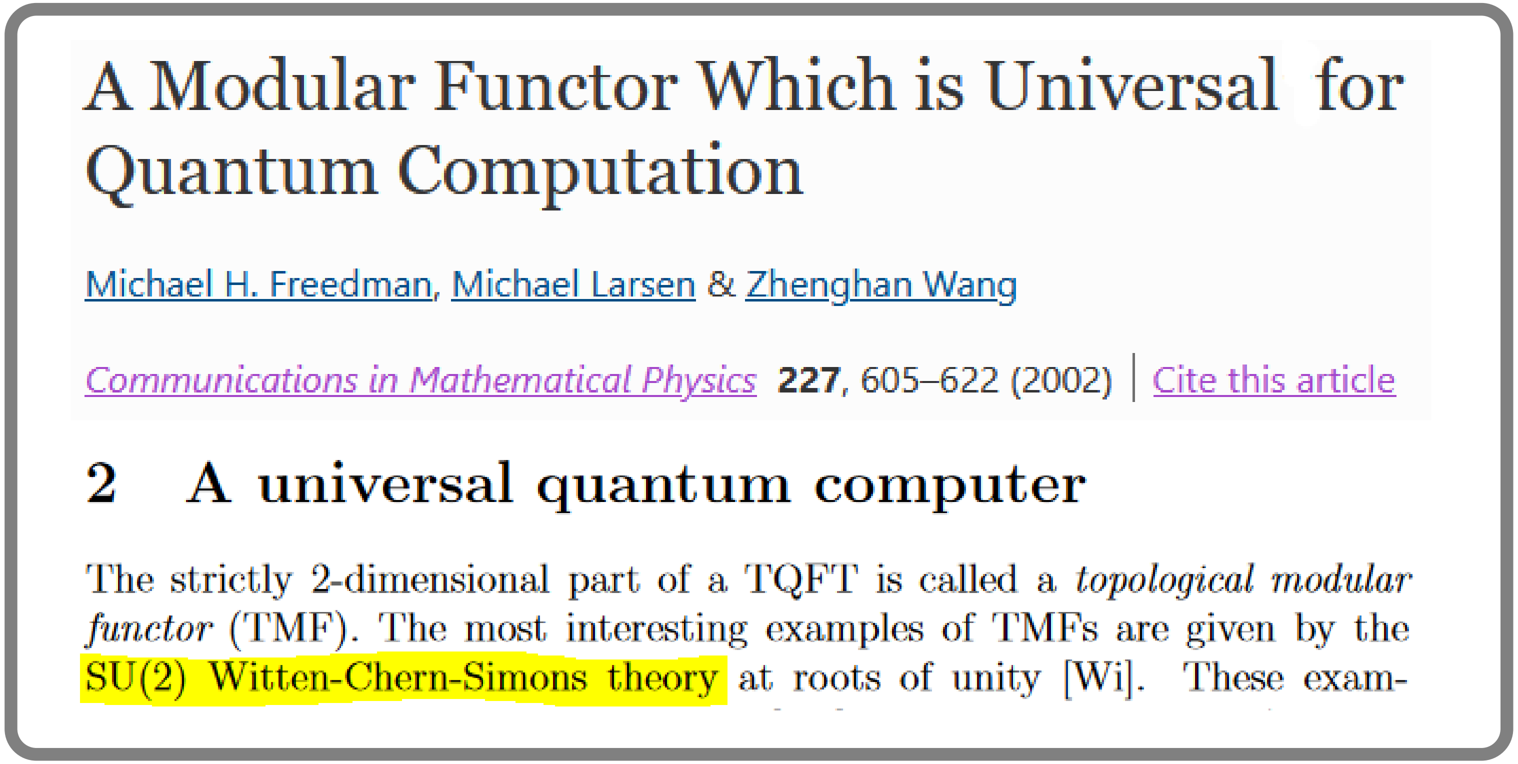
Berry phase unitary transformation = adiabatic quantum gate

$\left|\psi(t_2)\right\rangle$ another ground state for fixed defect positions $k_1, k_2, \cdots$ at time t_2

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).



A Modular Functor Which is Universal for Quantum Computation

Michael H. Freedman, Michael Larsen & Zhenghan Wang

Communications in Mathematical Physics **227**, 605–622 (2002) | Cite this article

2 A universal quantum computer

The strictly 2-dimensional part of a TQFT is called a *topological modular functor* (TMF). The most interesting examples of TMFs are given by the SU(2) Witten-Chern-Simons theory at roots of unity [Wi]. These exam-

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

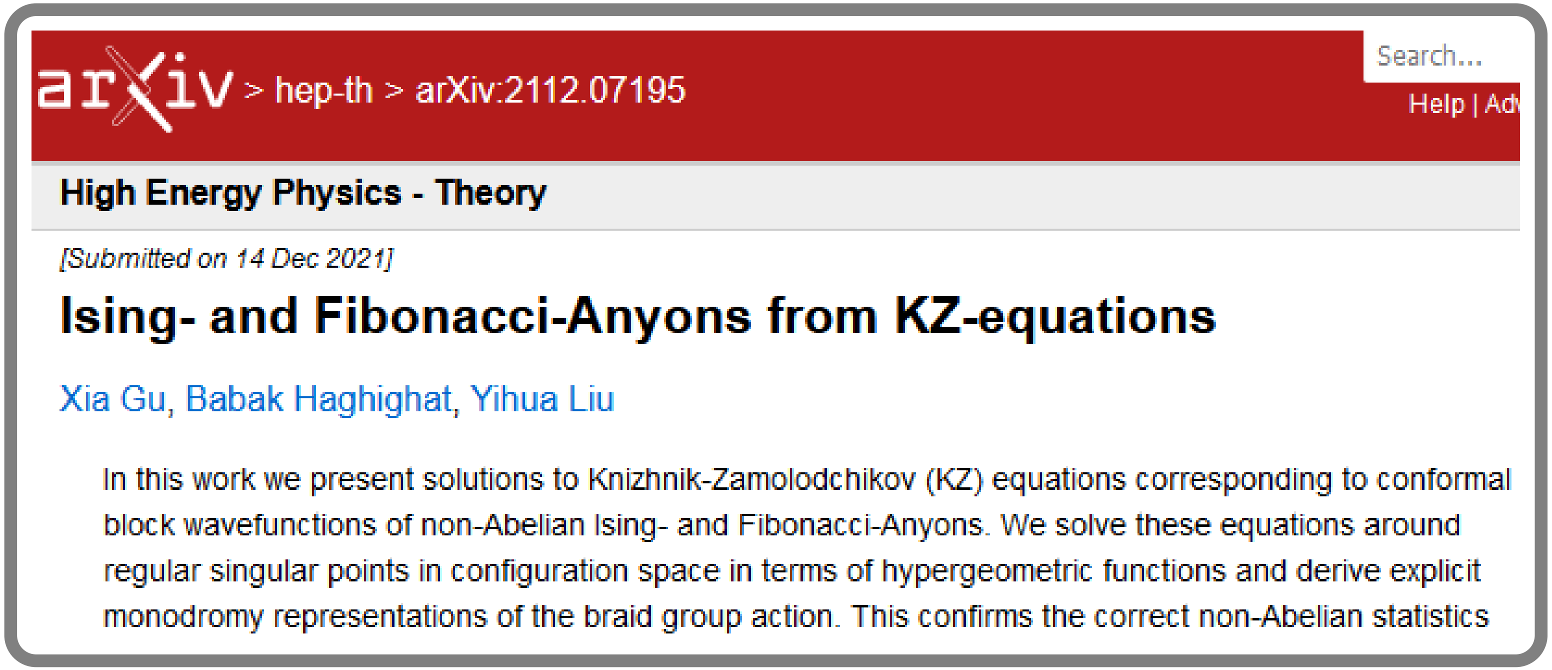
Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection
on bundles of conformal blocks of the

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection
on bundles of conformal blocks of the
chiral $\mathfrak{su}(2)$ WZW model CFT.

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection
on bundles of conformal blocks of the
chiral $\mathfrak{su}(2)$ WZW model CFT.



High Energy Physics - Theory

[Submitted on 14 Dec 2021]

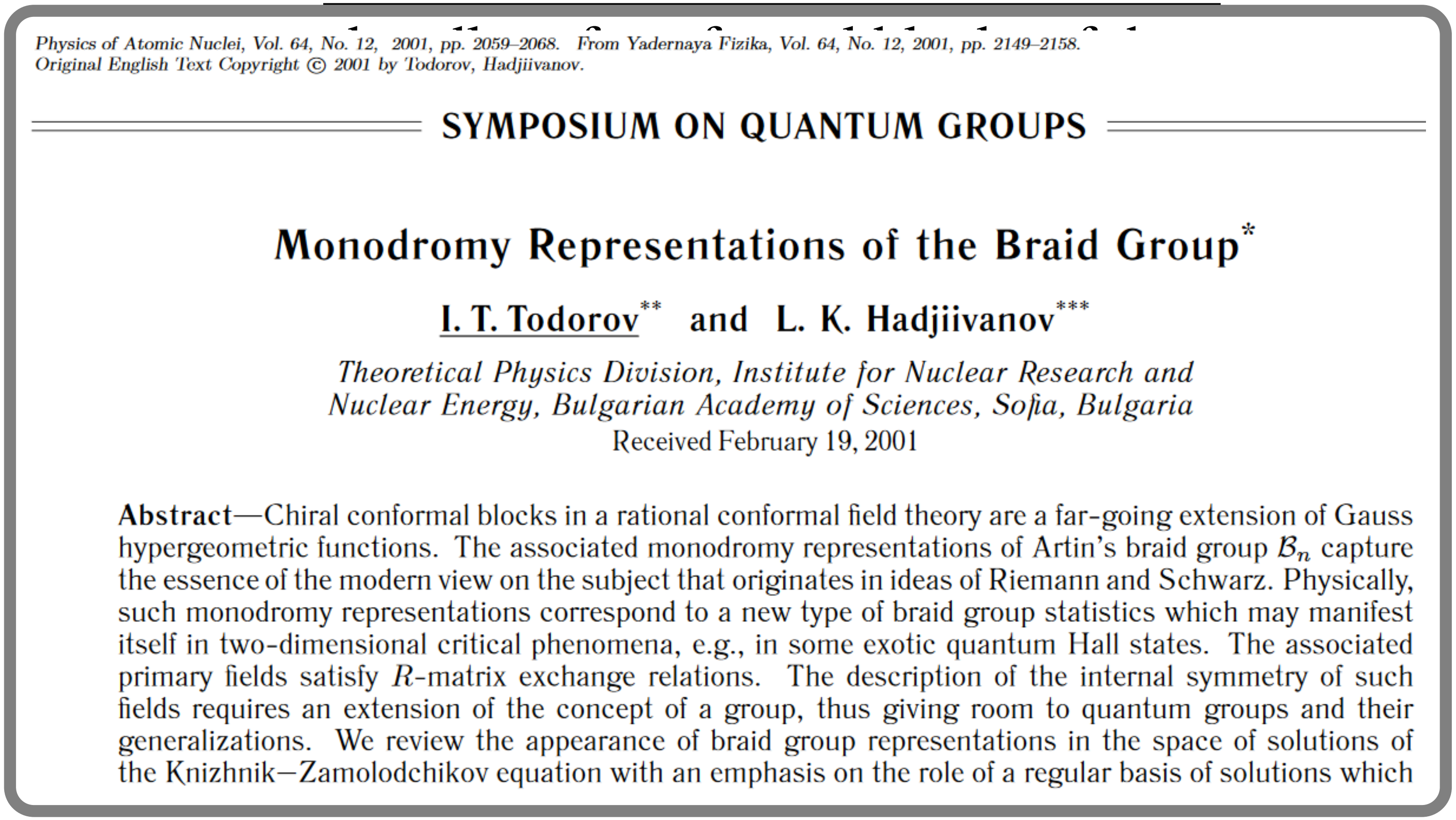
Ising- and Fibonacci-Anyons from KZ-equations

Xia Gu, Babak Haghighat, Yihua Liu

In this work we present solutions to Knizhnik-Zamolodchikov (KZ) equations corresponding to conformal block wavefunctions of non-Abelian Ising- and Fibonacci-Anyons. We solve these equations around regular singular points in configuration space in terms of hypergeometric functions and derive explicit monodromy representations of the braid group action. This confirms the correct non-Abelian statistics

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection



Physics of Atomic Nuclei, Vol. 64, No. 12, 2001, pp. 2059–2068. From Yadernaya Fizika, Vol. 64, No. 12, 2001, pp. 2149–2158.
Original English Text Copyright © 2001 by Todorov, Hadjiivanov.

SYMPOSIUM ON QUANTUM GROUPS

Monodromy Representations of the Braid Group*

I. T. Todorov** and **L. K. Hadjiivanov*****

Theoretical Physics Division, Institute for Nuclear Research and Nuclear Energy, Bulgarian Academy of Sciences, Sofia, Bulgaria

Received February 19, 2001

Abstract—Chiral conformal blocks in a rational conformal field theory are a far-going extension of Gauss hypergeometric functions. The associated monodromy representations of Artin's braid group $\mathcal{B}_n$ capture the essence of the modern view on the subject that originates in ideas of Riemann and Schwarz. Physically, such monodromy representations correspond to a new type of braid group statistics which may manifest itself in two-dimensional critical phenomena, e.g., in some exotic quantum Hall states. The associated primary fields satisfy R-matrix exchange relations. The description of the internal symmetry of such fields requires an extension of the concept of a group, thus giving room to quantum groups and their generalizations. We review the appearance of braid group representations in the space of solutions of the Knizhnik–Zamolodchikov equation with an emphasis on the role of a regular basis of solutions which

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection
on bundles of conformal blocks of the
chiral $\mathfrak{su}(2)$ WZW model CFT.

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

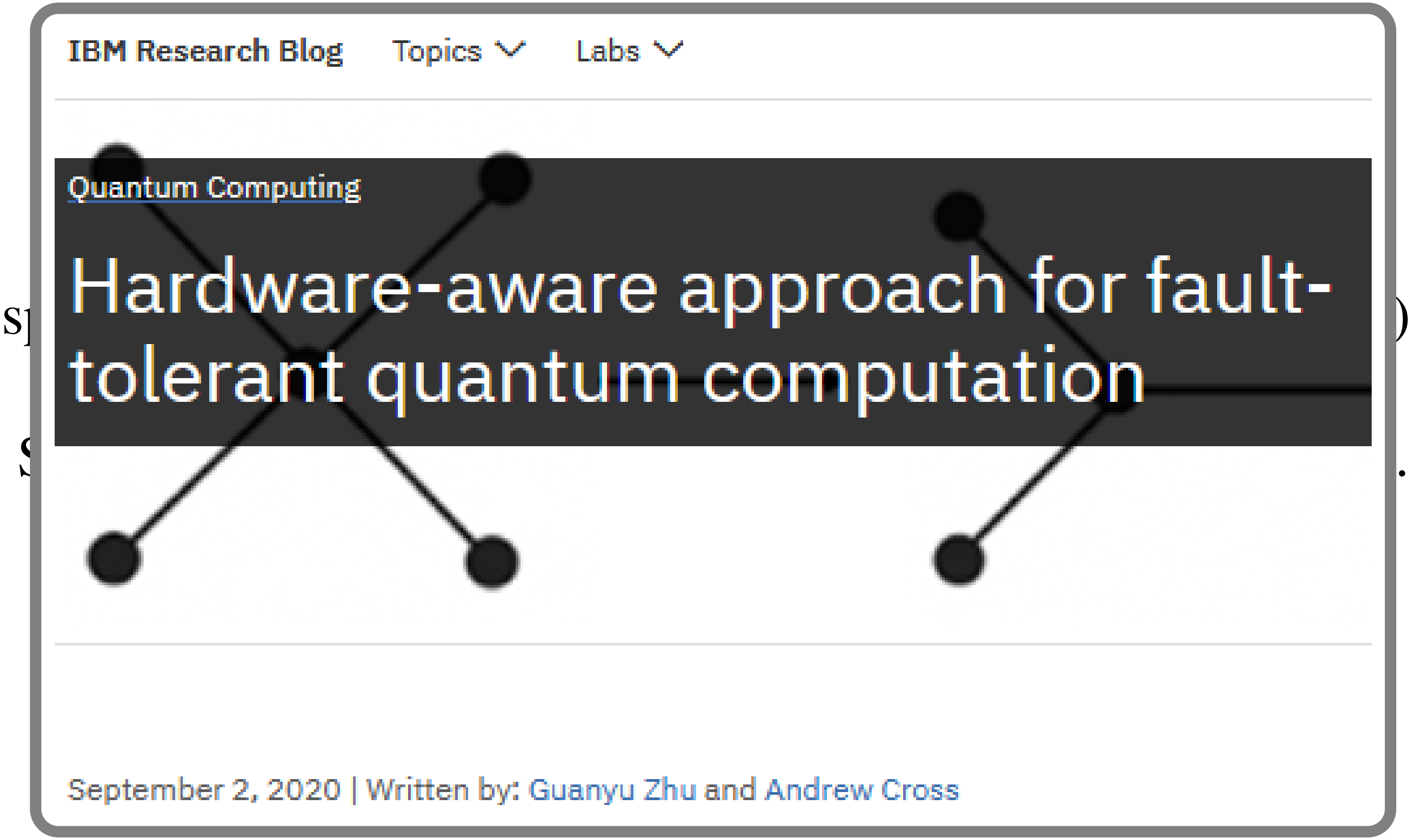
Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection
on bundles of conformal blocks of the
chiral $\mathfrak{su}(2)$ WZW model CFT.

Efficient programming of topological quantum computers

There are good arguments that if Quantum Computation is to be a practical reality then in the form of Topological Quantum Computation with quantum gates given by adiabatic braiding of anyons, specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates. Mathematically, they are the monodromy of the Knizhnik-Zamolodchikov connection on bundles of conformal blocks of the chiral $\mathfrak{su}(2)$ WZW model CFT.

Efficient programming of topological quantum computers must be *aware* of this peculiar $\mathfrak{su}(2)$-braid gate hardware.



Quantum Computing

Hardware-aware approach for fault-tolerant quantum computation

September 2, 2020 | Written by: Guanyu Zhu and Andrew Cross

Efficient programming of topological quantum computers must be *aware* of this peculiar $\mathfrak{su}(2)$-braid gate hardware.

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection
on bundles of conformal blocks of the
chiral $\mathfrak{su}(2)$ WZW model CFT.

Efficient programming of topological quantum computers
must be *aware* of this peculiar $\mathfrak{su}(2)$-braid gate hardware.

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection
on bundles of conformal blocks of the
chiral $\mathfrak{su}(2)$ WZW model CFT.

Efficient programming of topological quantum computers
must be *aware* of this peculiar $\mathfrak{su}(2)$-braid gate hardware.

However, bundles of conformal blocks superficially
appear to be a convoluted mathematical structure,

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection
on bundles of conformal blocks of the
chiral $\mathfrak{su}(2)$ WZW model CFT.

Efficient programming of topological quantum computers
must be *aware* of this peculiar $\mathfrak{su}(2)$-braid gate hardware.

However, bundles of conformal blocks superficially
appear to be a convoluted mathematical structure,
hardly suitable as a foundation for quantum programming.

There are good arguments that if Quantum Computation is to be a practical reality then in the form of Topological Quantum Computation with quantum gates given by adiabatic braiding of anyons, specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates. Mathematically, they are the monodromy of the Knizhnik-Zamolodchikov connection on bundles of conformal blocks of the chiral $\mathfrak{su}(2)$ WZW model CFT.

Efficient programming of topological quantum computers must be *aware* of this peculiar $\mathfrak{su}(2)$-braid gate hardware.

However, bundles of conformal blocks superficially *appear* to be a convoluted mathematical structure, hardly suitable as a foundation for quantum programming.

We show that the opposite is the case.

There are good arguments that
if Quantum Computation is to be a practical reality
then in the form of Topological Quantum Computation
with quantum gates given by adiabatic braiding of anyons,
specifically of $\mathfrak{su}(2)$-anyons (Ising/Majorana-, Fibonacci-, ...).

Such *braid gates* are rather special among all quantum gates.
Mathematically, they are the monodromy of
the Knizhnik-Zamolodchikov connection
on bundles of conformal blocks of the
chiral $\mathfrak{su}(2)$ WZW model CFT.

Efficient programming of topological quantum computers
must be *aware* of this peculiar $\mathfrak{su}(2)$-braid gate hardware.

However, bundles of conformal blocks superficially
appear to be a convoluted mathematical structure,
hardly suitable as a foundation for quantum programming.

We show that the opposite is the case.

Programming languages suited for describing *bundles* are *dependently typed*

Programming languages suited for describing *bundles* are *dependently typed*

system of X-dependent types

$$X \in \mathrm{Types}, \quad x \in X \qquad\qquad \vdash \qquad\qquad P(x) \in \mathrm{Types}$$

Programming languages suited for describing *bundles* are *dependently typed*

$$\begin{tikzcd} X \arrow[rr, "\overset{\textbf{classifying map}}{\vdash P}"] & & \mathrm{Types} \end{tikzcd}$$

$$X \in \mathrm{Types}, \quad x \in X \qquad \vdash \qquad \overset{\textbf{system of } X\textbf{-dependent types}}{P(x)} \in \mathrm{Types}$$

Programming languages suited for describing *bundles* are *dependently typed*

$$\begin{tikzcd}
 & \widehat{\mathrm{Types}} \arrow[d, "\text{univalent type bundle}"] \\
X \arrow[r, "\overset{\textbf{classifying map}}{\vdash P}"] & \mathrm{Types}
\end{tikzcd}$$

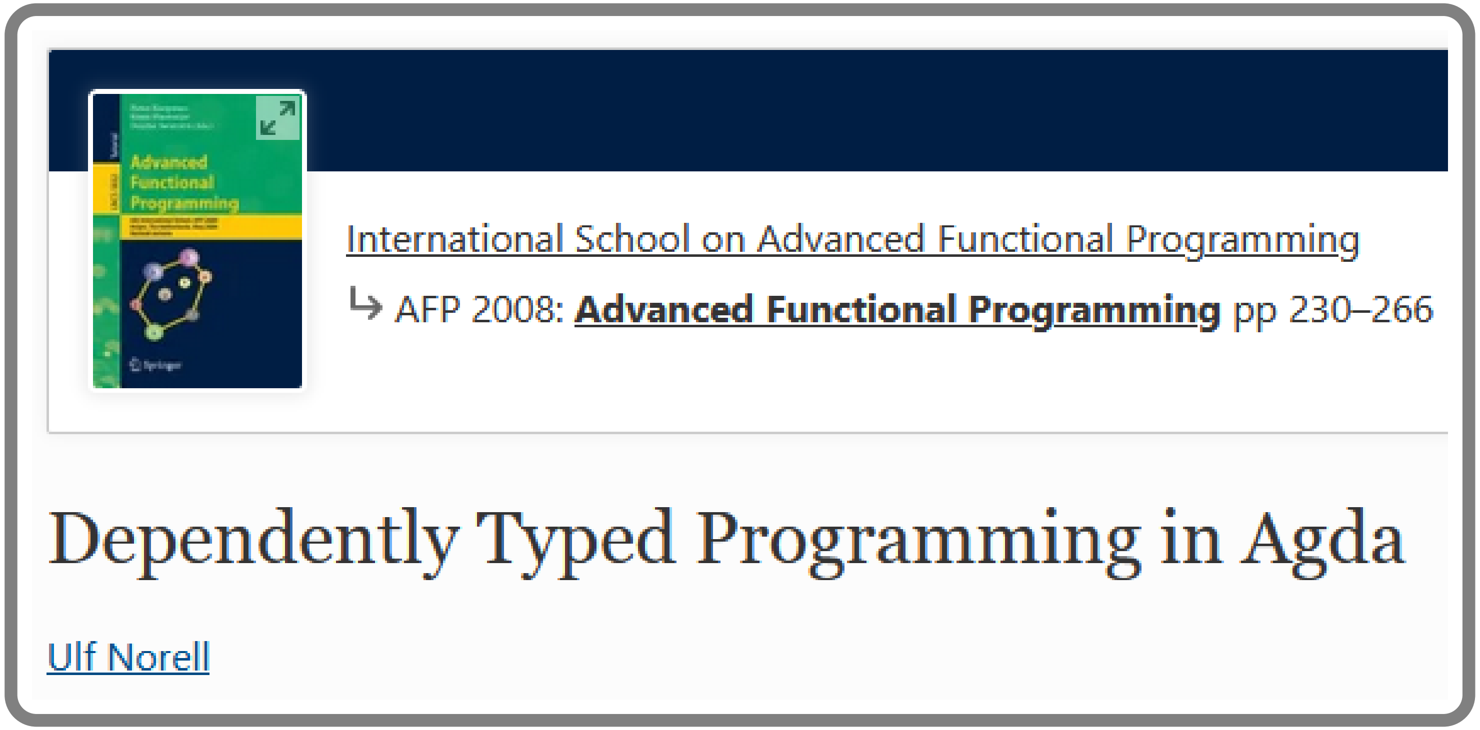
$$X \in \mathrm{Types}, \quad x \in X \qquad \vdash \qquad \overset{\textbf{system of } X\textbf{-dependent types}}{P(x)} \in \mathrm{Types}$$

Programming languages suited for describing *bundles* are *dependently typed*

$$\begin{tikzcd}
P \arrow[r] \arrow[d, "\text{bundle of } X\text{-dependent types}"'] \arrow[dr, phantom, "\text{(pb)}"] & \widehat{\mathrm{Types}} \arrow[d, "\text{univalent type bundle}"] \\
X \arrow[r, "\overset{\text{classifying map}}{\vdash P}"] & \mathrm{Types}
\end{tikzcd}$$

$$X \in \mathrm{Types}, \quad x \in X \qquad \vdash \qquad \overset{\text{system of } X\text{-dependent types}}{P(x) \in \mathrm{Types}}$$

Programming languages suited for describing *bundles* are *dependently typed*



International School on Advanced Functional Programming

↳ AFP 2008: **Advanced Functional Programming** pp 230–266

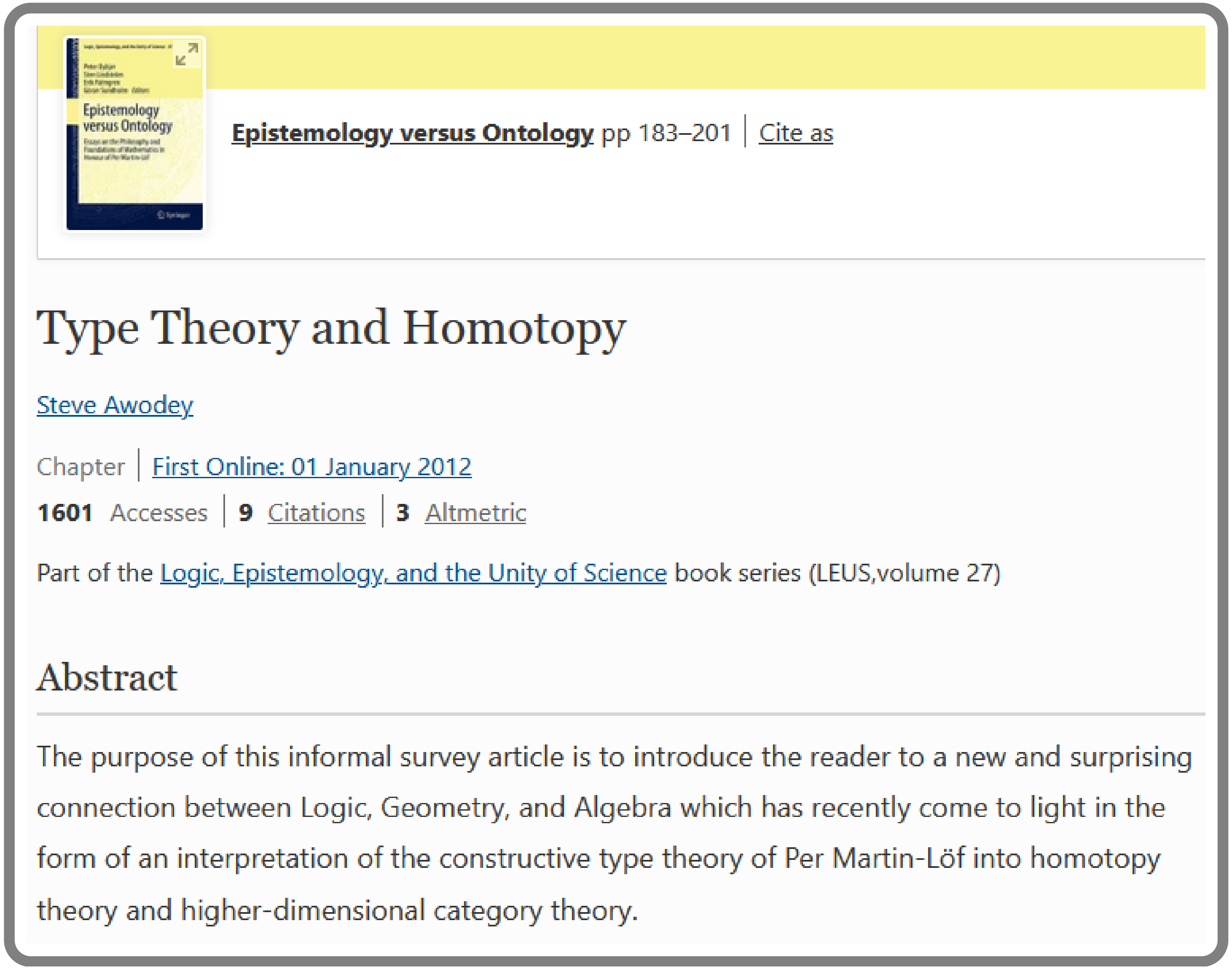
Dependently Typed Programming in Agda

Ulf Norell

Programming languages suited for describing
bundles are *dependently typed*

Programming languages suited for describing
bundles are *dependently typed*
and those which moreover describe
monodromy are *homotopically typed*.

Programming languages suited for describing
bundles are *dependently typed*
and those which moreover describe
monodromy are *homotopically typed*.



Epistemology versus Ontology pp 183–201 | Cite as

Type Theory and Homotopy

Steve Awodey

Chapter | First Online: 01 January 2012

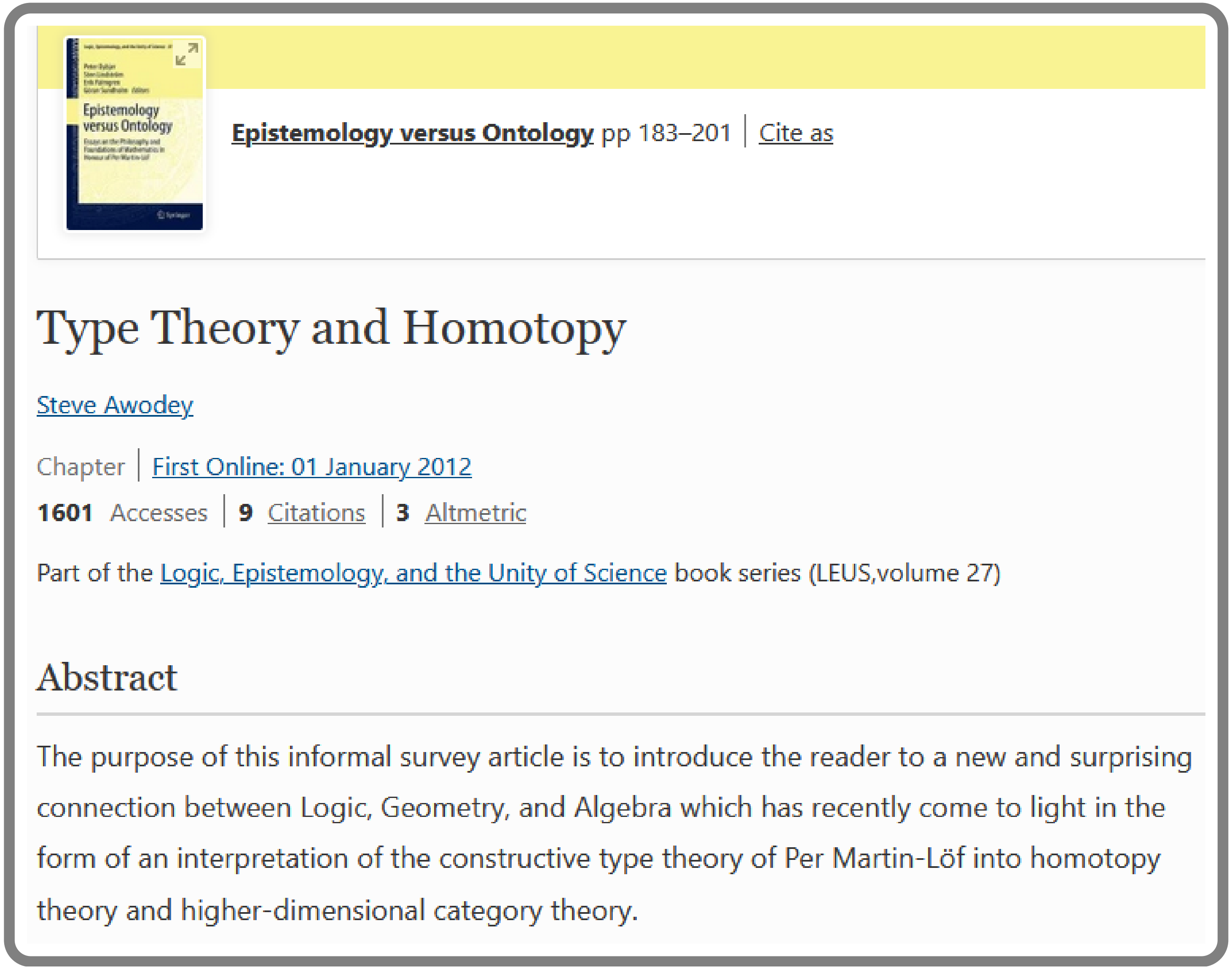
1601 Accesses | **9** Citations | **3** Altmetric

Part of the Logic, Epistemology, and the Unity of Science book series (LEUS,volume 27)

Abstract

The purpose of this informal survey article is to introduce the reader to a new and surprising connection between Logic, Geometry, and Algebra which has recently come to light in the form of an interpretation of the constructive type theory of Per Martin-Löf into homotopy theory and higher-dimensional category theory.

In HoTT, data types come with *paths* between their terms



Epistemology versus Ontology pp 183–201 | Cite as

Type Theory and Homotopy

Steve Awodey

Chapter | First Online: 01 January 2012

1601 Accesses | **9** Citations | **3** Altmetric

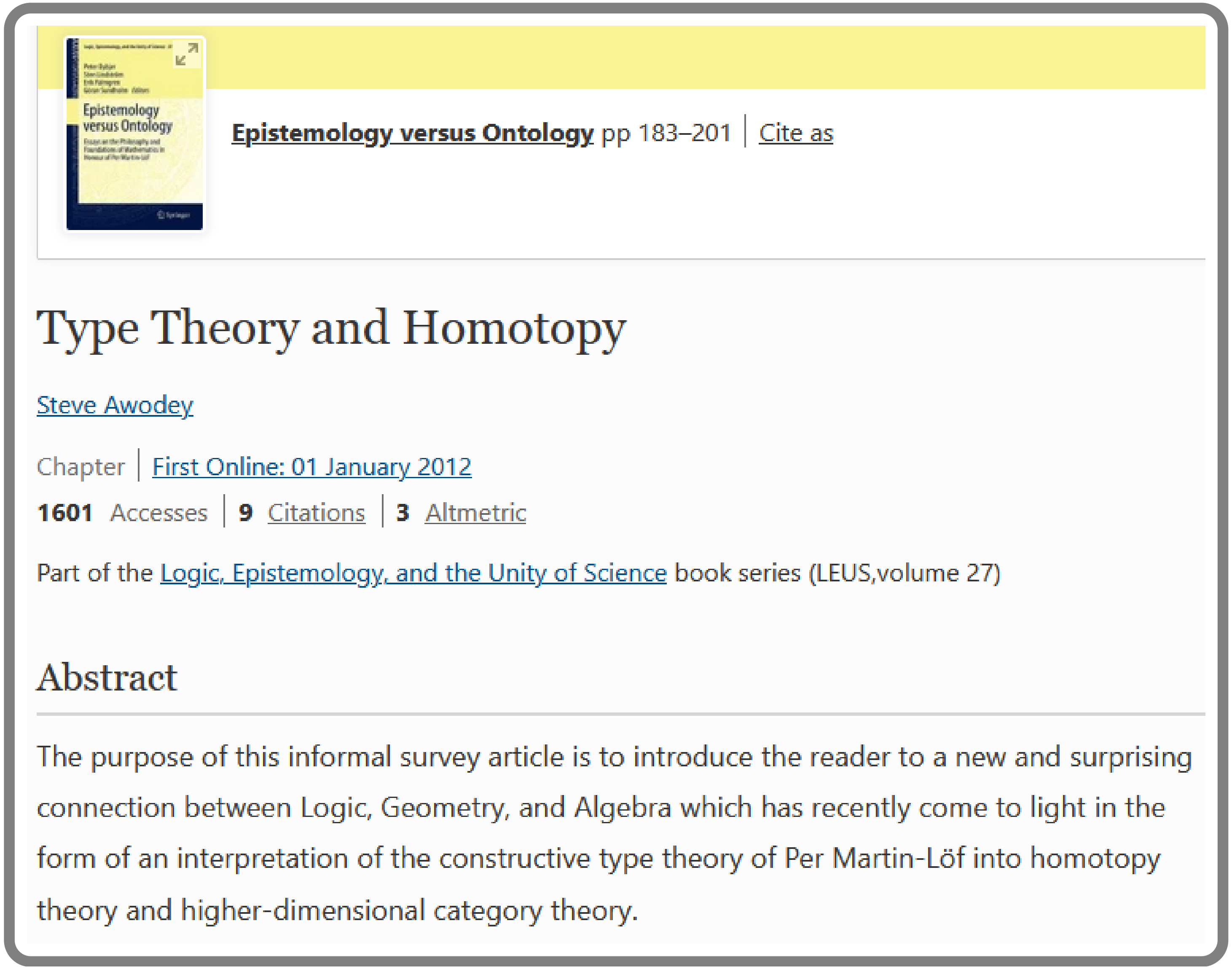
Part of the Logic, Epistemology, and the Unity of Science book series (LEUS,volume 27)

Abstract

The purpose of this informal survey article is to introduce the reader to a new and surprising connection between Logic, Geometry, and Algebra which has recently come to light in the form of an interpretation of the constructive type theory of Per Martin-Löf into homotopy theory and higher-dimensional category theory.

In HoTT, data types come with paths between their terms

$$\begin{array}{rl} X & \in \mathrm{Types} \\ x, y & \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}$$



Epistemology versus Ontology pp 183–201 | Cite as

Type Theory and Homotopy

Steve Awodey

Chapter | First Online: 01 January 2012

1601 Accesses | **9** Citations | **3** Altmetric

Part of the Logic, Epistemology, and the Unity of Science book series (LEUS,volume 27)

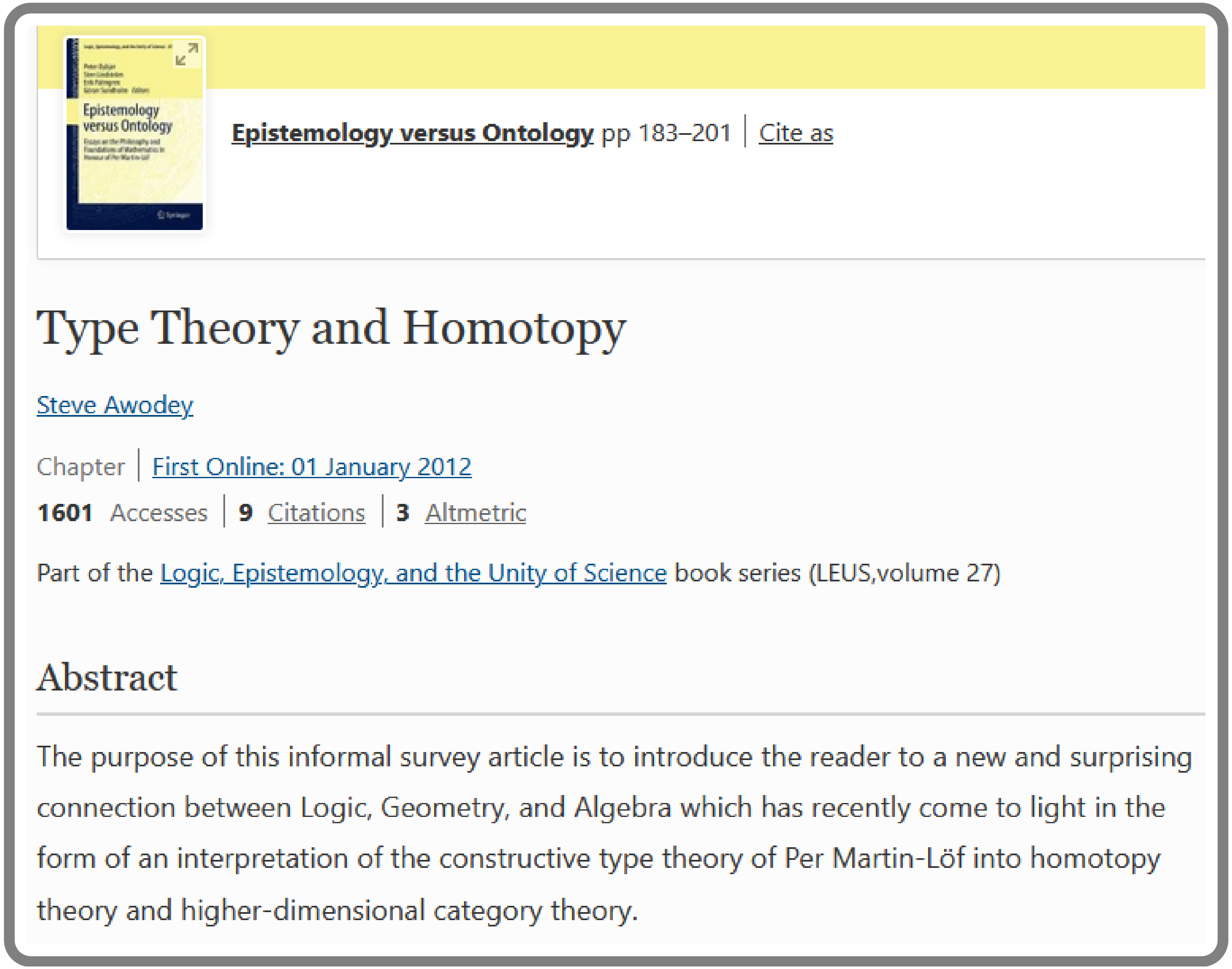
Abstract

The purpose of this informal survey article is to introduce the reader to a new and surprising connection between Logic, Geometry, and Algebra which has recently come to light in the form of an interpretation of the constructive type theory of Per Martin-Löf into homotopy theory and higher-dimensional category theory.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \in \mathrm{Types} \\ x, y \in X \end{array} \;\;\vdash\;\; \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.



Epistemology versus Ontology pp 183–201 | Cite as

Type Theory and Homotopy

Steve Awodey

Chapter | First Online: 01 January 2012

1601 Accesses | **9** Citations | **3** Altmetric

Part of the Logic, Epistemology, and the Unity of Science book series (LEUS,volume 27)

Abstract

The purpose of this informal survey article is to introduce the reader to a new and surprising connection between Logic, Geometry, and Algebra which has recently come to light in the form of an interpretation of the constructive type theory of Per Martin-Löf into homotopy theory and higher-dimensional category theory.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rcl} X & \in & \text{Types} \\ x, y & \in & X \end{array} \quad \vdash \quad \text{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \text{Types}}$$

akin to continuous paths in topological spaces.

OXFORD LOGIC GUIDES • 36

Twenty-Five Years of Constructive Type Theory

GIOVANNI SAMBIN
and
JAN SMITH

CHAPTER

6 The groupoid interpretation of type theory

Get access ›

Martin Hofmann, Thomas Streicher

https://doi.org/10.1093/oso/9780198501275.003.0008 Pages 83–112

Published: October 1998

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r l} X & \in \text{Types} \\ x, y & \in X \end{array} \quad \vdash \quad \text{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \text{Types}}$$

akin to continuous paths in topological spaces.

CHAPTER

6 The groupoid interpretation of type theory

Get access ›

Martin Hofmann, Thomas Streicher

https://doi.org/10.1093/oso/9780198501275.003.0008 Pages 83–112

Published: October 1998

Homotopy theoretic models of identity types

Published online by Cambridge University Press: **01 January 2009**

STEVE AWODEY and MICHAEL A. WARREN

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rl} X & \in \mathrm{Types} \\ x, y & \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \in \mathrm{Types} \\ x, y \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

E.g.: if G is a finitely presented group, then we get a type $\mathbf{B}G$ with essentially unique $* \in \mathbf{B}G$ s.t. $\mathrm{Paths}_{\mathbf{B}G}(*,*) \simeq G$.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \in \mathrm{Types} \\ x, y \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

E.g.: if G is a finitely presented group, then we get a type $\mathbf{B}G$ with essentially unique $* \in \mathbf{B}G$ s.t. $\mathrm{Paths}_{\mathbf{B}G}(*,*) \simeq G$.

$$\mathbf{B}G = \left\{ \begin{tikzcd} & * \arrow[dr, squiggly, "g_2"] & \\ * \arrow[ur, squiggly, "g_1"] \arrow[rr, squiggly, "g_1 \cdot g_2"] & & * \end{tikzcd} \;\middle|\; g_i \in G \right\}$$

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rcl} X & \in & \mathrm{Types} \\ x,y & \in & X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

E.g.: if G is a finitely presented group, then we get a type $\mathbf{B}G$ with essentially unique $* \in \mathbf{B}G$ s.t. $\mathrm{Paths}_{\mathbf{B}G}(*,*) \simeq G$.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \in \mathrm{Types} \\ x, y \in X \end{array} \;\vdash\; \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

E.g.: if G is a finitely presented group, then we get a type $\mathbf{B}G$ with essentially unique $* \in \mathbf{B}G$ s.t. $\mathrm{Paths}_{\mathbf{B}G}(*,*) \simeq G$.

For $G = \mathrm{Br}(n)$ an Artin braid group this is the homotopy type of configurations of points: $\mathbf{B}\mathrm{Br}(n) \simeq \int \mathrm{Conf}_n(\mathbb{C})$.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rcl} X & \in & \mathrm{Types} \\ x, y & \in & X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

E.g.: if G is a finitely presented group, then we get a type $\mathbf{B}G$ with essentially unique $* \in \mathbf{B}G$ s.t. $\mathrm{Paths}_{\mathbf{B}G}(*,*) \simeq G$. For $G = \mathrm{Br}(n)$ an Artin braid group this is the homotopy type of configurations of points: $\mathbf{B}\mathrm{Br}(n) \simeq \int \mathrm{Conf}_n(\mathbb{C})$.

$\mathbf{B}\mathrm{Br}(3) = \Big\{$

$\Big\}$

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \,\in\, \mathrm{Types} \\ x, y \,\in\, X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) \,=\, \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \,\in\, \mathrm{Types}}$$

akin to continuous paths in topological spaces.

E.g.: if G is a finitely presented group, then we get a type $\mathbf{B}G$ with essentially unique $* \in \mathbf{B}G$ s.t. $\mathrm{Paths}_{\mathbf{B}G}(*,*) \;\simeq\; G$. For $G = \mathrm{Br}(n)$ an Artin braid group this is the homotopy type of configurations of points: $\mathbf{B}\mathrm{Br}(n) \;\simeq\; \int \mathrm{Conf}_n(\mathbb{C})\,.$

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \in \mathrm{Types} \\ x,y \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

E.g.: if G is a finitely presented group, then we get a type $\mathbf{B}G$ with essentially unique $* \in \mathbf{B}G$ s.t. $\mathrm{Paths}_{\mathbf{B}G}(*,*) \simeq G$. For $G = \mathrm{Br}(n)$ an Artin braid group this is the homotopy type of configurations of points: $\mathbf{B}\mathrm{Br}(n) \simeq \int \mathrm{Conf}_n(\mathbb{C})$.

An X-dependent type family $\boxed{x \in X \;\vdash\; P(x) \in \mathrm{Types}}$ inherits *transport* (monodromy!) along base paths:

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r}X \in \mathrm{Types} \\ x, y \in X\end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

E.g.: if G is a finitely presented group, then we get a type $\mathbf{B}G$ with essentially unique $* \in \mathbf{B}G$ s.t. $\mathrm{Paths}_{\mathbf{B}G}(*,*) \simeq G$.
For $G = \mathrm{Br}(n)$ an Artin braid group this is the homotopy type of configurations of points: $\mathbf{B}\mathrm{Br}(n) \simeq \int \mathrm{Conf}_n(\mathbb{C})$.

An X-dependent type family $\boxed{x \in X \;\vdash\; P(x) \in \mathrm{Types}}$
inherits *transport* (monodromy!) along base paths:

$P : X \to \mathrm{Types}$

$$\begin{tikzcd}
 & P(y) \arrow[dr, "\mathrm{tr}(\gamma_2)"] & \\
P(x) \arrow[ur, "\mathrm{tr}(\gamma_1)"] \arrow[rr, "\mathrm{tr}(\gamma_3)"] & & P(z)
\end{tikzcd}$$

$X : \mathrm{Types}$

$$\begin{tikzcd}
 & y \arrow[dr, squiggly, "\gamma_2"] & \\
x \arrow[ur, squiggly, "\gamma_1"] \arrow[rr, squiggly, "\gamma_3"] & & z
\end{tikzcd}$$

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \,\in\, \mathrm{Types} \\ x, y \,\in\, X \end{array} \;\;\vdash\;\; \mathrm{Paths}_X(x,y) \,=\, \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \,\in\, \mathrm{Types}}$$

akin to continuous paths in topological spaces.

E.g.: if G is a finitely presented group, then we get a type $\mathbf{B}G$ with essentially unique $* \in \mathbf{B}G$ s.t. $\mathrm{Paths}_{\mathbf{B}G}(*,*) \;\simeq\; G$. For $G = \mathrm{Br}(n)$ an Artin braid group this is the homotopy type of configurations of points: $\mathbf{B}\mathrm{Br}(n) \;\simeq\; \int \mathrm{Conf}_n(\mathbb{C})$.

An X-dependent type family $\boxed{x \in X \;\;\vdash\;\; P(x) \,\in\, \mathrm{Types}}$

and compatible *path lifting*:

$P : \; X \longrightarrow \mathrm{Types}$

$$\begin{tikzcd}
& p_y \arrow[dr, squiggly, "\widehat{\gamma_2}"] & \\
p_x \arrow[ur, squiggly, "\widehat{\gamma_1}"] \arrow[rr, squiggly, "\widehat{\gamma_3}"] & & p_z
\end{tikzcd}$$

$$\begin{tikzcd}
& y \arrow[dr, squiggly, "\gamma_2"] & \\
x \arrow[ur, squiggly, "\gamma_1"] \arrow[rr, squiggly, "\gamma_3"] & & z
\end{tikzcd}$$

$X : \; \mathrm{Types}$

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rl} X & \in \text{Types} \\ x, y & \in X \end{array} \quad \vdash \quad \text{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \text{Types}}$$

akin to continuous paths in topological spaces.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rl} X & \in \mathrm{Types} \\ x, y & \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

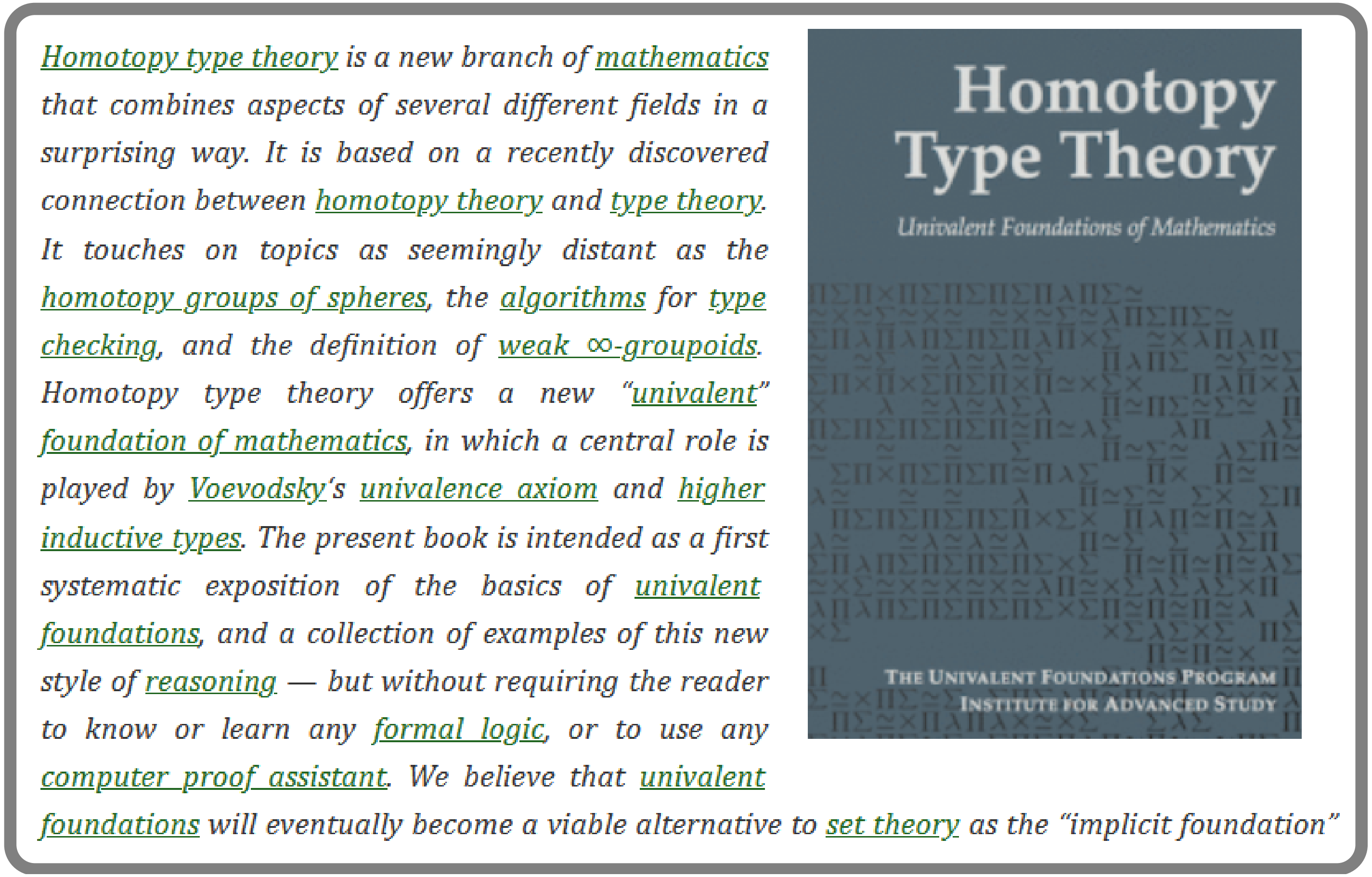
Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \in \text{Types} \\ x,y \in X \end{array} \;\vdash\; \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \text{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.



Homotopy type theory is a new branch of mathematics that combines aspects of several different fields in a surprising way. It is based on a recently discovered connection between homotopy theory and type theory. It touches on topics as seemingly distant as the homotopy groups of spheres, the algorithms for type checking, and the definition of weak ∞-groupoids. Homotopy type theory offers a new "univalent" foundation of mathematics, in which a central role is played by Voevodsky's univalence axiom and higher inductive types. The present book is intended as a first systematic exposition of the basics of univalent foundations, and a collection of examples of this new style of reasoning — but without requiring the reader to know or learn any formal logic, or to use any computer proof assistant. We believe that univalent foundations will eventually become a viable alternative to set theory as the "implicit foundation"

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rl} X \in & \text{Types} \\ x, y \in & X \end{array} \quad \vdash \quad \text{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \text{Types}}$$

akin to continuous paths in topological spaces.

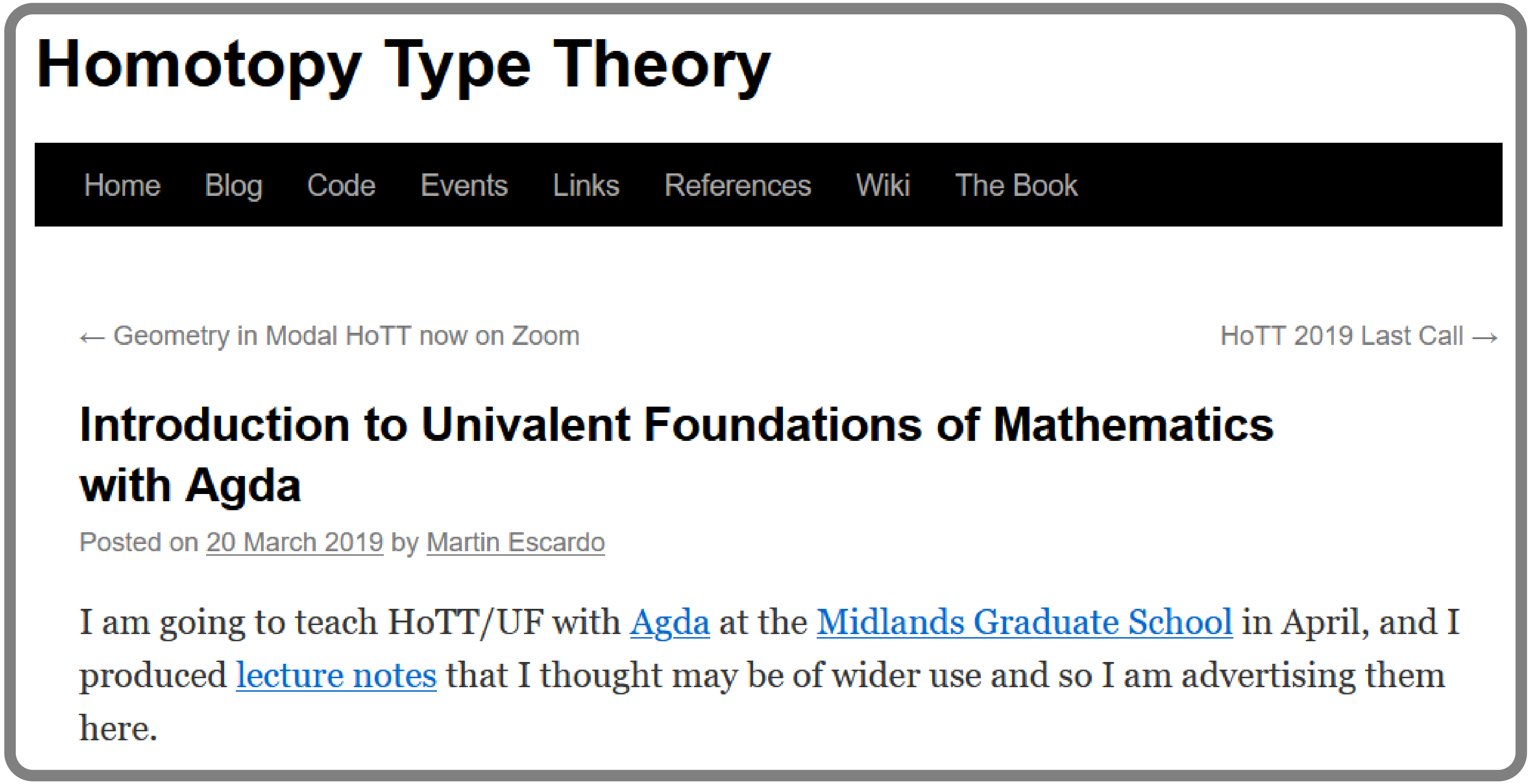
Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rl} X & \in \mathrm{Types} \\ x,y & \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.



← Geometry in Modal HoTT now on Zoom

HoTT 2019 Last Call →

Introduction to Univalent Foundations of Mathematics with Agda

Posted on 20 March 2019 by Martin Escardo

I am going to teach HoTT/UF with Agda at the Midlands Graduate School in April, and I produced lecture notes that I thought may be of wider use and so I am advertising them here.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r@{\;}l} X & \in \mathrm{Types} \\ x, y & \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rl} X & \in \text{Types} \\ x,y & \in X \end{array} \quad \vdash \quad \text{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \text{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rl} X & \in \text{Types} \\ x, y & \in X \end{array} \;\;\vdash\;\; \text{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \text{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.

In HoTT, data types come with paths between their terms

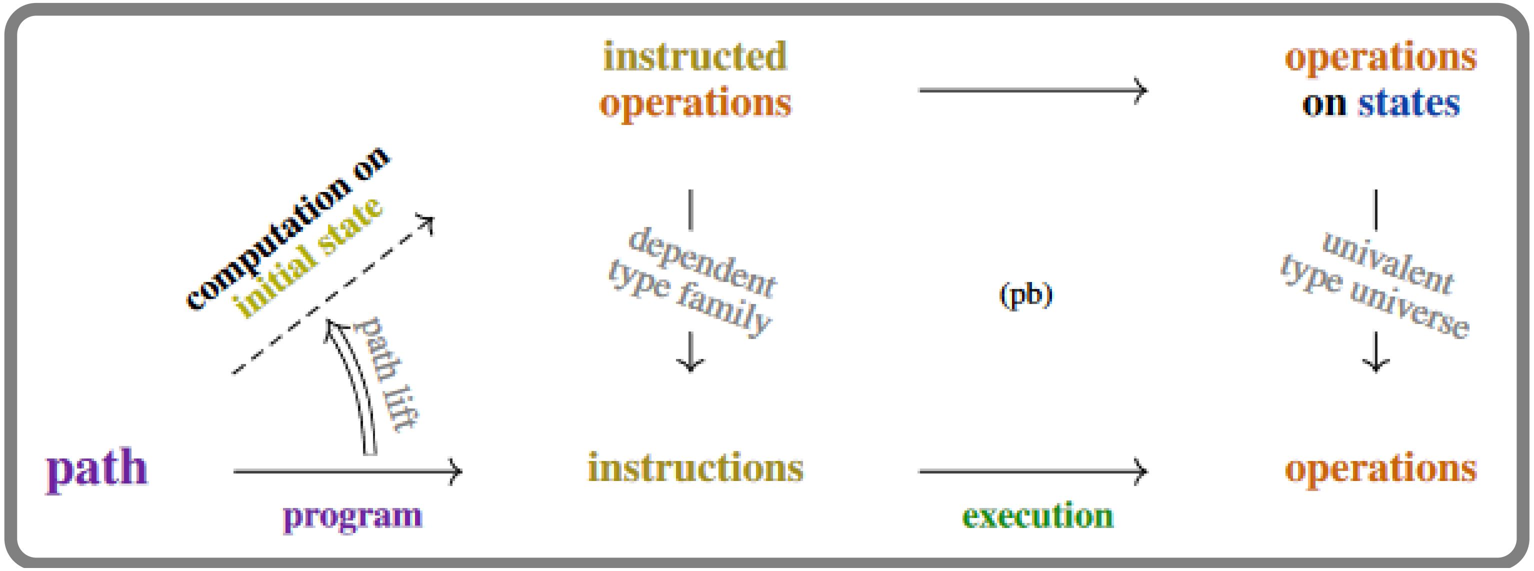
$$\boxed{\begin{array}{rcl} X & \in & \mathrm{Types} \\ x,y & \in & X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.



$$\begin{tikzcd}
 & \text{instructed operations} \arrow[r] \arrow[d, "\text{dependent type family}"] \arrow[dr, phantom, "\text{(pb)}"] & \text{operations on states} \arrow[d, "\text{univalent type universe}"] \\
\text{path} \arrow[r, "\text{program}"', ""{name=P}] \arrow[ur, dashed, "\text{computation on initial state}", ""{name=C, swap}] & \text{instructions} \arrow[r, "\text{execution}"'] & \text{operations}
\arrow[Rightarrow, from=P, to=C, "\text{path lift}"']
\end{tikzcd}$$

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \in \mathrm{Types} \\ x, y \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rcl} X & \in & \mathrm{Types} \\ x, y & \in & X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.

(2.) The dependent homotopy type family of $\mathfrak{su}(2)$-conformal blocks has a slick construction in HoTT programming languages;

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \in \mathrm{Types} \\ x, y \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

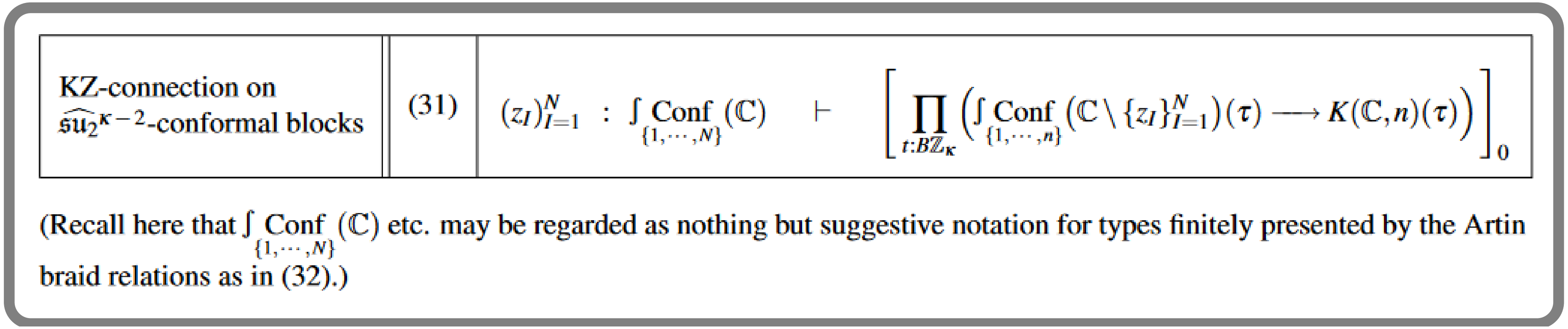
akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.

(2.) The dependent homotopy type family of $\mathfrak{su}(2)$-conformal blocks has a slick construction in HoTT programming languages;



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad \vdash \quad \left[\prod_{t:B\mathbb{Z}_\kappa} \left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}(\mathbb{C} \setminus \{z_I\}_{I=1}^N)(\tau) \longrightarrow K(\mathbb{C},n)(\tau) \right) \right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rl} X & \in \mathrm{Types} \\ x, y & \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.

(2.) The dependent homotopy type family of $\mathfrak{su}(2)$-conformal blocks has a slick construction in HoTT programming languages;

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rcl} X & \in & \mathrm{Types} \\ x,y & \in & X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.

(2.) The dependent homotopy type family of $\mathfrak{su}(2)$-conformal blocks has a slick construction in HoTT programming languages;

(3.) its path lifting operation is exactly anyonic braid gate execution.

In HoTT, data types come with paths between their terms

$X \in$ Types
$x, y \in X$

$y\big\} \in$ Types

akin t

paces.

Such HoTT pr

be remarkably

fundamental, arg

for mathematics.

We now first offer

(1.) Reversible c

um computation –

is described

topy type families.

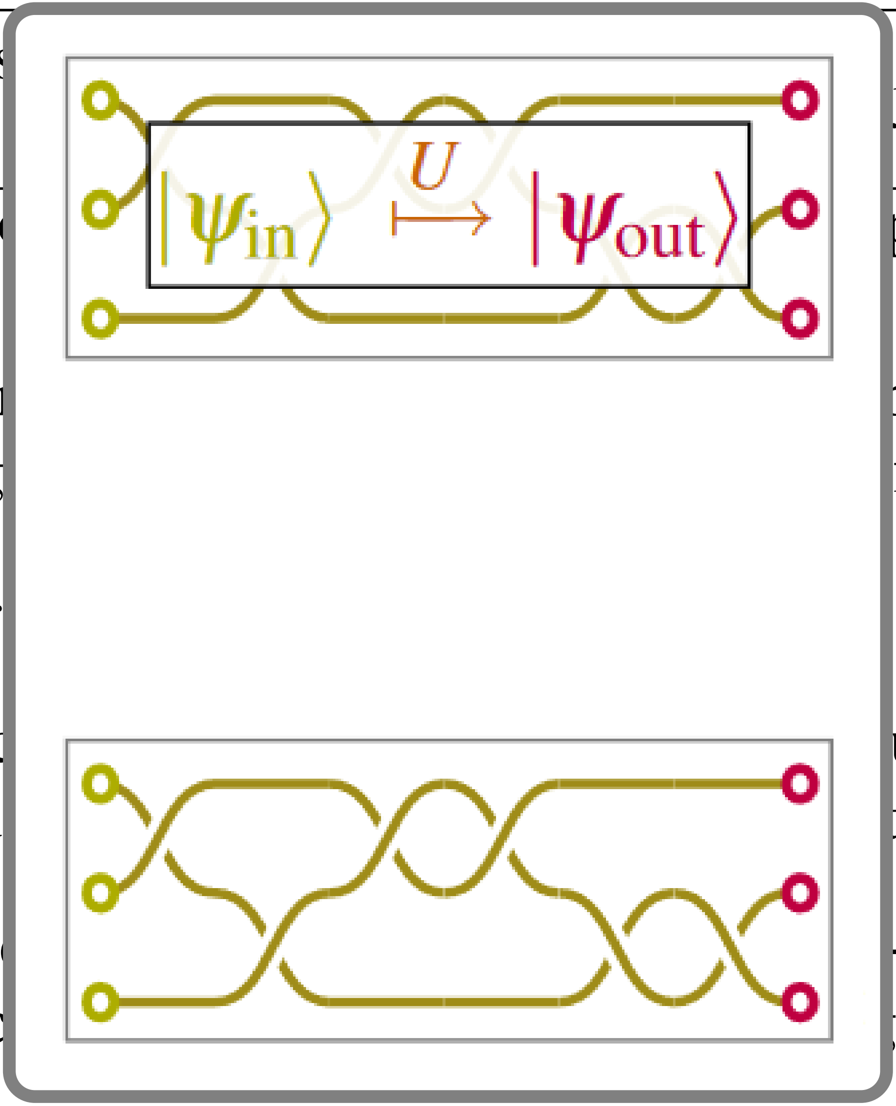
(2.) The depend

-conformal blocks

has a slick c

languages;

(3.) its path lifting operation is exactly anyonic braid gate execution.



In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rl} X & \in \mathrm{Types} \\ x, y & \in X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.

(2.) The dependent homotopy type family of $\mathfrak{su}(2)$-conformal blocks has a slick construction in HoTT programming languages;

(3.) its path lifting operation is exactly anyonic braid gate execution.

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{r} X \in \text{Types} \\ x, y \in X \end{array} \quad \vdash \quad \text{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \text{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.

(2.) The dependent homotopy type family of $\mathfrak{su}(2)$-conformal blocks has a slick construction in HoTT programming languages;

(3.) its path lifting operation is exactly anyonic braid gate execution.

$\Rightarrow$ natural & powerful topological-hardware-aware Q-pogramming paradigm

In HoTT, data types come with paths between their terms

$$\boxed{\begin{array}{rcl} X & \in & \mathrm{Types} \\ x, y & \in & X \end{array} \quad \vdash \quad \mathrm{Paths}_X(x,y) = \left\{ x \overset{\gamma}{\rightsquigarrow} y \right\} \in \mathrm{Types}}$$

akin to continuous paths in topological spaces.

Such HoTT programming languages turn out to be remarkably fundamental, arguably serving as a new foundation for mathematics.

We now first offer the following observations:

(1.) Reversible circuit execution – such as in quantum computation – is described by *path lifting* in dependent homotopy type families.

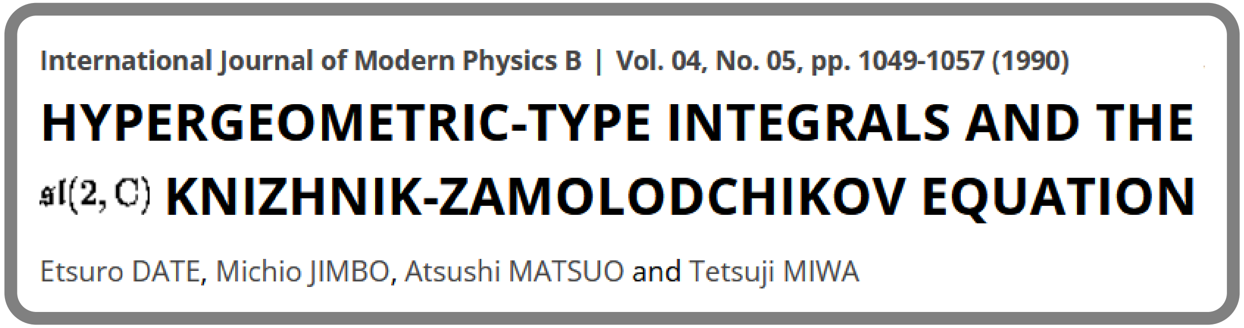
(2.) The dependent homotopy type family of $\mathfrak{su}(2)$-conformal blocks has a slick construction in HoTT programming languages;

(3.) its path lifting operation is exactly anyonic braid gate execution.

⇒ natural & powerful topological-hardware-aware Q-pogramming paradigm

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks secretly happen to have a purely *cohomological* definition.

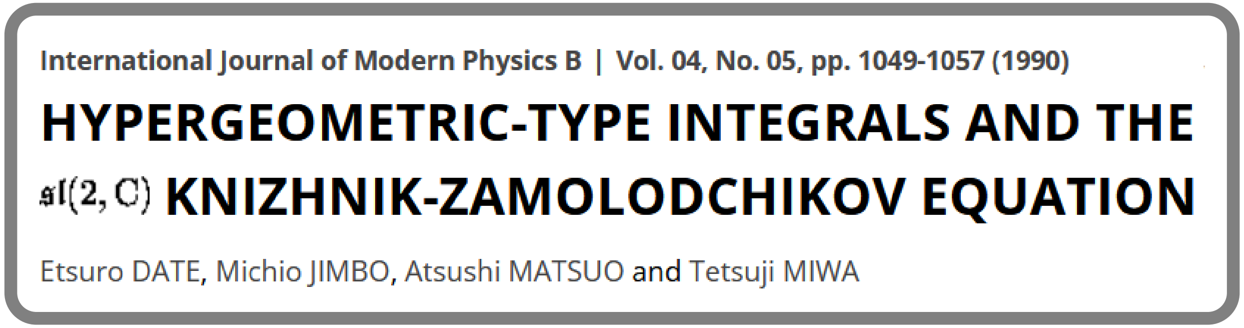


International Journal of Modern Physics B | Vol. 04, No. 05, pp. 1049-1057 (1990)

HYPERGEOMETRIC-TYPE INTEGRALS AND THE $\mathfrak{sl}(2, \mathbb{C})$ KNIZHNIK-ZAMOLODCHIKOV EQUATION

Etsuro DATE, Michio JIMBO, Atsushi MATSUO and Tetsuji MIWA

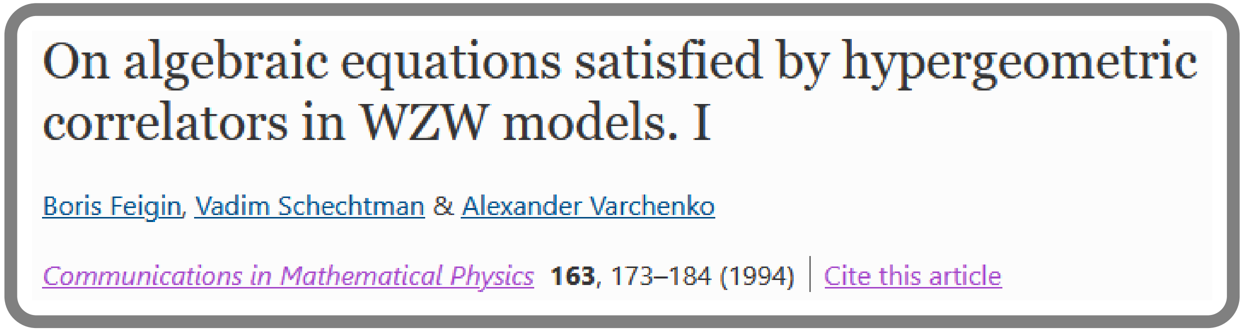
Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.



International Journal of Modern Physics B | Vol. 04, No. 05, pp. 1049-1057 (1990)

HYPERGEOMETRIC-TYPE INTEGRALS AND THE $\mathfrak{sl}(2,\mathbb{C})$ KNIZHNIK-ZAMOLODCHIKOV EQUATION

Etsuro DATE, Michio JIMBO, Atsushi MATSUO and Tetsuji MIWA

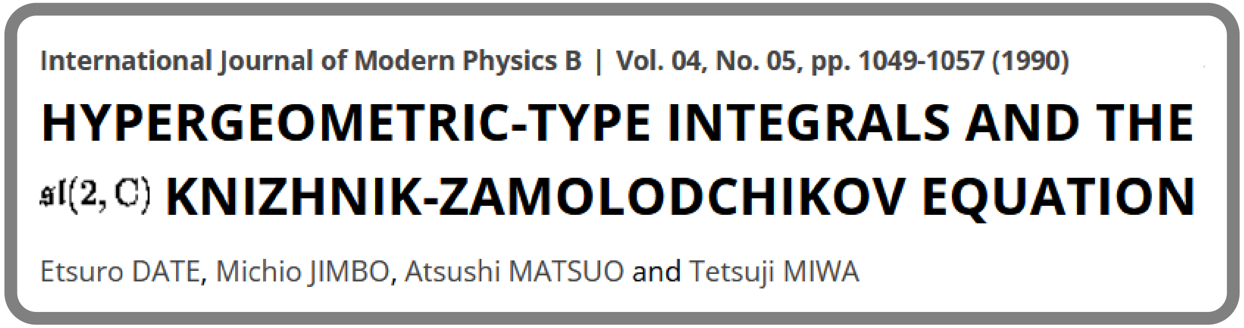


On algebraic equations satisfied by hypergeometric correlators in WZW models. I

Boris Feigin, Vadim Schechtman & Alexander Varchenko

Communications in Mathematical Physics **163**, 173–184 (1994) | Cite this article

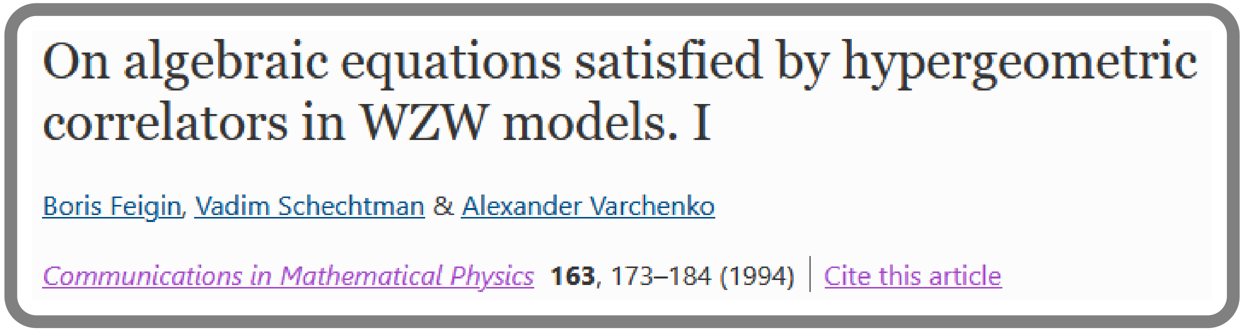
Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.



International Journal of Modern Physics B | Vol. 04, No. 05, pp. 1049-1057 (1990)

HYPERGEOMETRIC-TYPE INTEGRALS AND THE $\mathfrak{sl}(2,\mathbb{C})$ KNIZHNIK-ZAMOLODCHIKOV EQUATION

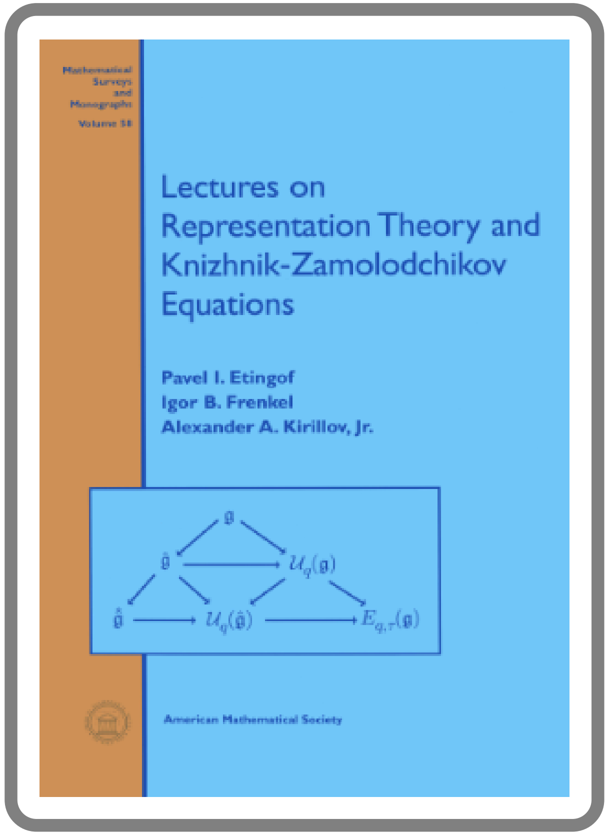
Etsuro DATE, Michio JIMBO, Atsushi MATSUO and Tetsuji MIWA



On algebraic equations satisfied by hypergeometric correlators in WZW models. I

Boris Feigin, Vadim Schechtman & Alexander Varchenko

Communications in Mathematical Physics **163**, 173–184 (1994) | Cite this article



Mathematical
Surveys
and
Monographs
Volume 58

Lectures on
Representation Theory and
Knizhnik-Zamolodchikov
Equations

Pavel I. Etingof
Igor B. Frenkel
Alexander A. Kirillov, Jr.

$$\begin{tikzcd}
 & & \mathfrak{g} \arrow[dl] \arrow[drr] & & \\
 & \hat{\mathfrak{g}} \arrow[rrr] \arrow[dl] \arrow[dr] & & & \mathcal{U}_q(\mathfrak{g}) \arrow[dll] \arrow[dr] \\
\hat{\hat{\mathfrak{g}}} \arrow[rr] & & \mathcal{U}_q(\hat{\mathfrak{g}}) \arrow[rrr] & & & E_{q,\tau}(\mathfrak{g})
\end{tikzcd}$$

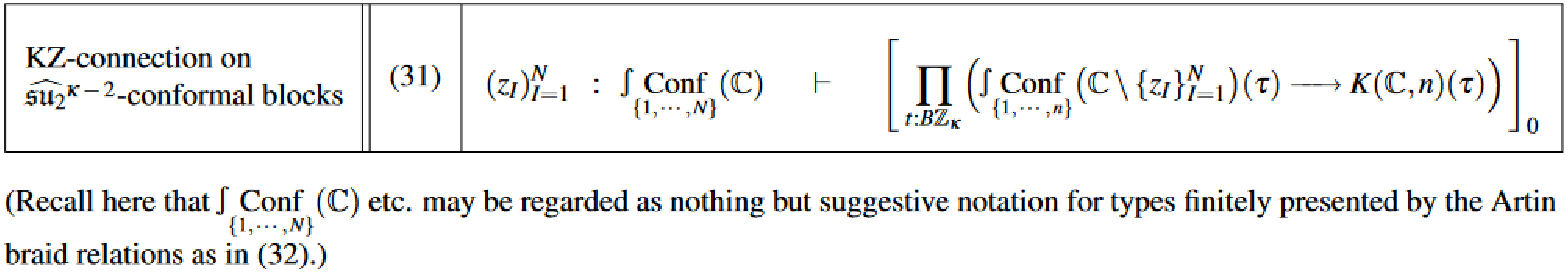
American Mathematical Society

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.

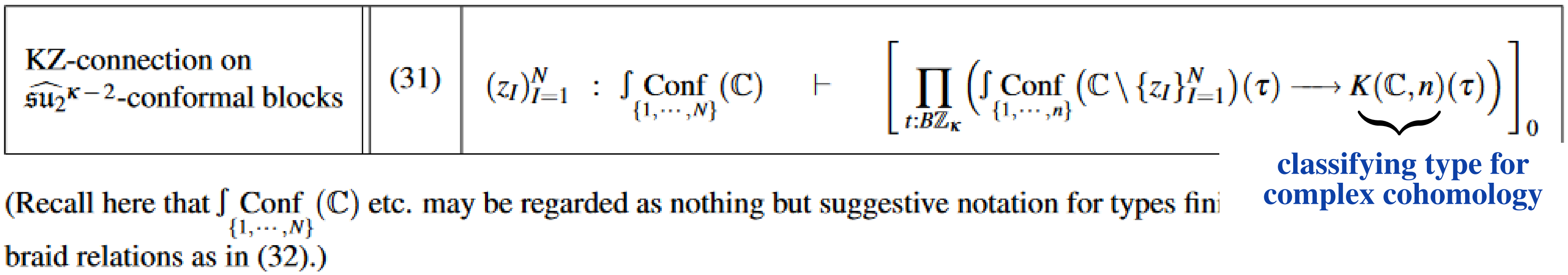
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad\vdash\quad \left[\prod_{t:B\mathbb{Z}_\kappa}\left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}\big(\mathbb{C}\setminus\{z_I\}_{I=1}^N\big)(\tau) \longrightarrow K(\mathbb{C},n)(\tau)\right)\right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

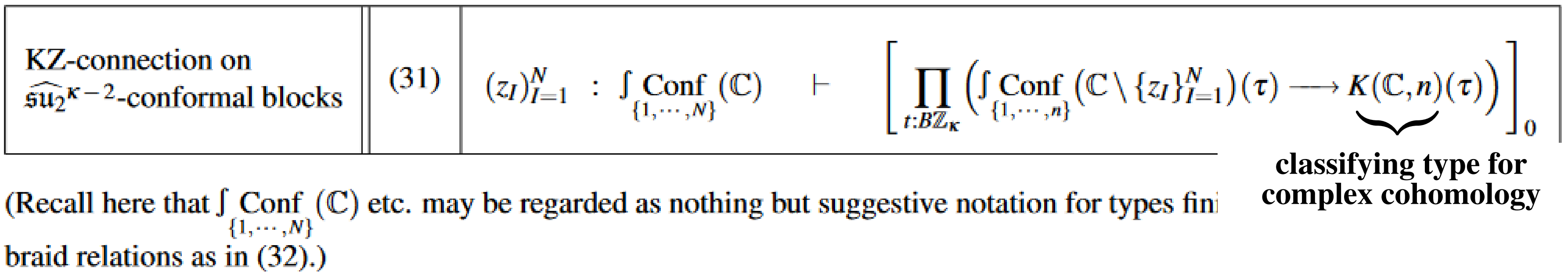
Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad \vdash \quad \left[\prod_{t:B\mathbb{Z}_\kappa} \left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}\big(\mathbb{C}\setminus\{z_I\}_{I=1}^N\big)(\tau) \longrightarrow \underbrace{K(\mathbb{C},n)(\tau)}_{\text{classifying type for complex cohomology}}\right)\right]_0$

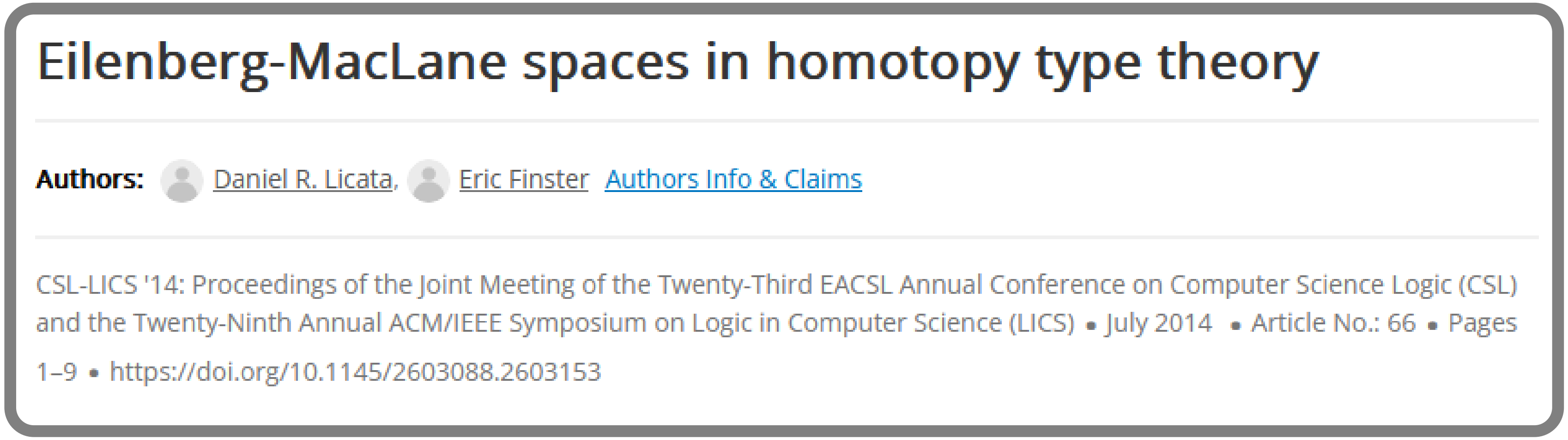
(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types fini
braid relations as in (32).)

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \;\vdash\; \left[\prod_{t:B\mathbb{Z}_\kappa} \left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}\big(\mathbb{C}\setminus\{z_I\}_{I=1}^N\big)(\tau) \longrightarrow \underbrace{K(\mathbb{C},n)(\tau)}_{\text{classifying type for complex cohomology}}\right)\right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types fini
braid relations as in (32).)

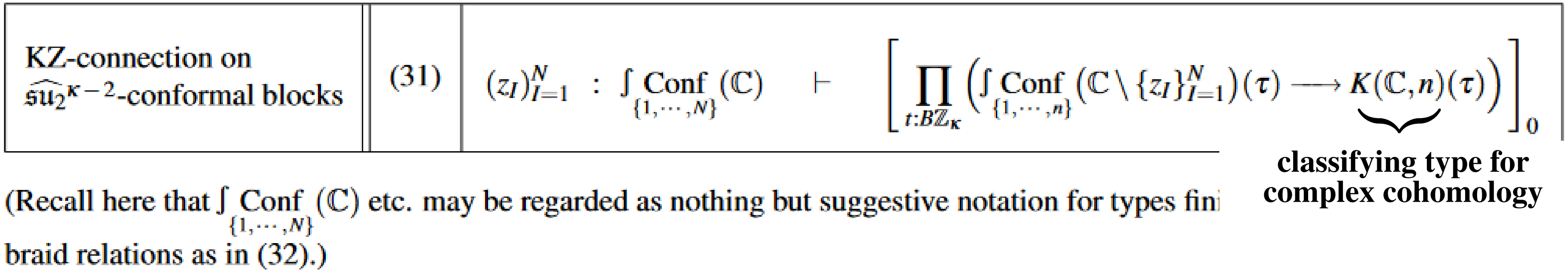


Eilenberg-MacLane spaces in homotopy type theory

Authors: Daniel R. Licata, Eric Finster Authors Info & Claims

CSL-LICS '14: Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS) • July 2014 • Article No.: 66 • Pages 1–9 • https://doi.org/10.1145/2603088.2603153

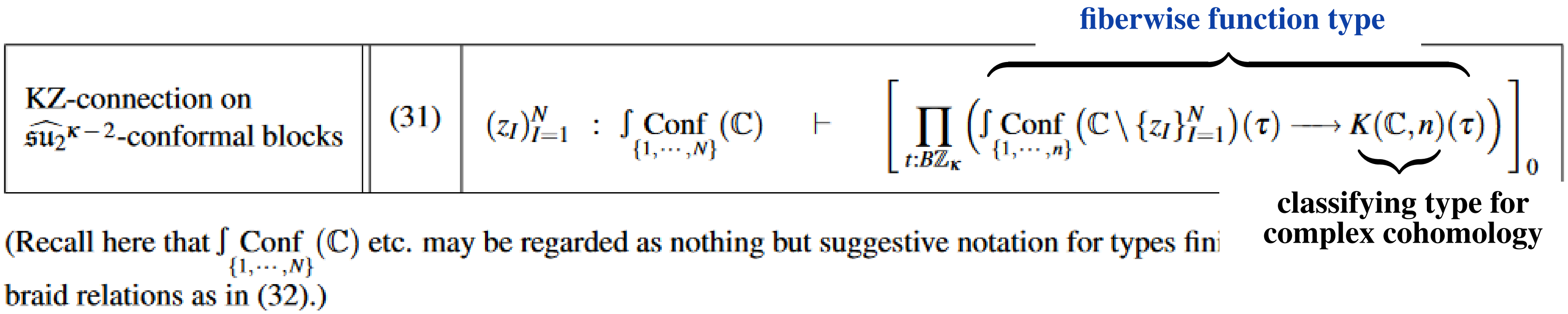
Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad \vdash \quad \left[\prod_{t:B\mathbb{Z}_\kappa} \left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}} (\mathbb{C} \setminus \{z_I\}_{I=1}^N)(\tau) \longrightarrow \underbrace{K(\mathbb{C},n)}_{\text{classifying type for complex cohomology}}(\tau) \right) \right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types fini
braid relations as in (32).)

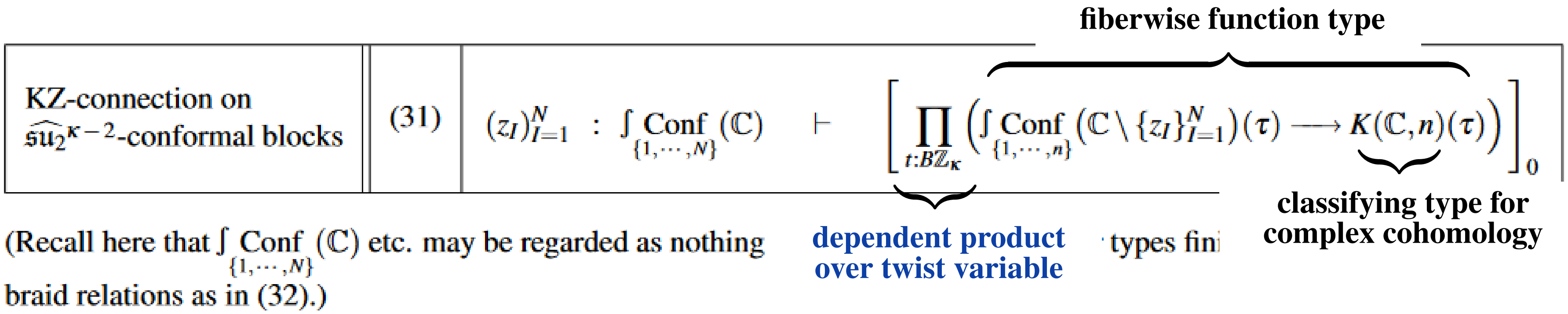
Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad \vdash \quad \left[\prod_{t:B\mathbb{Z}_\kappa} \overbrace{\left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}(\mathbb{C}\setminus\{z_I\}_{I=1}^N)(\tau) \longrightarrow K\underbrace{(\mathbb{C},n)}_{\text{classifying type for complex cohomology}}(\tau)\right)}^{\text{fiberwise function type}}\right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types fini
braid relations as in (32).)

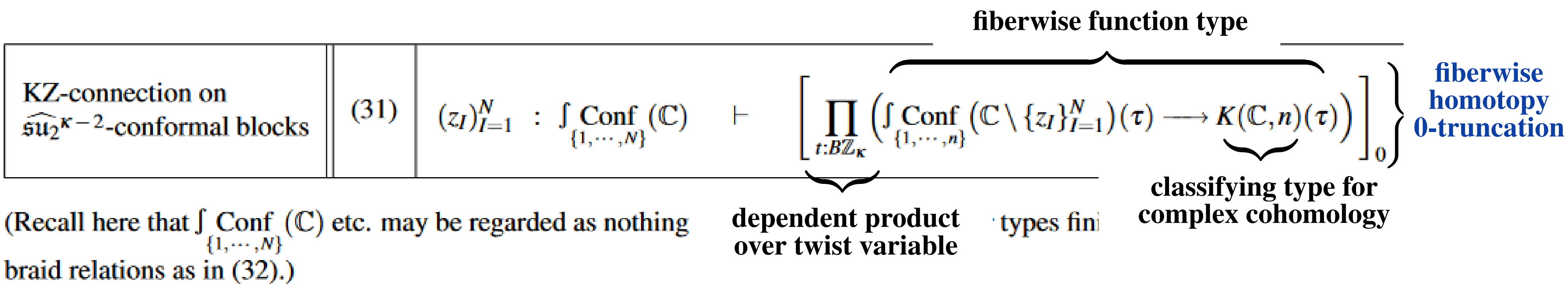
Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \mathrm{Conf}_{\{1,\cdots,N\}}(\mathbb{C}) \;\vdash\; \Big[\underbrace{\prod_{t:B\mathbb{Z}_\kappa}}_{\text{dependent product over twist variable}} \Big(\overbrace{\int \mathrm{Conf}_{\{1,\cdots,n\}}\big(\mathbb{C}\setminus\{z_I\}_{I=1}^N\big)(\tau) \longrightarrow K\underbrace{(\mathbb{C},n)}_{\text{classifying type for complex cohomology}}(\tau)}^{\text{fiberwise function type}} \Big) \Big]_0$

(Recall here that $\int \mathrm{Conf}_{\{1,\cdots,N\}}(\mathbb{C})$ etc. may be regarded as nothing types fini
braid relations as in (32).)

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:

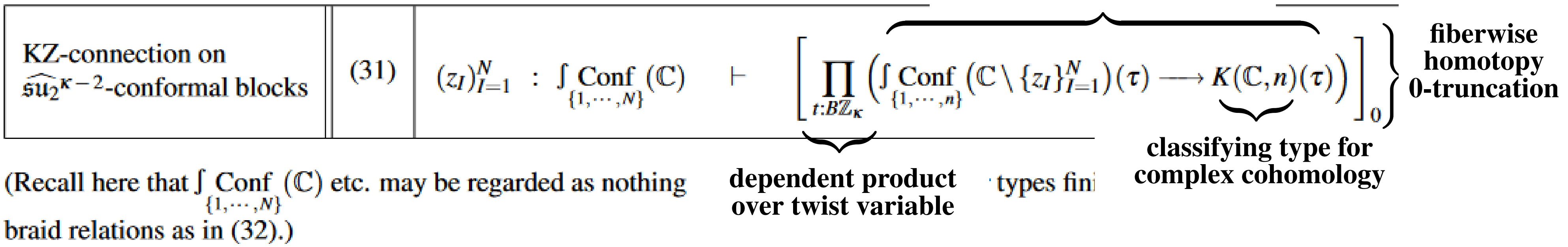


KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \mathrm{Conf}_{\{1,\cdots,N\}}(\mathbb{C}) \;\vdash\; \left[\underbrace{\prod_{t:B\mathbb{Z}_\kappa}}_{\text{dependent product over twist variable}} \overbrace{\left(\int \mathrm{Conf}_{\{1,\cdots,n\}}\big(\mathbb{C}\setminus\{z_I\}_{I=1}^N\big)(\tau) \longrightarrow \underbrace{K(\mathbb{C},n)}_{\text{classifying type for complex cohomology}}(\tau)\right)}^{\text{fiberwise function type}}\right]_0$

fiberwise homotopy 0-truncation

(Recall here that $\int \mathrm{Conf}_{\{1,\cdots,N\}}(\mathbb{C})$ etc. may be regarded as nothing types fini
braid relations as in (32).)

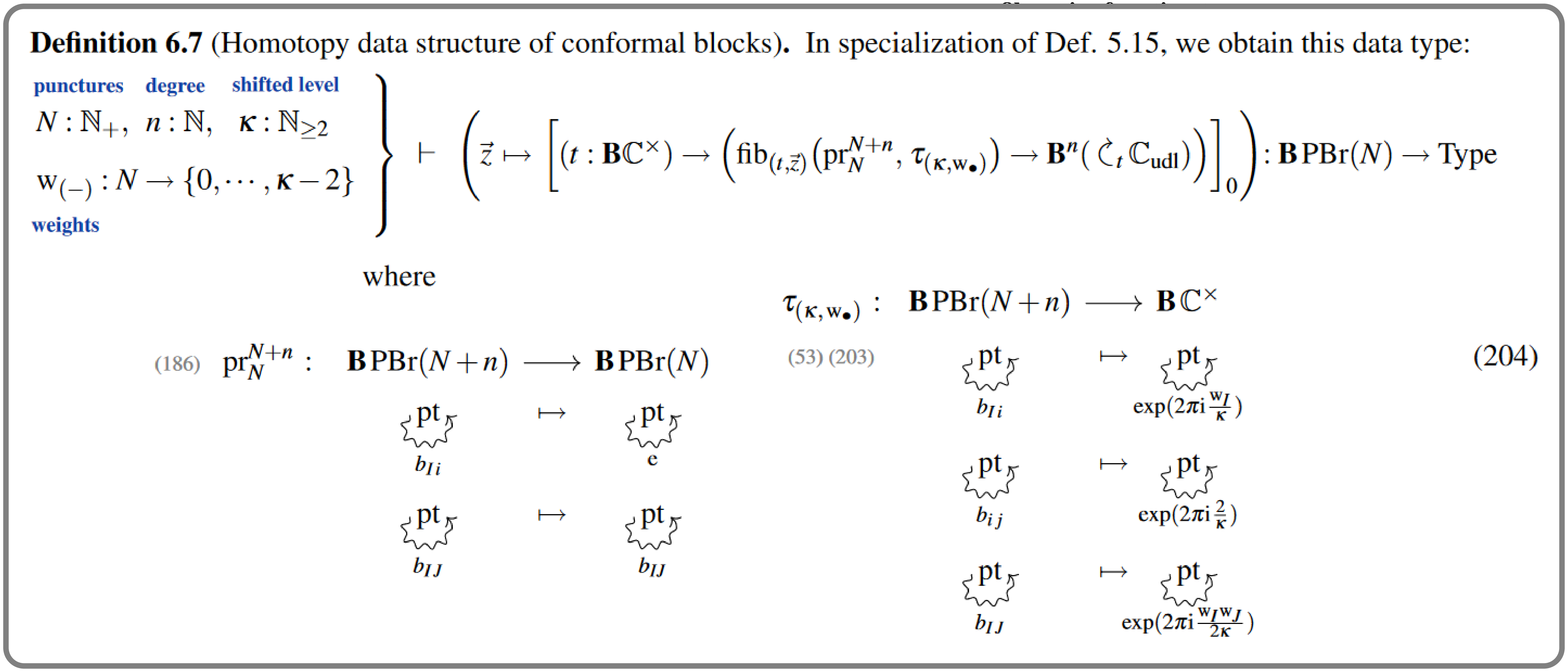
Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \;\vdash\; \left.\left[\underbrace{\prod_{t:B\mathbb{Z}_\kappa}}_{\text{dependent product over twist variable}} \overbrace{\left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}\left(\mathbb{C}\setminus\{z_I\}_{I=1}^N\right)(\tau) \longrightarrow \underbrace{K(\mathbb{C},n)}_{\text{classifying type for complex cohomology}}(\tau)\right)}^{\text{fiberwise function type}}\right]_0\right\}\text{fiberwise homotopy 0-truncation}$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing types fini
braid relations as in (32).)

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



Definition 6.7 (Homotopy data structure of conformal blocks)**.** In specialization of Def. 5.15, we obtain this data type:

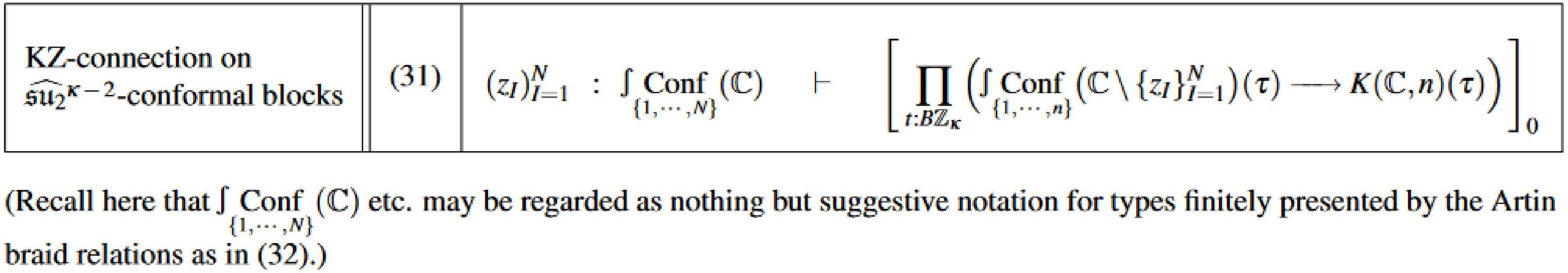
$$\left.\begin{array}{l}\overset{\text{punctures}}{N : \mathbb{N}_+}, \;\; \overset{\text{degree}}{n : \mathbb{N}}, \;\; \overset{\text{shifted level}}{\kappa : \mathbb{N}_{\geq 2}} \\ \underset{\text{weights}}{\mathrm{w}_{(-)} : N \to \{0, \cdots, \kappa - 2\}}\end{array}\right\} \vdash \left(\vec{z} \mapsto \Big[(t : \mathbf{B}\mathbb{C}^\times) \to \Big(\mathrm{fib}_{(t,\vec{z})}\big(\mathrm{pr}_N^{N+n}, \tau_{(\kappa,\mathrm{w}_\bullet)}\big) \to \mathbf{B}^n\big(\circlearrowleft_t \mathbb{C}_{\mathrm{udl}}\big)\Big)\Big]_0\right) : \mathbf{B}\,\mathrm{PBr}(N) \to \mathrm{Type}$$

where

$$\begin{array}{lllll} & & & \tau_{(\kappa,\mathrm{w}_\bullet)} : & \mathbf{B}\,\mathrm{PBr}(N+n) \longrightarrow \mathbf{B}\mathbb{C}^\times \\ \text{(186)} \;\; \mathrm{pr}_N^{N+n} : & \mathbf{B}\,\mathrm{PBr}(N+n) \longrightarrow \mathbf{B}\,\mathrm{PBr}(N) & & \text{(53) (203)} & \underset{b_{Ii}}{\mathrm{pt}\circlearrowright} \;\mapsto\; \underset{\exp(2\pi\mathrm{i}\frac{\mathrm{w}_I}{\kappa})}{\mathrm{pt}\circlearrowright} \\ & \underset{b_{Ii}}{\mathrm{pt}\circlearrowright} \;\mapsto\; \underset{\mathrm{e}}{\mathrm{pt}\circlearrowright} & & & \underset{b_{ij}}{\mathrm{pt}\circlearrowright} \;\mapsto\; \underset{\exp(2\pi\mathrm{i}\frac{2}{\kappa})}{\mathrm{pt}\circlearrowright} \\ & \underset{b_{IJ}}{\mathrm{pt}\circlearrowright} \;\mapsto\; \underset{b_{IJ}}{\mathrm{pt}\circlearrowright} & & & \underset{b_{IJ}}{\mathrm{pt}\circlearrowright} \;\mapsto\; \underset{\exp(2\pi\mathrm{i}\frac{\mathrm{w}_I \mathrm{w}_J}{2\kappa})}{\mathrm{pt}\circlearrowright} \end{array} \qquad (204)$$

Myers, Sati & Schreiber: *Topological Quantum Gates in Homotopy Type Theory*

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:

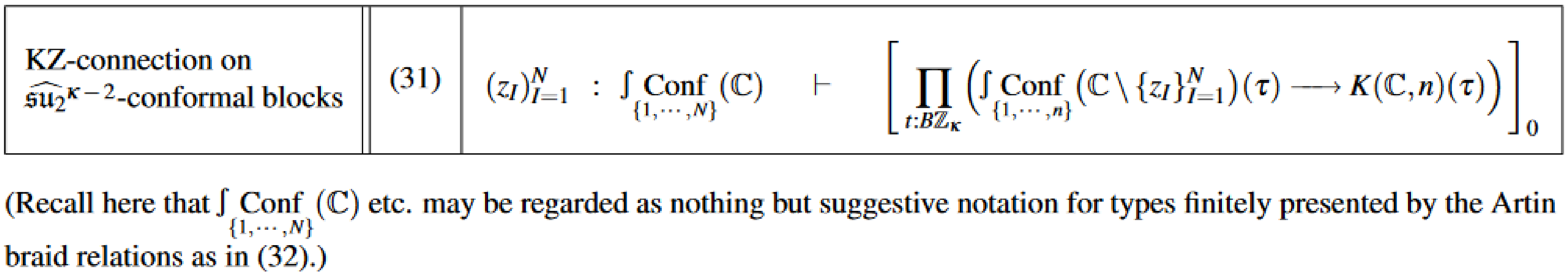


KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad\vdash\quad \Big[\prod_{t:B\mathbb{Z}_\kappa}\Big(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}\big(\mathbb{C}\setminus\{z_I\}_{I=1}^N\big)(\tau) \longrightarrow K(\mathbb{C},n)(\tau)\Big)\Big]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

This $\left\langle \begin{array}{c}\text{works because}\\ \text{and uses that}\end{array} \right\rangle$ HoTT has categorical semantics
in Parameterized Homotopy Theory.

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:

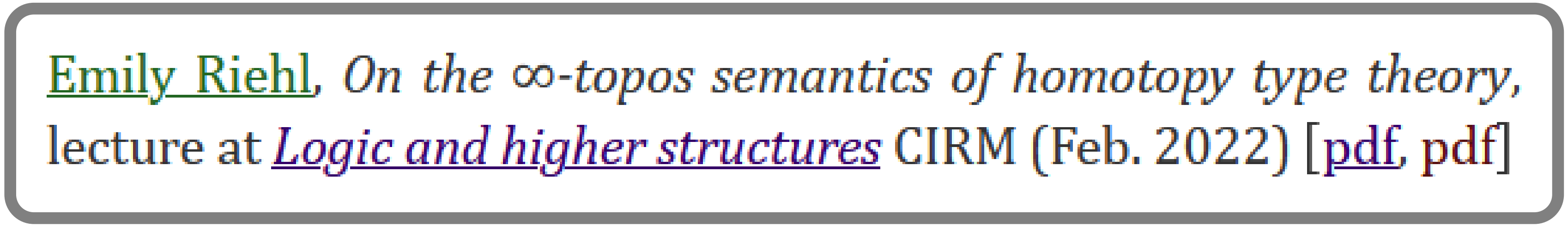


KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad \vdash \quad \left[\prod_{t:B\mathbb{Z}_\kappa} \left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}(\mathbb{C} \setminus \{z_I\}_{I=1}^N)(\tau) \longrightarrow K(\mathbb{C},n)(\tau)\right)\right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

$$\text{This} \left\langle \begin{matrix}\text{works because} \\ \text{and uses that}\end{matrix} \right\rangle \text{HoTT has categorical semantics}$$

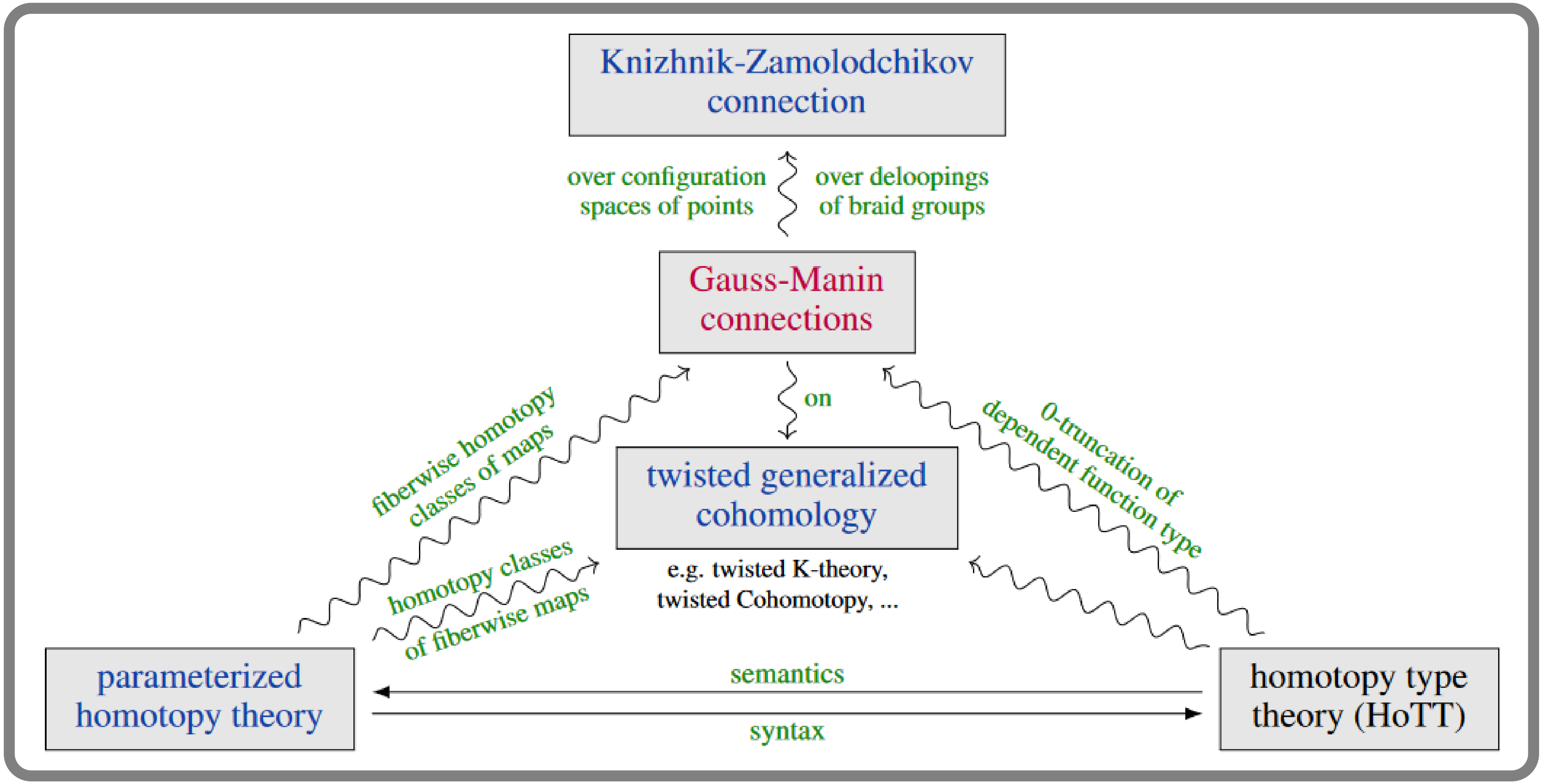
in Parameterized Homotopy Theory.



Emily Riehl, *On the ∞-topos semantics of homotopy type theory*, lecture at *Logic and higher structures* CIRM (Feb. 2022) [pdf, pdf]

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:

KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int_{\{1,\cdots,N\}} \mathrm{Conf}(\mathbb{C}) \quad \vdash \quad \left[\prod_{t:B\mathbb{Z}_\kappa} \left(\int_{\{1,\cdots,n\}} \mathrm{Conf}(\mathbb{C} \setminus \{z_I\}_{I=1}^N)(\tau) \longrightarrow K(\mathbb{C},n)(\tau) \right) \right]_0$



Knizhnik-Zamolodchikov connection

over configuration spaces of points

over deloopings of braid groups

Gauss-Manin connections

on

twisted generalized cohomology

e.g. twisted K-theory,
twisted Cohomotopy, ...

fiberwise homotopy classes of maps

homotopy classes of fiberwise maps

0-truncation of dependent function type

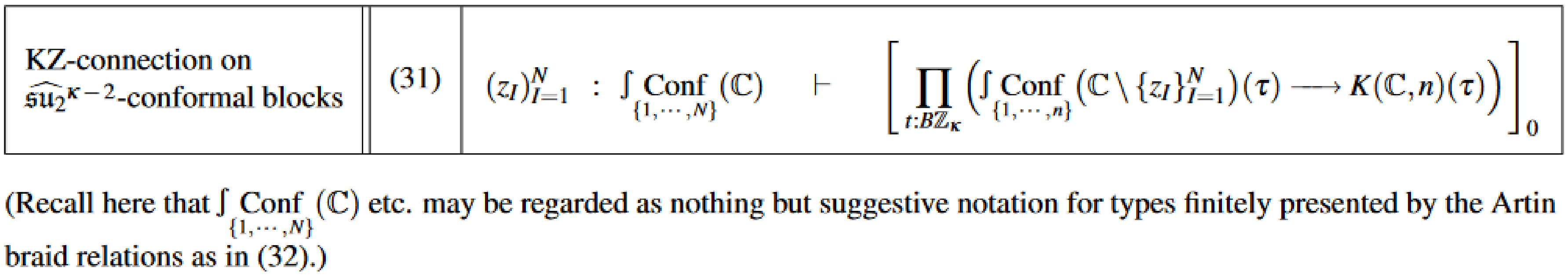
parameterized homotopy theory

semantics

syntax

homotopy type theory (HoTT)

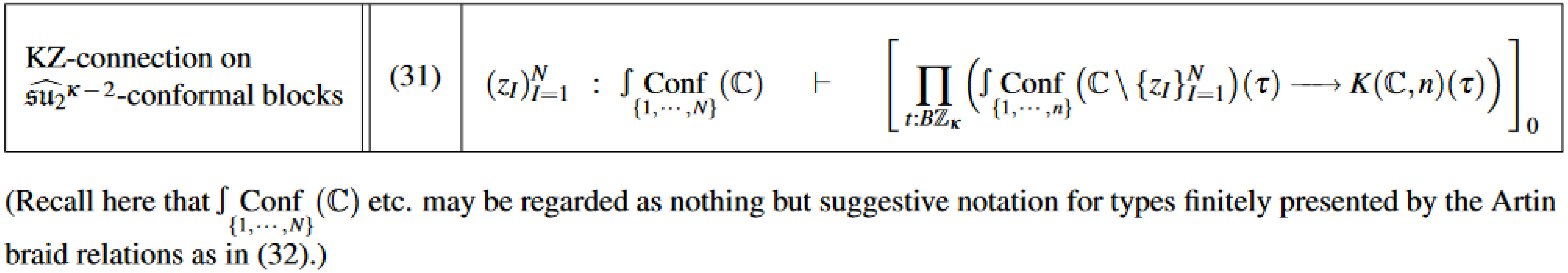
Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad \vdash \quad \left[\prod_{t:B\mathbb{Z}_\kappa}\left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}(\mathbb{C}\setminus\{z_I\}_{I=1}^N)(\tau) \longrightarrow K(\mathbb{C},n)(\tau)\right)\right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad\vdash\quad \left[\prod_{t:B\mathbb{Z}_\kappa}\left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}(\mathbb{C}\setminus\{z_I\}_{I=1}^N)(\tau) \longrightarrow K(\mathbb{C},n)(\tau)\right)\right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

Claim: Its transport operation is the monodromy braid representation

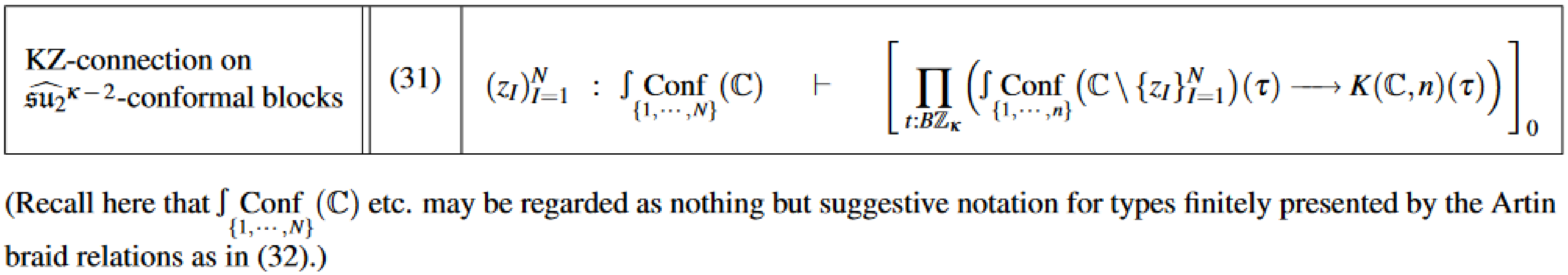
$$P : \; X \longrightarrow \mathrm{Types}$$

$$\begin{tikzcd}
& P(y) \arrow[dr, "\mathrm{tr}(\gamma_2)"] & \\
P(x) \arrow[ur, "\mathrm{tr}(\gamma_1)"] \arrow[rr, "\mathrm{tr}(\gamma_3)"] & & P(z)
\end{tikzcd}$$

$$X : \; \mathrm{Types}$$

$$\begin{tikzcd}
& y \arrow[dr, squiggly, "\gamma_2"] & \\
x \arrow[ur, squiggly, "\gamma_1"] \arrow[rr, squiggly, "\gamma_3"] & & z
\end{tikzcd}$$

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad \vdash \quad \left[\prod_{t:B\mathbb{Z}_\kappa}\left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}(\mathbb{C}\setminus\{z_I\}_{I=1}^N)(\tau) \longrightarrow K(\mathbb{C},n)(\tau)\right)\right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

Claim: Its transport operation is the monodromy braid representation
and its path lifting is execution of $\mathfrak{su}(2)$-anyon braid gates.

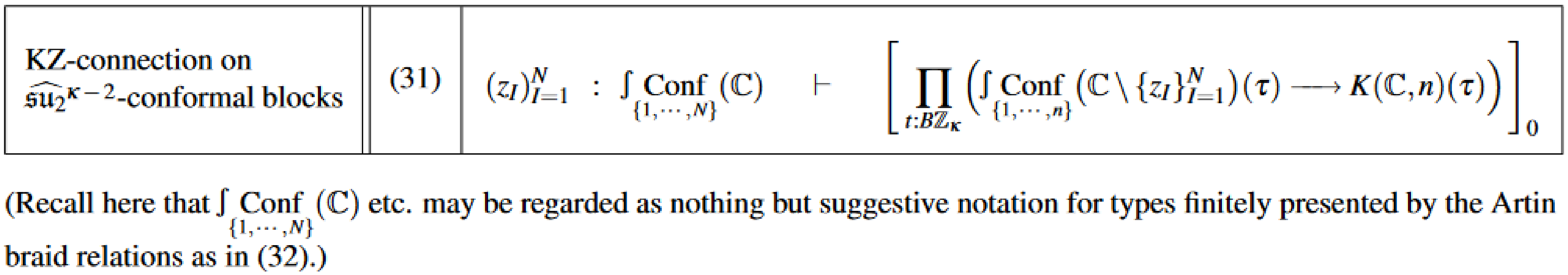
$P : X \longrightarrow \mathrm{Types}$

$$\begin{tikzcd}
 & p_y \arrow[rd, rightsquigarrow, "\widehat{\gamma_2}"] & \\
p_x \arrow[ru, rightsquigarrow, "\widehat{\gamma_1}"] \arrow[rr, rightsquigarrow, "\widehat{\gamma_3}"] & & p_z
\end{tikzcd}$$

$$\begin{tikzcd}
 & y \arrow[rd, rightsquigarrow, "\gamma_2"] & \\
x \arrow[ru, rightsquigarrow, "\gamma_1"] \arrow[rr, rightsquigarrow, "\gamma_3"] & & z
\end{tikzcd}$$

$X : \mathrm{Types}$

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



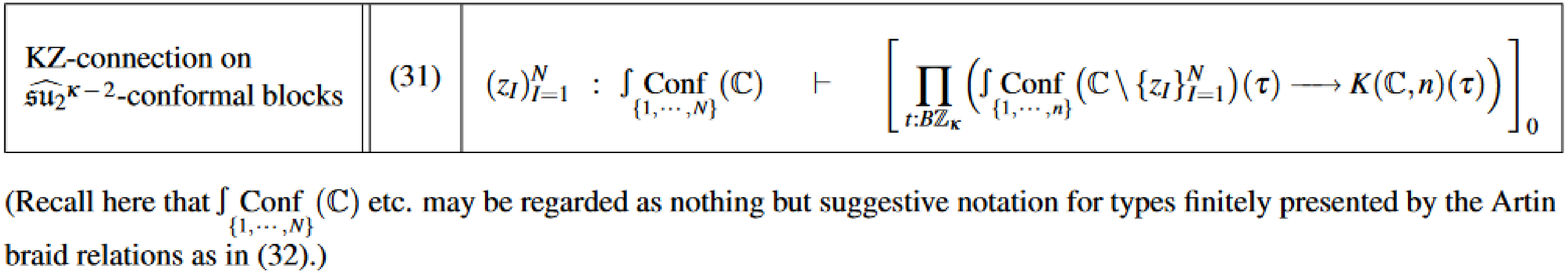
KZ-connection on $\widehat{\mathfrak{su}}_2{}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int\underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad\vdash\quad \left[\prod_{t:B\mathbb{Z}_\kappa}\left(\int\underset{\{1,\cdots,n\}}{\mathrm{Conf}}(\mathbb{C}\setminus\{z_I\}_{I=1}^N)(\tau)\longrightarrow K(\mathbb{C},n)(\tau)\right)\right]_0$

(Recall here that $\int\underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

Claim: Its transport operation is the monodromy braid representation
and its path lifting is execution of $\mathfrak{su}(2)$-anyon braid gates.

Hence coding this type family into a HoTT language like `Agda`
gives a Topological Quantum Programming Language

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



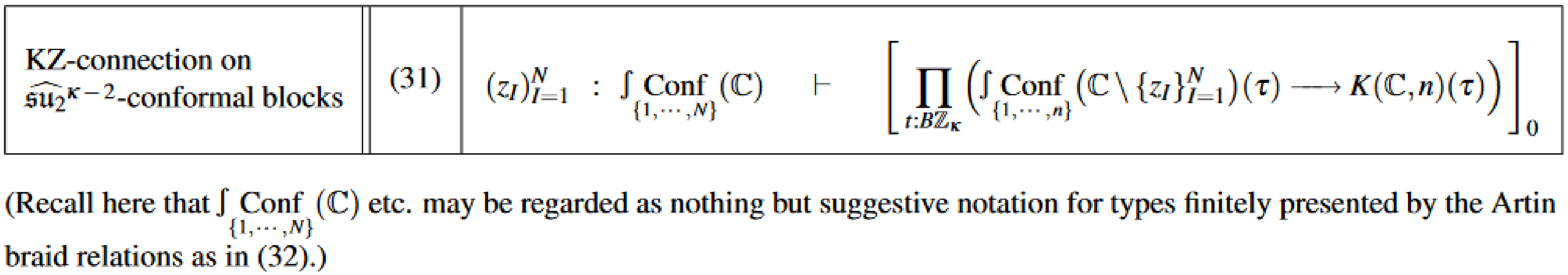
KZ-connection on $\widehat{\mathfrak{su}_2}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int_{\{1,\cdots,N\}} \mathrm{Conf}\,(\mathbb{C}) \quad \vdash \quad \left[\prod_{t : B\mathbb{Z}_\kappa} \left(\int_{\{1,\cdots,n\}} \mathrm{Conf}\,(\mathbb{C} \setminus \{z_I\}_{I=1}^N)(\tau) \longrightarrow K(\mathbb{C},n)(\tau) \right) \right]_0$

(Recall here that $\int_{\{1,\cdots,N\}} \mathrm{Conf}\,(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

Claim: Its transport operation is the monodromy braid representation
and its path lifting is execution of $\mathfrak{su}(2)$-anyon braid gates.

Hence coding this type family into a HoTT language like `Agda`
gives a Topological Quantum Programming Language
which is fully aware of topological anyon braid quantum gates.

Namely, bundles of $\mathfrak{su}(2)$-conformal blocks
secretly happen to have a purely *cohomological* definition.
We show how to construct this as a dependent type family in HoTT:



KZ-connection on $\widehat{\mathfrak{su}}_2{}^{\kappa-2}$-conformal blocks	(31)	$(z_I)_{I=1}^N \;:\; \int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C}) \quad \vdash \quad \left[\prod_{t:B\mathbb{Z}_\kappa}\left(\int \underset{\{1,\cdots,n\}}{\mathrm{Conf}}(\mathbb{C}\setminus\{z_I\}_{I=1}^N)(\tau) \longrightarrow K(\mathbb{C},n)(\tau)\right)\right]_0$

(Recall here that $\int \underset{\{1,\cdots,N\}}{\mathrm{Conf}}(\mathbb{C})$ etc. may be regarded as nothing but suggestive notation for types finitely presented by the Artin braid relations as in (32).)

Claim: Its transport operation is the monodromy braid representation
and its path lifting is execution of $\mathfrak{su}(2)$-anyon braid gates.

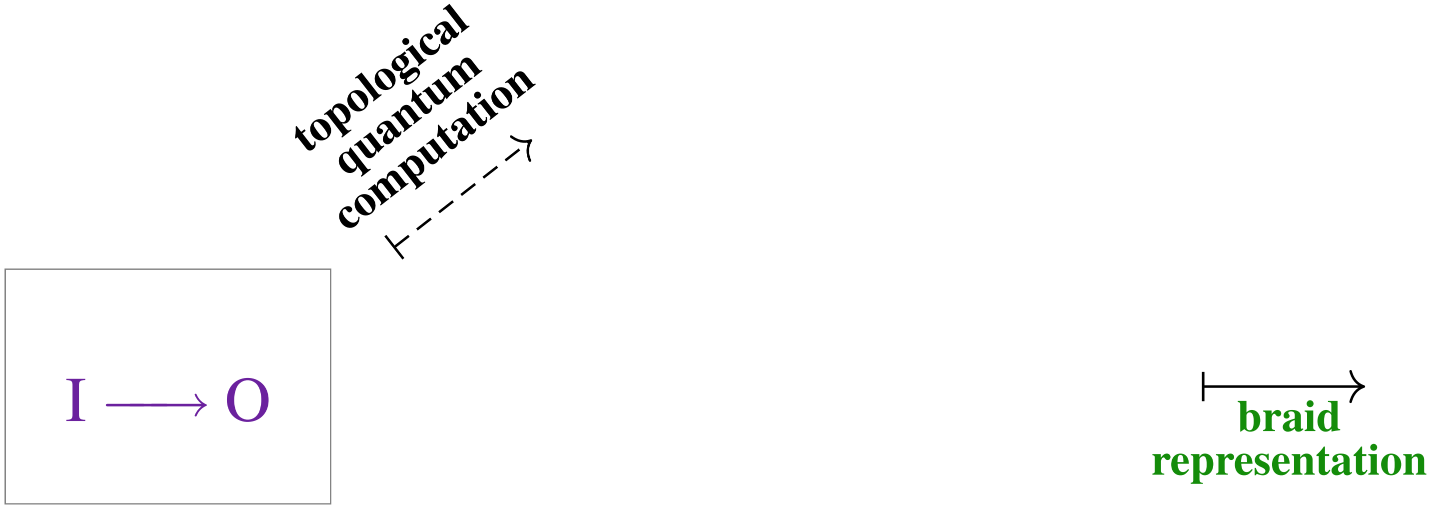
Hence coding this type family into a HoTT language like `Agda`
gives a Topological Quantum Programming Language
which is fully aware of topological anyon braid quantum gates.

To compute is

the case of
Topological Quantum Computation
[Sati & Schreiber, PlanQC **2022** 33 (2022)]

To compute is to **execute**

the case of
Topological Quantum Computation
[Sati & Schreiber, PlanQC **2022** 33 (2022)]



$I \longrightarrow O$

topological quantum computation

braid representation

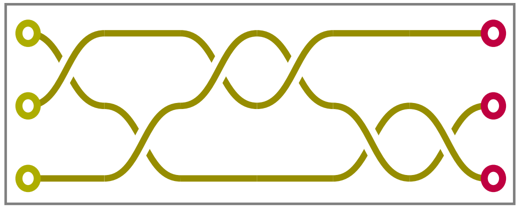
To compute is to **execute**
sequences of **instructions**

the case of
Topological Quantum Computation
[Sati & Schreiber, PlanQC **2022** 33 (2022)]

topological quantum computation

$I \longrightarrow O$

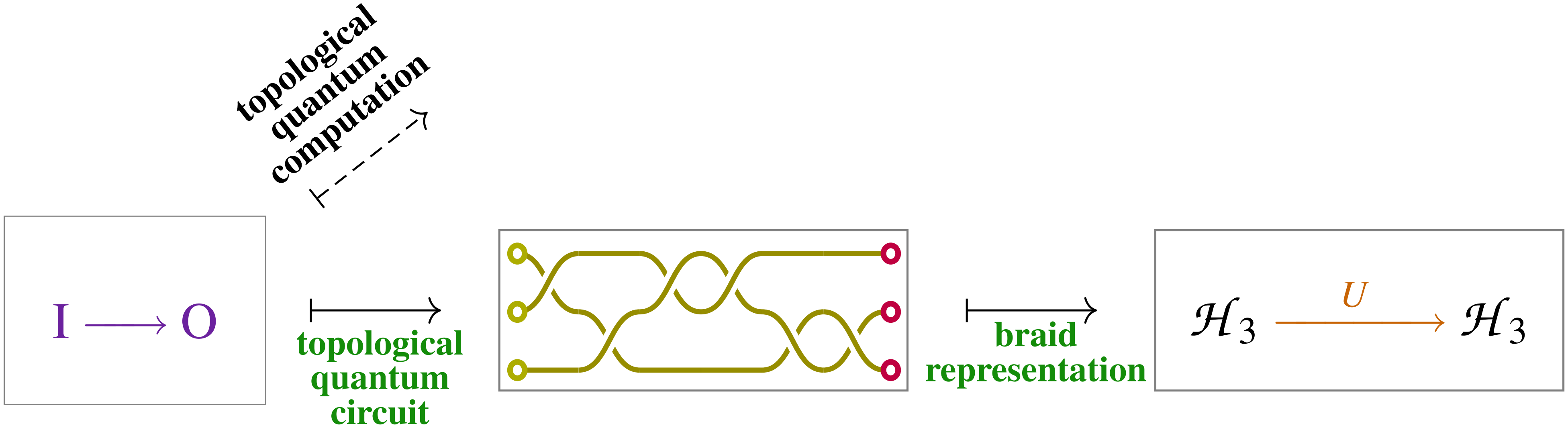
topological quantum circuit



braid representation

To compute is to **execute** sequences of **instructions** as composable **operations**

the case of
Topological Quantum Computation
[Sati & Schreiber, PlanQC **2022** 33 (2022)]



$\mathrm{I} \longrightarrow \mathrm{O}$

topological quantum computation

topological quantum circuit

braid representation

$\mathcal{H}_3 \xrightarrow{U} \mathcal{H}_3$

To compute is to **execute**
sequences of **instructions**
as composable **operations**
on a chosen **state space**,

the case of
Topological Quantum Computation
[Sati & Schreiber, PlanQC **2022** 33 (2022)]

$$\begin{tikzcd} \mathcal{H}_3 \arrow[r, "U"] & \mathcal{H}_3 \\ |\psi_{\mathrm{in}}\rangle \arrow[r, mapsto] & |\psi_{\mathrm{out}}\rangle \end{tikzcd}$$

$$\begin{tikzcd} {} \arrow[d, mapsto] \\ {} \end{tikzcd}$$

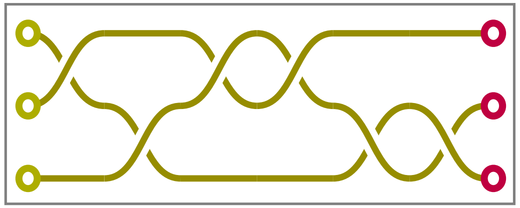
topological quantum computation

$$\begin{tikzcd} {} \arrow[r, mapsto, dashed] & {} \end{tikzcd}$$

$$\mathrm{I} \longrightarrow \mathrm{O}$$

$$\begin{tikzcd} {} \arrow[r, mapsto] & {} \end{tikzcd}$$

topological quantum circuit



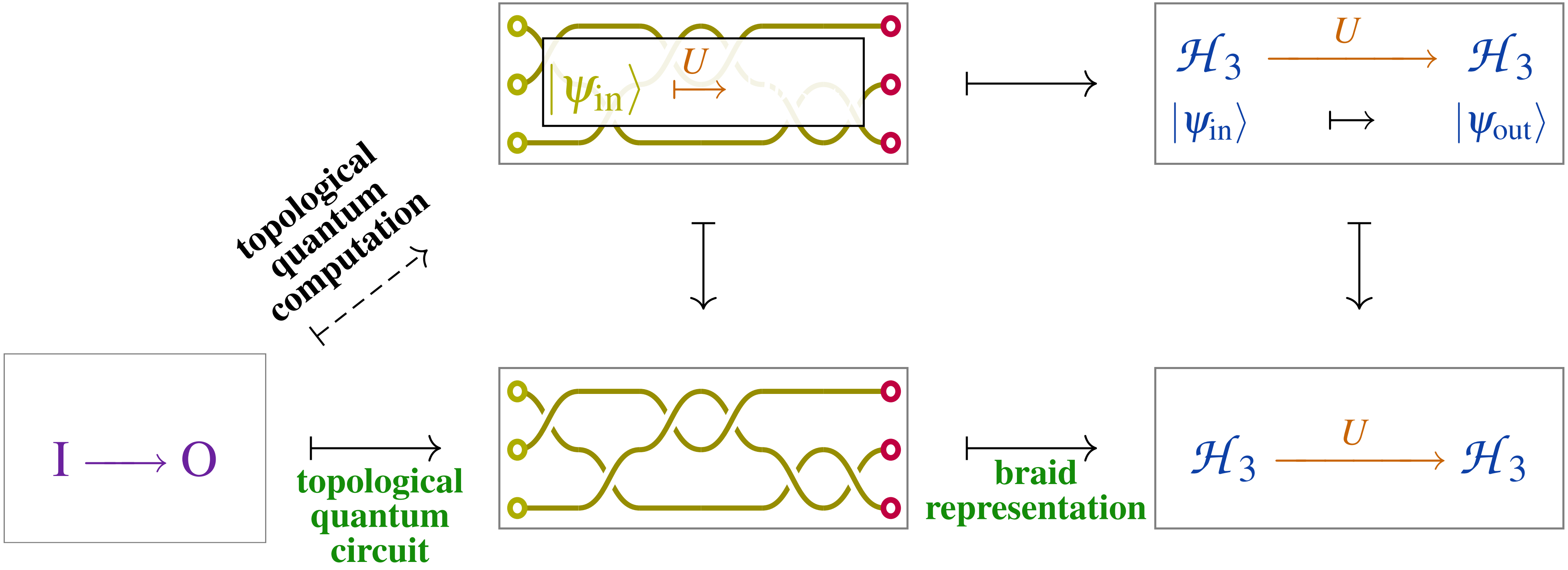
$$\begin{tikzcd} {} \arrow[r, mapsto] & {} \end{tikzcd}$$

braid representation

$$\begin{tikzcd} \mathcal{H}_3 \arrow[r, "U"] & \mathcal{H}_3 \end{tikzcd}$$

To compute is to **execute**
sequences of **instructions**
as composable **operations**
on a chosen **state space**,
turning a given **initial state**

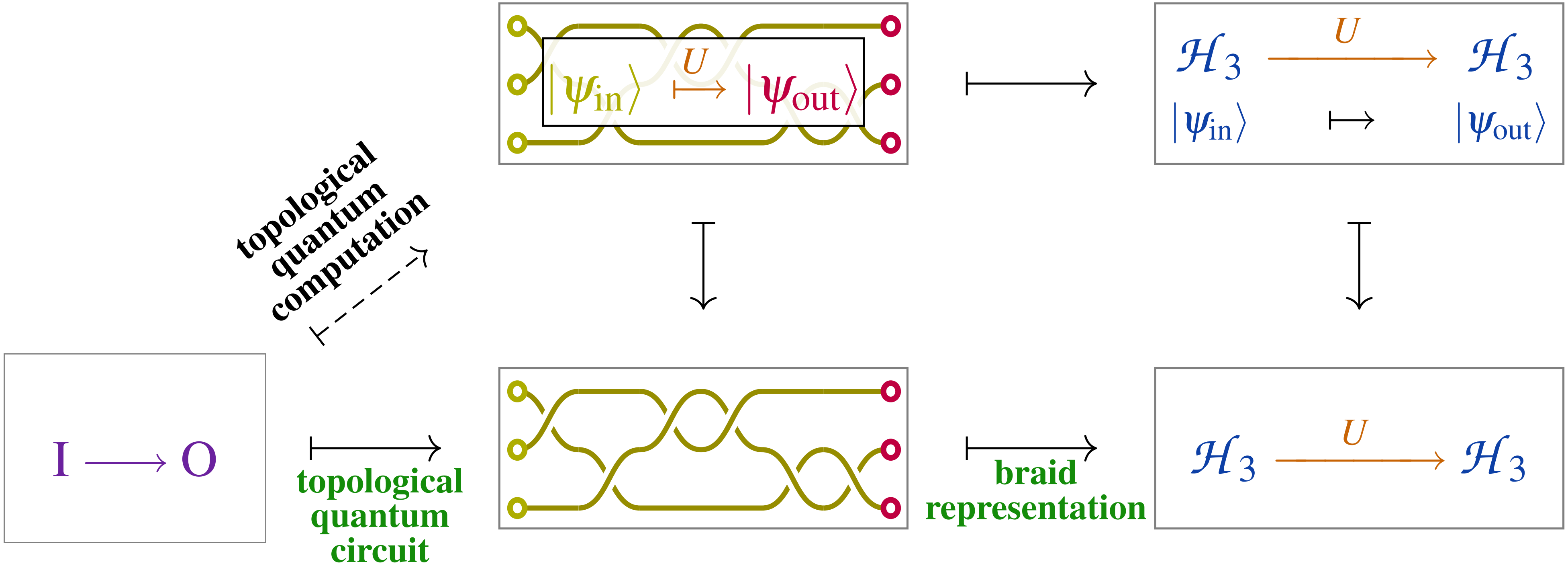
the case of
Topological Quantum Computation
[Sati & Schreiber, PlanQC **2022** 33 (2022)]



To compute is to **execute**
sequences of **instructions**
as composable **operations**
on a chosen **state space**,

turning a given **initial state**
into the computed **result**.

the case of
Topological Quantum Computation
[Sati & Schreiber, PlanQC **2022** 33 (2022)]



To compute is to **execute**
sequences of **instructions**
as composable **operations**
on a chosen **state space**,

turning a given **initial state**
into the computed **result**.

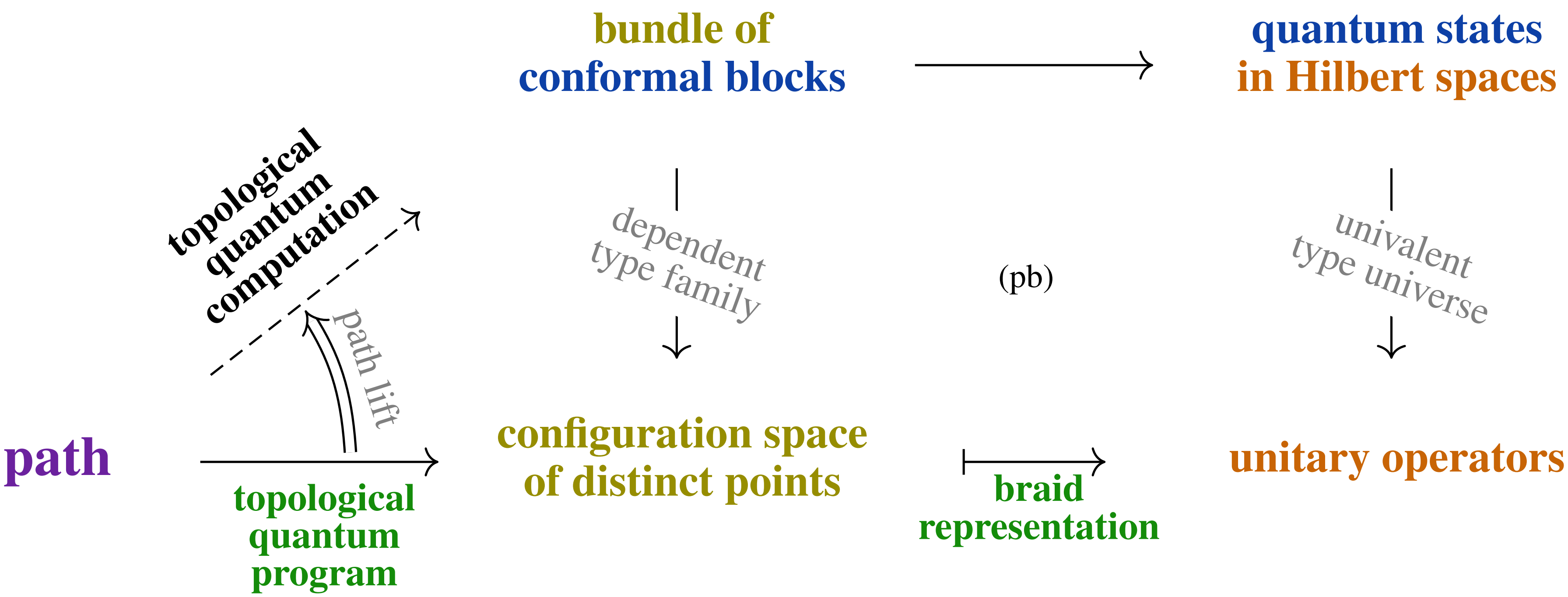
the case of
Topological Quantum Computation
[Sati & Schreiber, PlanQC **2022** 33 (2022)]

$$\begin{tikzcd}
 & \textbf{bundle of conformal blocks} \arrow[r] \arrow[d] \arrow[dr, phantom, "\text{(pb)}"] & \textbf{quantum states in Hilbert spaces} \arrow[d] \\
\textbf{path} \arrow[ur, dashed, "\textbf{topological quantum computation}"] \arrow[r, "\textbf{topological quantum program}"'] & \textbf{configuration space of distinct points} \arrow[r, mapsto, "\textbf{braid representation}"'] & \textbf{unitary operators}
\end{tikzcd}$$

To compute is to **execute**
sequences of **instructions**
as composable **operations**
on a chosen **state space**,
turning a given **initial state**
into the computed **result**.

the case of
Topological Quantum Computation
[Sati & Schreiber, PlanQC **2022** 33 (2022)]

Claim: This has natural construction in HoTT languages:



bundle of
conformal blocks

quantum states
in Hilbert spaces

dependent
type family

(pb)

univalent
type universe

topological
quantum
computation

path lift

path

topological
quantum
program

configuration space
of distinct points

braid
representation

unitary operators

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology;

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology;

$$\begin{tikzcd}[column sep=small]
\overset{\text{even-periodic de Rham cohomology}}{\underset{\substack{d \in \mathbb{Z} \\ 1 \leq r \leq \kappa}}{\bigoplus} H^{n+2d}\big(\Omega^\bullet_{\mathrm{dR}}(\mathrm{X};\mathbb{C}), \overset{\substack{\text{twisted by tensor powers of} \\ \text{the flat complex line bundle}}}{\mathrm{d} + r\cdot\omega_1\wedge}\big)} \arrow[rr] \arrow[dr, "\substack{\text{secondary} \\ \text{Chern character}}"'] & & \overset{\substack{BC^{+}_{\kappa}\text{-twisted} \\ C_\kappa\text{-equivariant} \\ \text{differential K-theory}}}{\underset{\substack{\text{smooth manifold inside} \\ C_\kappa\text{-orbi-singularity}}}{\mathrm{KU}^{n-1+[\omega_1]}_{\mathrm{diff}}\big(\mathrm{X}\times * /\!\!/ C_\kappa\big)}} \arrow[rr] \arrow[dr, "\text{forget}"'] & & \overset{\text{even-periodic twisted de Rham cohomology in opposite degree}}{\underset{\substack{d \in \mathbb{Z} \\ 1 \leq r \leq \kappa}}{\bigoplus} H^{n-1+2d}\big(\Omega^\bullet_{\mathrm{dR}}(\mathrm{X};\mathbb{C}), \mathrm{d} + r\cdot\omega_1\wedge\big)}. \\
 & \underset{\text{flat TED-K-theory}}{\mathrm{KU}^{n-1+[\omega_1]}_{\flat}\big(\mathrm{X}\times * /\!\!/ C_\kappa\big)} \arrow[ur, "\text{include}"] \arrow[rr] & & \underset{\text{underlying twisted equivariant K-theory}}{\mathrm{KU}^{n-1+[\omega_1]}\big(\mathrm{X}\times * /\!\!/ C_\kappa\big)} \arrow[ur, "\text{Chern character}"'] &
\end{tikzcd}$$

arXiv > hep-th > arXiv:2203.11838

High Energy Physics - Theory

[Submitted on 22 Mar 2022]

Anyonic Defect Branes and Conformal Blocks in Twisted Equivariant Differential (TED) K-theory

Hisham Sati, Urs Schreiber

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

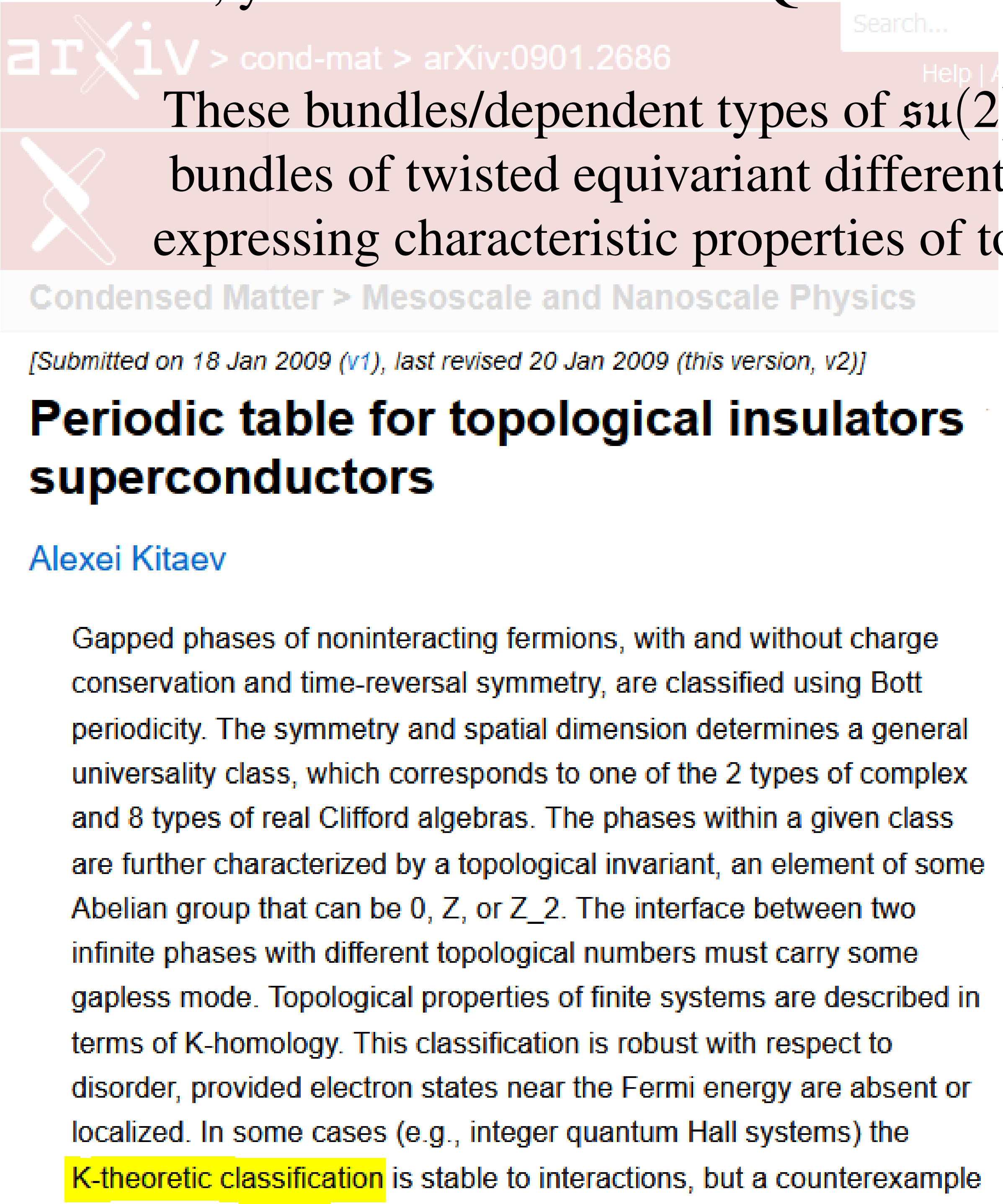
These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology;

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to
bundles of twisted equivariant differential (TED) K-cohomology;
expressing characteristic properties of topological phases of matter

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology; expressing characteristic properties of topological phases of matter



Condensed Matter > Mesoscale and Nanoscale Physics

[Submitted on 18 Jan 2009 (v1), last revised 20 Jan 2009 (this version, v2)]

Periodic table for topological insulators superconductors

Alexei Kitaev

Gapped phases of noninteracting fermions, with and without charge conservation and time-reversal symmetry, are classified using Bott periodicity. The symmetry and spatial dimension determines a general universality class, which corresponds to one of the 2 types of complex and 8 types of real Clifford algebras. The phases within a given class are further characterized by a topological invariant, an element of some Abelian group that can be 0, Z, or Z_2. The interface between two infinite phases with different topological numbers must carry some gapless mode. Topological properties of finite systems are described in terms of K-homology. This classification is robust with respect to disorder, provided electron states near the Fermi energy are absent or localized. In some cases (e.g., integer quantum Hall systems) the K-theoretic classification is stable to interactions, but a counterexample

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

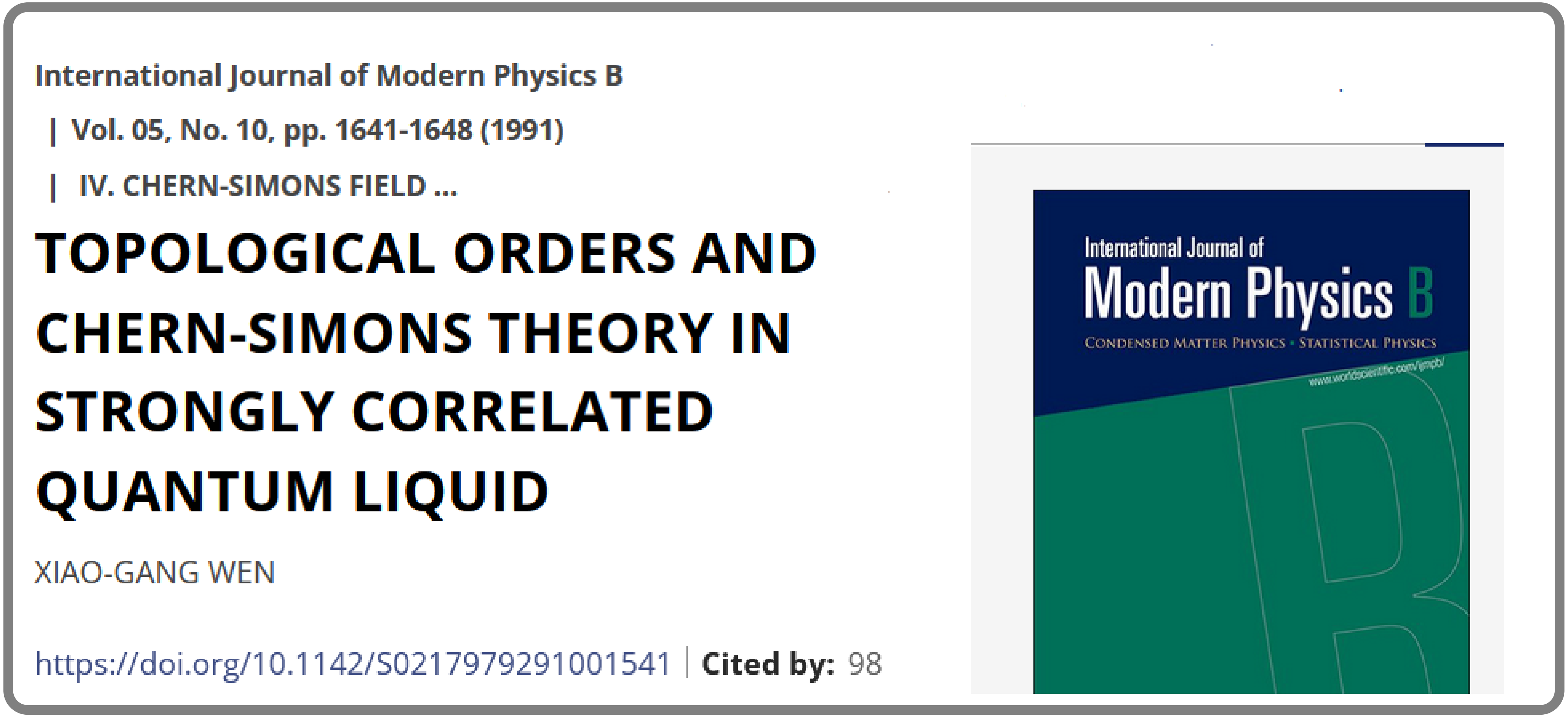
These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to
bundles of twisted equivariant differential (TED) K-cohomology;
expressing characteristic properties of topological phases of matter

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to
bundles of twisted equivariant differential (TED) K-cohomology;
expressing characteristic properties of topological phases of matter
hosting anyonic defects in their topologically ordered ground states.

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology; expressing characteristic properties of topological phases of matter hosting anyonic defects in their topologically ordered ground states.



International Journal of Modern Physics B

| Vol. 05, No. 10, pp. 1641-1648 (1991)

| IV. CHERN-SIMONS FIELD ...

TOPOLOGICAL ORDERS AND CHERN-SIMONS THEORY IN STRONGLY CORRELATED QUANTUM LIQUID

XIAO-GANG WEN

https://doi.org/10.1142/S0217979291001541 | **Cited by:** 98

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology; expressing characteristic properties of topological phases of matter hosting anyonic defects in their topologically ordered ground states.

arXiv > hep-th > arXiv:2206.13563

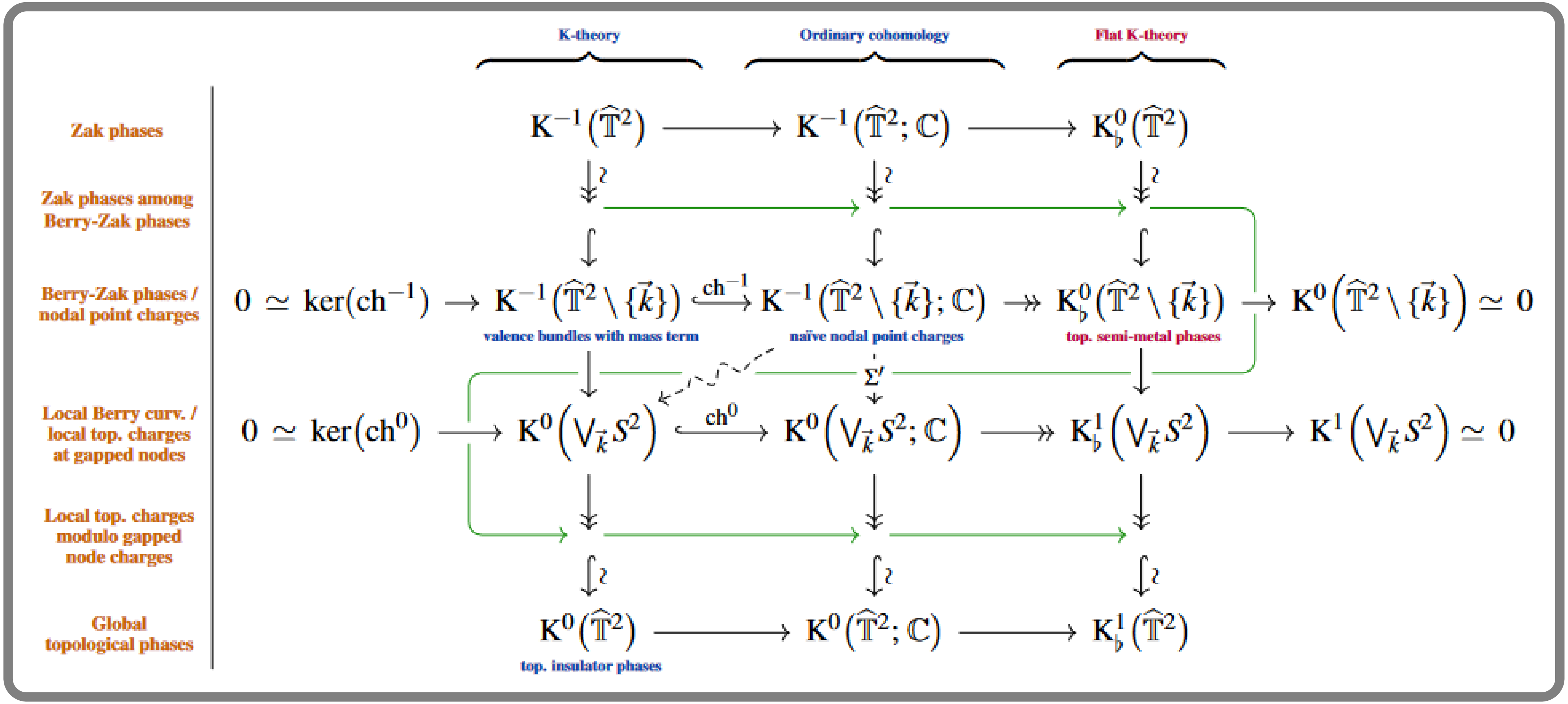
High Energy Physics - Theory

[Submitted on 27 Jun 2022]

Anyonic Topological Order in Twisted Equivariant Differential (TED) K-Theory

Hisham Sati, Urs Schreiber

expressing characteristic properties of topological phases of matter hosting anyonic defects in their topologically ordered ground states.



K-theory

Ordinary cohomology

Flat K-theory

Zak phases

Zak phases among Berry-Zak phases

Berry-Zak phases / nodal point charges

Local Berry curv. / local top. charges at gapped nodes

Local top. charges modulo gapped node charges

Global topological phases

$$\begin{tikzcd}
 & K^{-1}\big(\widehat{\mathbb{T}}^2\big) \arrow[r] \arrow[d, two heads, "\wr"] & K^{-1}\big(\widehat{\mathbb{T}}^2;\mathbb{C}\big) \arrow[r] \arrow[d, two heads, "\wr"] & K^0_\flat\big(\widehat{\mathbb{T}}^2\big) \arrow[d, two heads, "\wr"] & \\
 & {} \arrow[r] \arrow[d, hook] & {} \arrow[r] \arrow[d, hook] & {} \arrow[d, hook] \arrow[dddll] & \\
0 \simeq \ker(\mathrm{ch}^{-1}) \arrow[r] & K^{-1}\big(\widehat{\mathbb{T}}^2 \setminus \{\vec{k}\}\big) \arrow[r, hook, "\mathrm{ch}^{-1}"] \arrow[d] & K^{-1}\big(\widehat{\mathbb{T}}^2 \setminus \{\vec{k}\};\mathbb{C}\big) \arrow[r, two heads] \arrow[d, dashed, "\Sigma'"] \arrow[dl, dashed, squiggly] & K^0_\flat\big(\widehat{\mathbb{T}}^2 \setminus \{\vec{k}\}\big) \arrow[r] \arrow[d] & K^0\Big(\widehat{\mathbb{T}}^2 \setminus \{\vec{k}\}\Big) \simeq 0 \\
0 \simeq \ker(\mathrm{ch}^0) \arrow[r] & K^0\big(V_{\vec{k}}S^2\big) \arrow[r, hook, "\mathrm{ch}^0"] \arrow[d, two heads] & K^0\big(V_{\vec{k}}S^2;\mathbb{C}\big) \arrow[r, two heads] \arrow[d, two heads] & K^1_\flat\big(V_{\vec{k}}S^2\big) \arrow[r] \arrow[d, two heads] & K^1\big(V_{\vec{k}}S^2\big) \simeq 0 \\
 & {} \arrow[r] \arrow[d, hook, "\wr"] & {} \arrow[r] \arrow[d, hook, "\wr"] & {} \arrow[d, hook, "\wr"] & \\
 & K^0\big(\widehat{\mathbb{T}}^2\big) \arrow[r] & K^0\big(\widehat{\mathbb{T}}^2;\mathbb{C}\big) \arrow[r] & K^1_\flat\big(\widehat{\mathbb{T}}^2\big) &
\end{tikzcd}$$

valence bundles with mass term

naïve nodal point charges

top. semi-metal phases

top. insulator phases

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

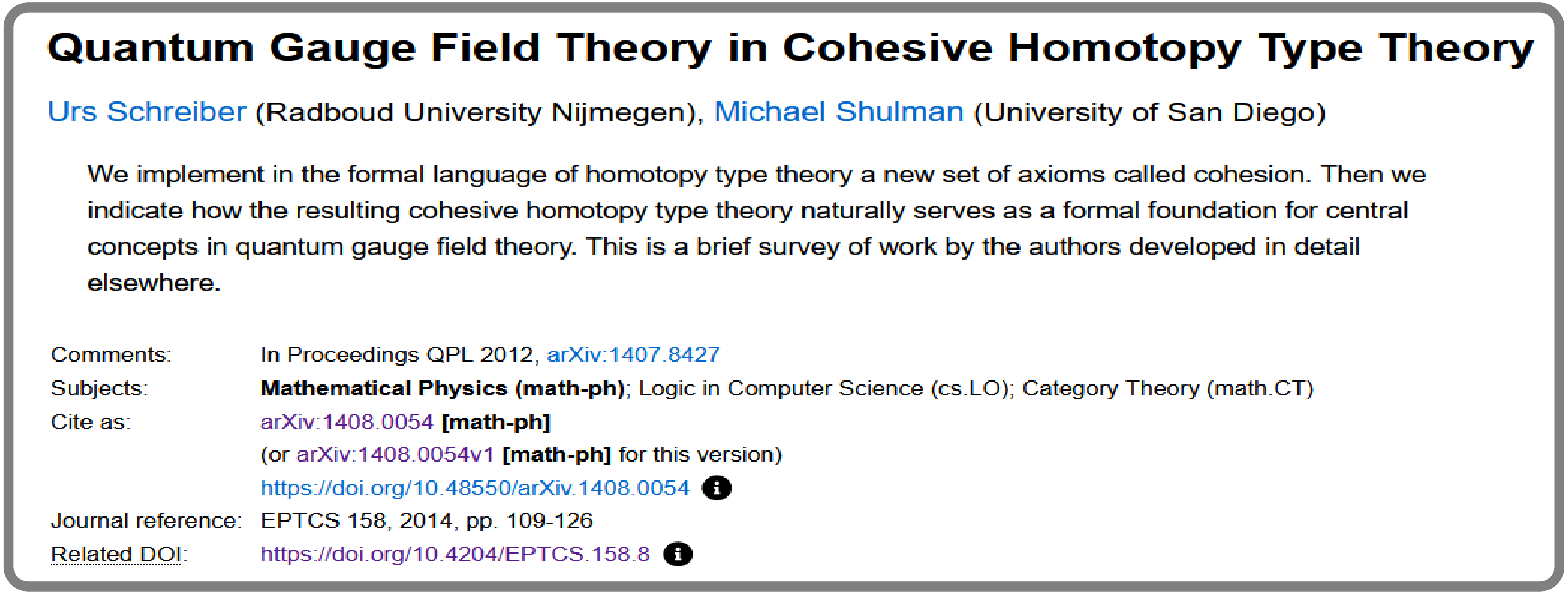
These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to
bundles of twisted equivariant differential (TED) K-cohomology;
expressing characteristic properties of topological phases of matter
hosting anyonic defects in their topologically ordered ground states.

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology; expressing characteristic properties of topological phases of matter hosting anyonic defects in their topologically ordered ground states.

But `TED-K` theory and hence topologically ordered phases are naturally expressible in an enhancement of HoTT languages called Cohesive HoTT.

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:



Quantum Gauge Field Theory in Cohesive Homotopy Type Theory

Urs Schreiber (Radboud University Nijmegen), Michael Shulman (University of San Diego)

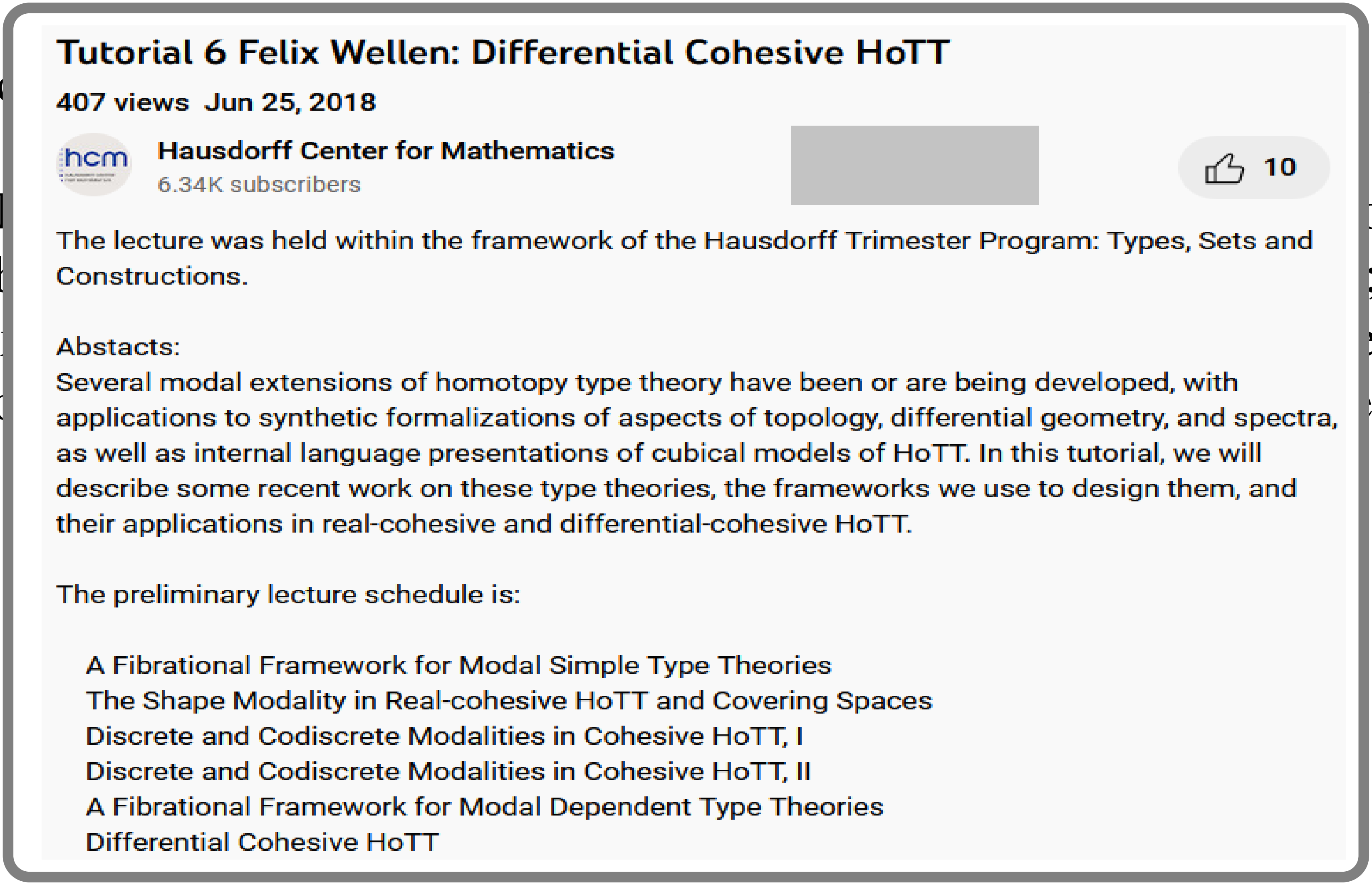
We implement in the formal language of homotopy type theory a new set of axioms called cohesion. Then we indicate how the resulting cohesive homotopy type theory naturally serves as a formal foundation for central concepts in quantum gauge field theory. This is a brief survey of work by the authors developed in detail elsewhere.

Comments:	In Proceedings QPL 2012, arXiv:1407.8427
Subjects:	**Mathematical Physics (math-ph)**; Logic in Computer Science (cs.LO); Category Theory (math.CT)
Cite as:	arXiv:1408.0054 **[math-ph]** (or arXiv:1408.0054v1 **[math-ph]** for this version) https://doi.org/10.48550/arXiv.1408.0054 ℹ
Journal reference:	EPTCS 158, 2014, pp. 109-126
Related DOI:	https://doi.org/10.4204/EPTCS.158.8 ℹ

But `TED-K` theory and hence topologically ordered phases
are naturally expressible in an enhancement of HoTT languages
called Cohesive HoTT.

In fa able:

T o
b ;
e er
h s.



Tutorial 6 Felix Wellen: Differential Cohesive HoTT

407 views Jun 25, 2018

Hausdorff Center for Mathematics
6.34K subscribers

The lecture was held within the framework of the Hausdorff Trimester Program: Types, Sets and Constructions.

Abstacts:
Several modal extensions of homotopy type theory have been or are being developed, with applications to synthetic formalizations of aspects of topology, differential geometry, and spectra, as well as internal language presentations of cubical models of HoTT. In this tutorial, we will describe some recent work on these type theories, the frameworks we use to design them, and their applications in real-cohesive and differential-cohesive HoTT.

The preliminary lecture schedule is:

- A Fibrational Framework for Modal Simple Type Theories
- The Shape Modality in Real-cohesive HoTT and Covering Spaces
- Discrete and Codiscrete Modalities in Cohesive HoTT, I
- Discrete and Codiscrete Modalities in Cohesive HoTT, II
- A Fibrational Framework for Modal Dependent Type Theories
- Differential Cohesive HoTT

are naturally expressible in an enhancement of HoTT languages called Cohesive HoTT.

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to
bundles of twisted equivariant differential (TED) K-cohomology;
expressing characteristic properties of topological phases of matter
hosting anyonic defects in their topologically ordered ground states.

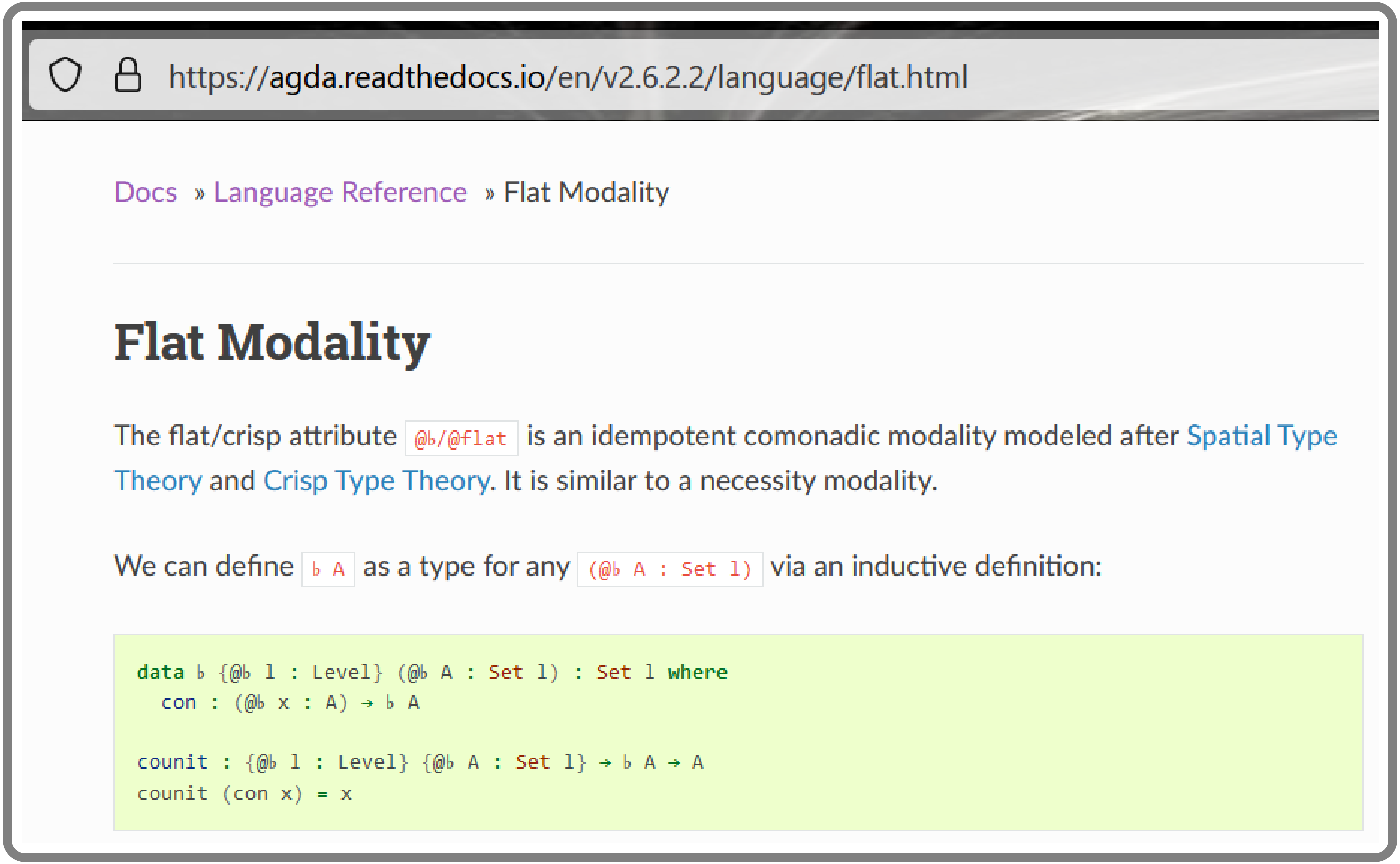
But `TED-K` and hence topologically ordered phases
are naturally expressible in an enhancement of HoTT languages
called Cohesive HoTT.

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology; expressing characteristic properties of topological phases of matter hosting anyonic defects in their topologically ordered ground states.

But `TED-K` and hence topologically ordered phases are naturally expressible in an enhancement of HoTT languages called Cohesive HoTT.

Parts of Cohesive HoTT have already been implemented in `Agda`.



https://agda.readthedocs.io/en/v2.6.2.2/language/flat.html

Docs » Language Reference » Flat Modality

Flat Modality

The flat/crisp attribute `@♭/@flat` is an idempotent comonadic modality modeled after Spatial Type Theory and Crisp Type Theory. It is similar to a necessity modality.

We can define `♭ A` as a type for any `(@♭ A : Set l)` via an inductive definition:

```
data ♭ {@♭ l : Level} (@♭ A : Set l) : Set l where
  con : (@♭ x : A) → ♭ A

counit : {@♭ l : Level} {@♭ A : Set l} → ♭ A → A
counit (con x) = x
```

Parts of Cohesive HoTT have already been implemented in `Agda`.

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to
bundles of twisted equivariant differential (TED) K-cohomology;
expressing characteristic properties of topological phases of matter
hosting anyonic defects in their topologically ordered ground states.

But `TED-K` and hence topologically ordered phases
are naturally expressible in an enhancement of HoTT languages
called Cohesive HoTT.

Parts of Cohesive HoTT have already been implemented in `Agda`.

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology; expressing characteristic properties of topological phases of matter hosting anyonic defects in their topologically ordered ground states.

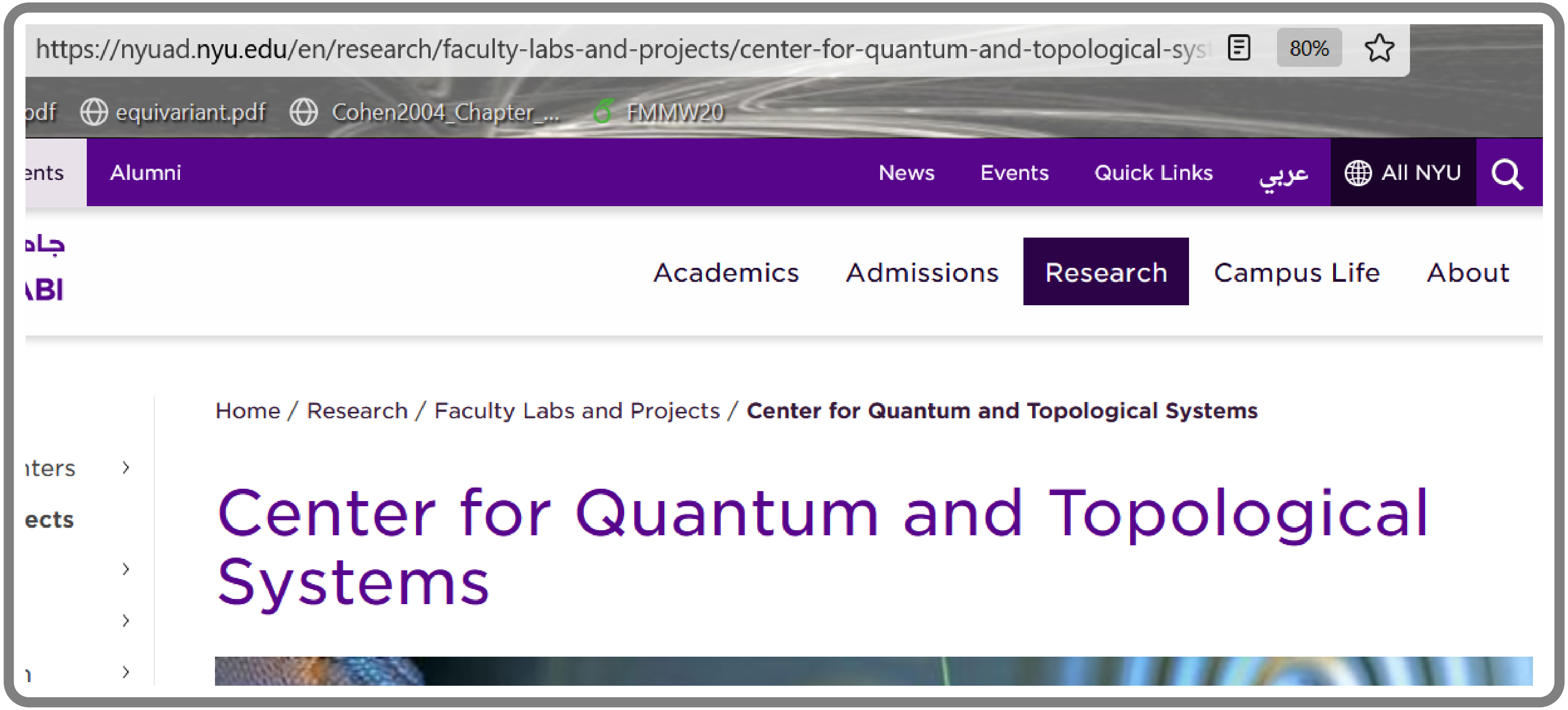
But `TED-K` and hence topologically ordered phases are naturally expressible in an enhancement of HoTT languages called Cohesive HoTT.

Parts of Cohesive HoTT have already been implemented in `Agda`.

Further development at CQTS.

In fact, yet more fine-detail of TQC hardware is naturally HoTT-codeable:

These bundles/dependent types of $\mathfrak{su}(2)$-conformal blocks map to bundles of twisted equivariant differential (TED) K-cohomology; expressing characteristic properties of topological phases of matter hosting anyonic defects in their topologically ordered ground states.



Home / Research / Faculty Labs and Projects / **Center for Quantum and Topological Systems**

Center for Quantum and Topological Systems

Further development at CQTS.

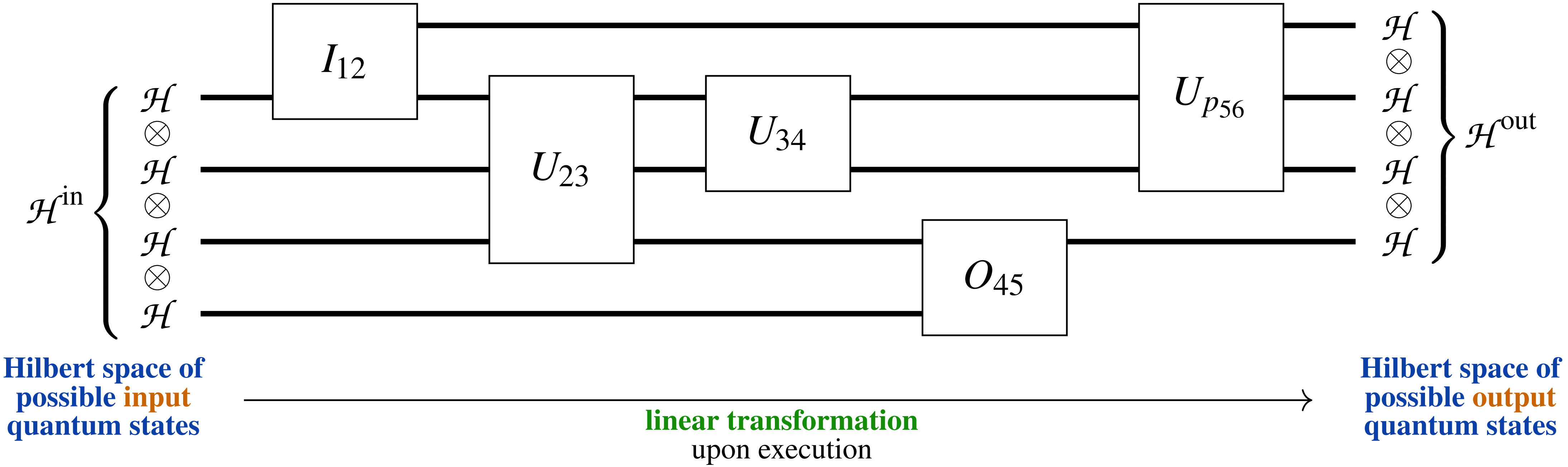
Part I
Verifying realistic topological quantum gates

Part II
Verifying their compilation into quantum circuits

The Problem

Pure quantum circuits are easy...

Linear operator composed & tensored from given *quantum logic gates*



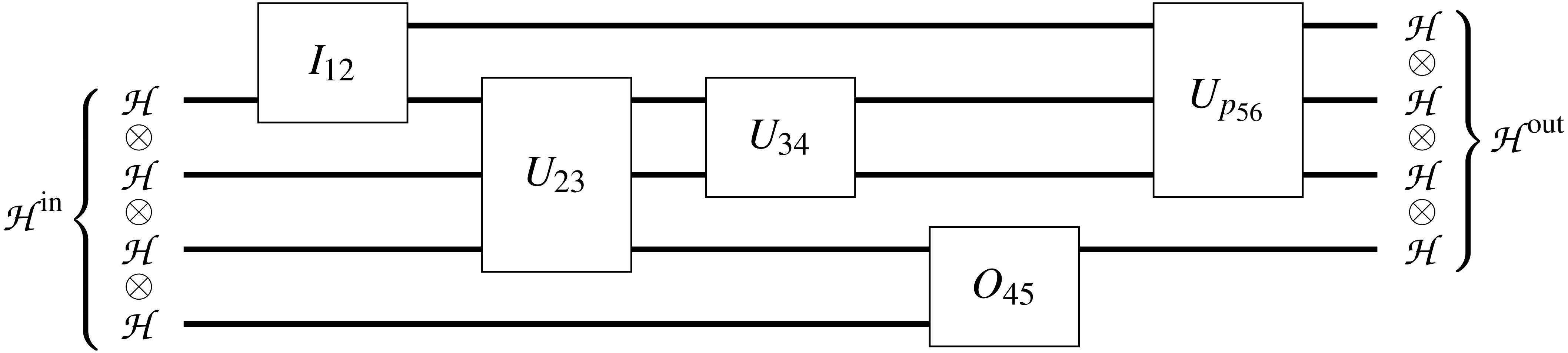
Hilbert space of possible input quantum states

linear transformation
upon execution

Hilbert space of possible output quantum states

Pure quantum circuits are easy...

Linear operator composed & tensored from given *quantum logic gates*



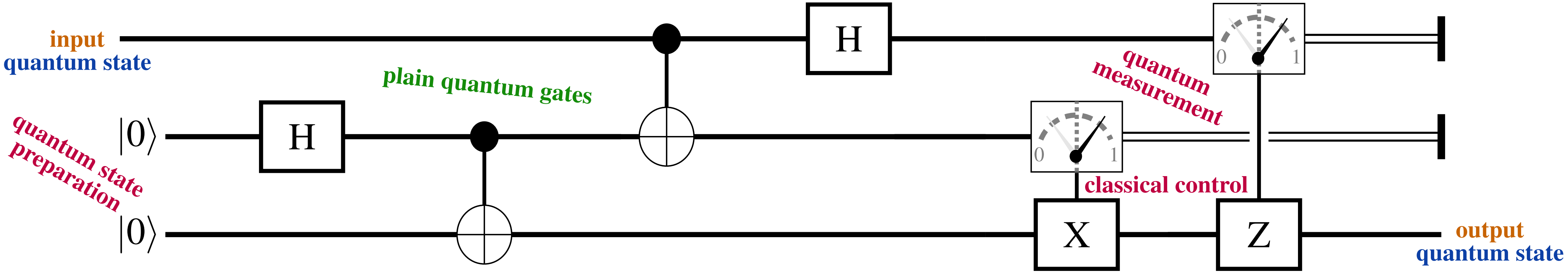
Hilbert space of possible input quantum states

linear transformation
upon execution

Hilbert space of possible output quantum states

but real quantum circuits have classical control & effects

(Example: QBit Teleportation protocol)



input quantum state

plain quantum gates

quantum state preparation

quantum measurement

classical control

output quantum state

full reality is a loop: $\text{Classical} \underset{\text{prepare}}{\overset{\text{measure}}{\leftrightarrows}} \text{Quantum}$

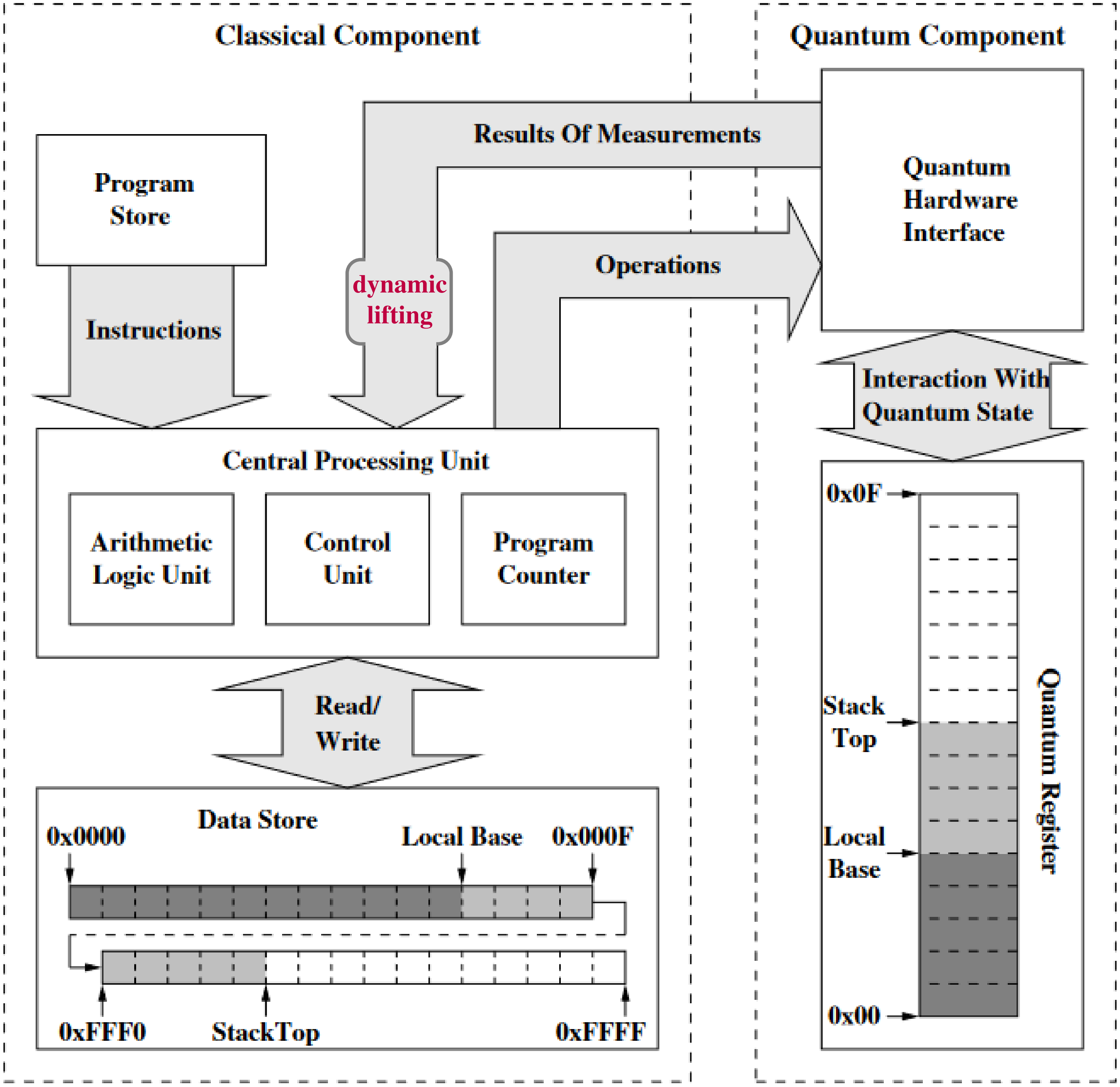
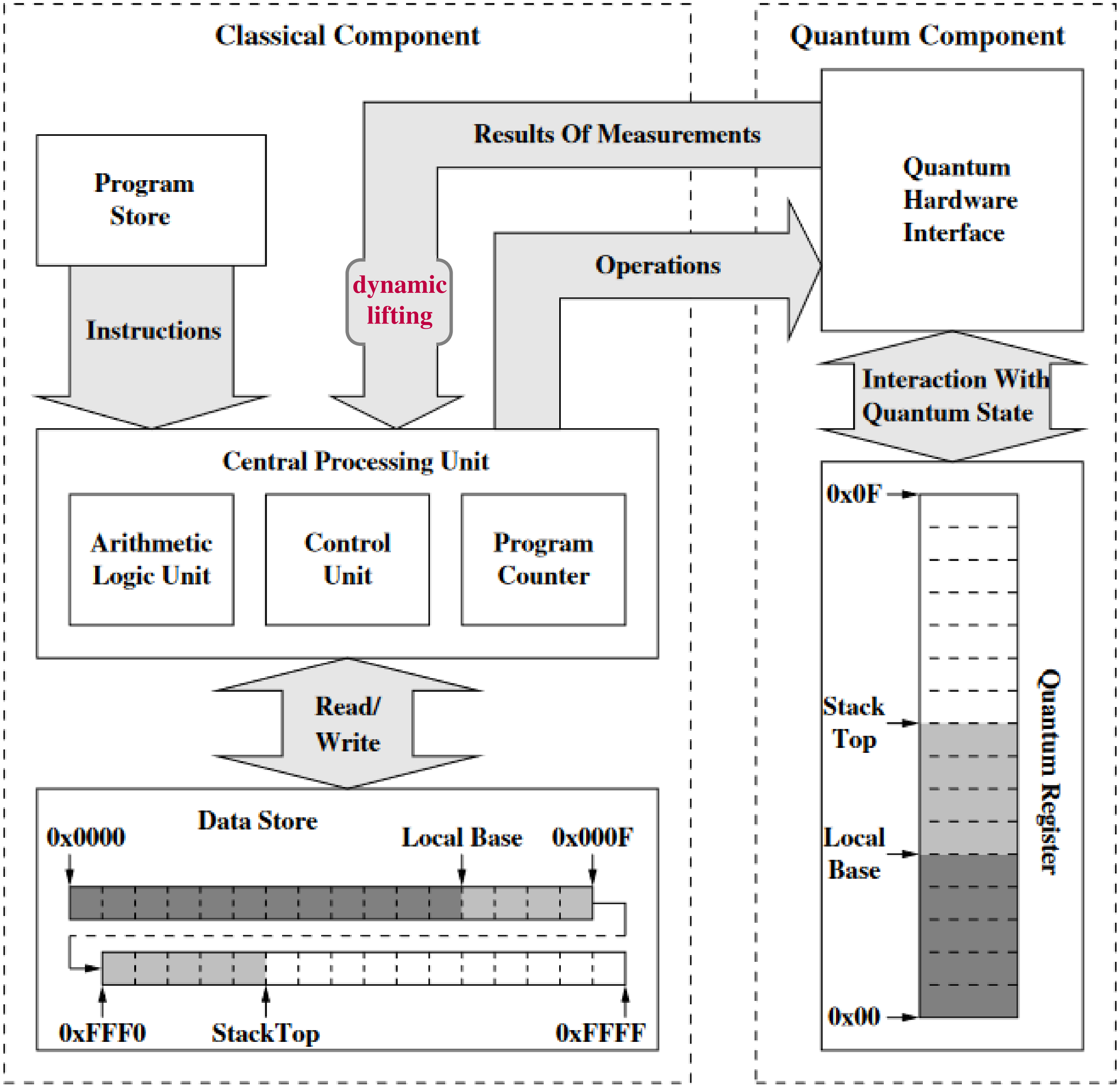


diagram adapted from Nagarajan et al. (2007)

full reality is a loop: Classical $\underset{\xrightarrow{\text{prepare}}}{\xleftarrow{\text{measure}}}$ Quantum



full loop needed e.g. for quantum error correction

diagram adapted from Nagarajan et al. (2007)

full reality is a loop: Classical $\underset{\text{prepare}}{\overset{\text{measure}}{\leftrightarrows}}$ Quantum

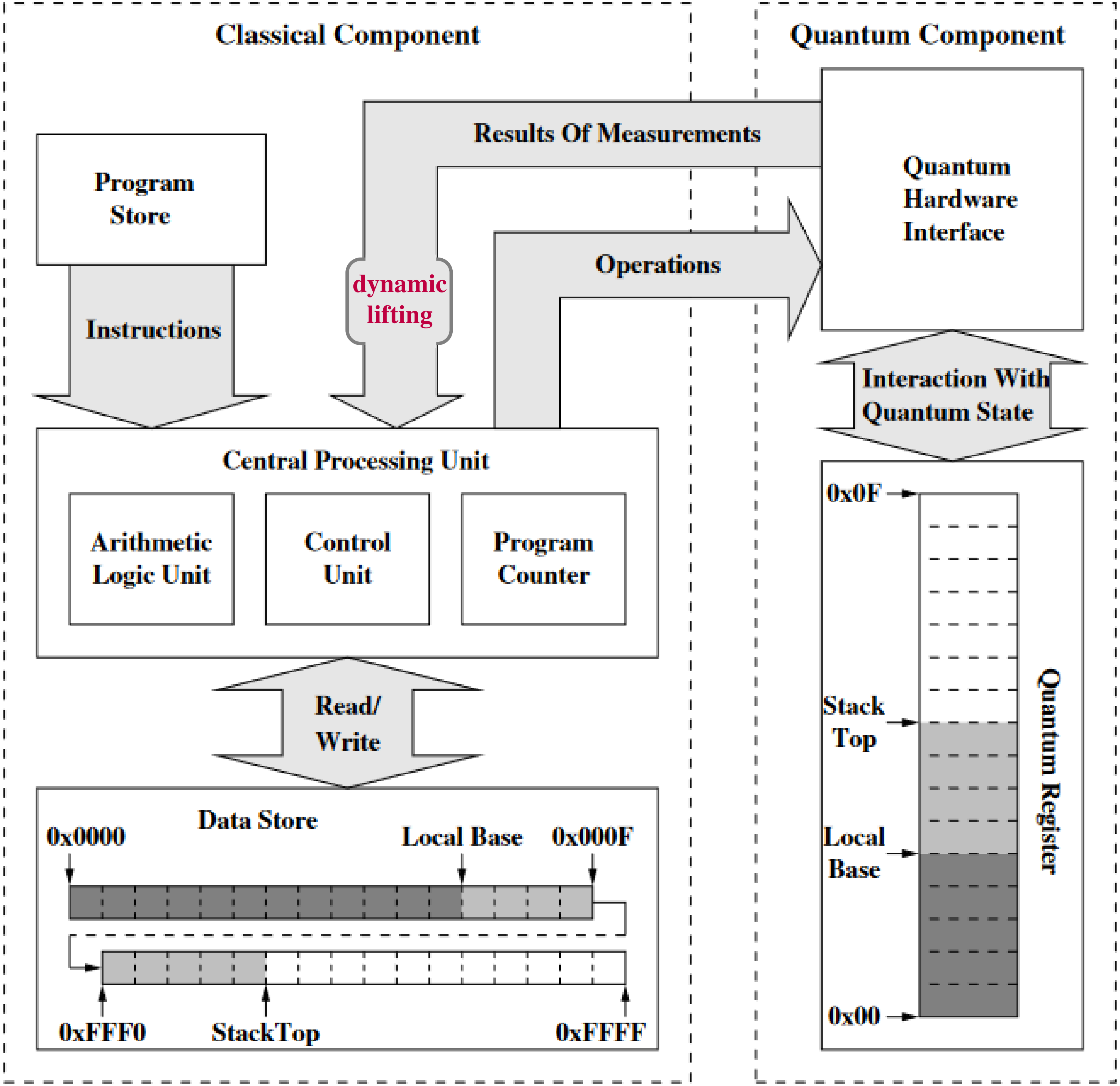


diagram adapted from Nagarajan et al. (2007)

full loop needed e.g. for quantum error correction

but its formal language theory had remained thin

full reality is a loop: Classical $\underset{\text{prepare}}{\overset{\text{measure}}{\leftrightarrows}}$ Quantum

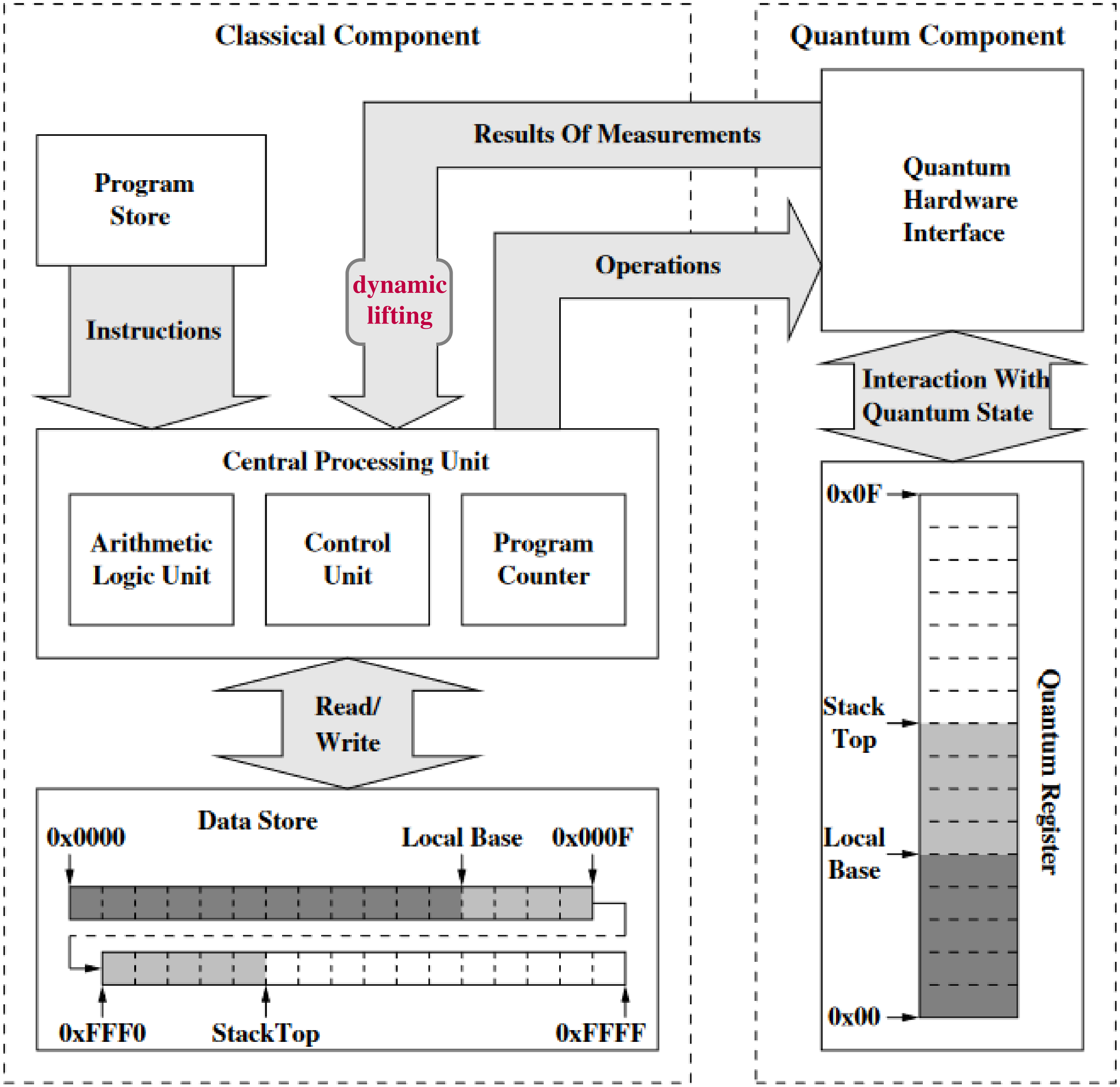


diagram adapted from Nagarajan et al. (2007)

full loop needed e.g. for quantum error correction

but its formal language theory had remained thin

existing models for dynamic lifting are ad hoc & unverified

Existing quantum typed circuit languages

are embedded inside *classical* type theories:

Existing quantum typed circuit languages

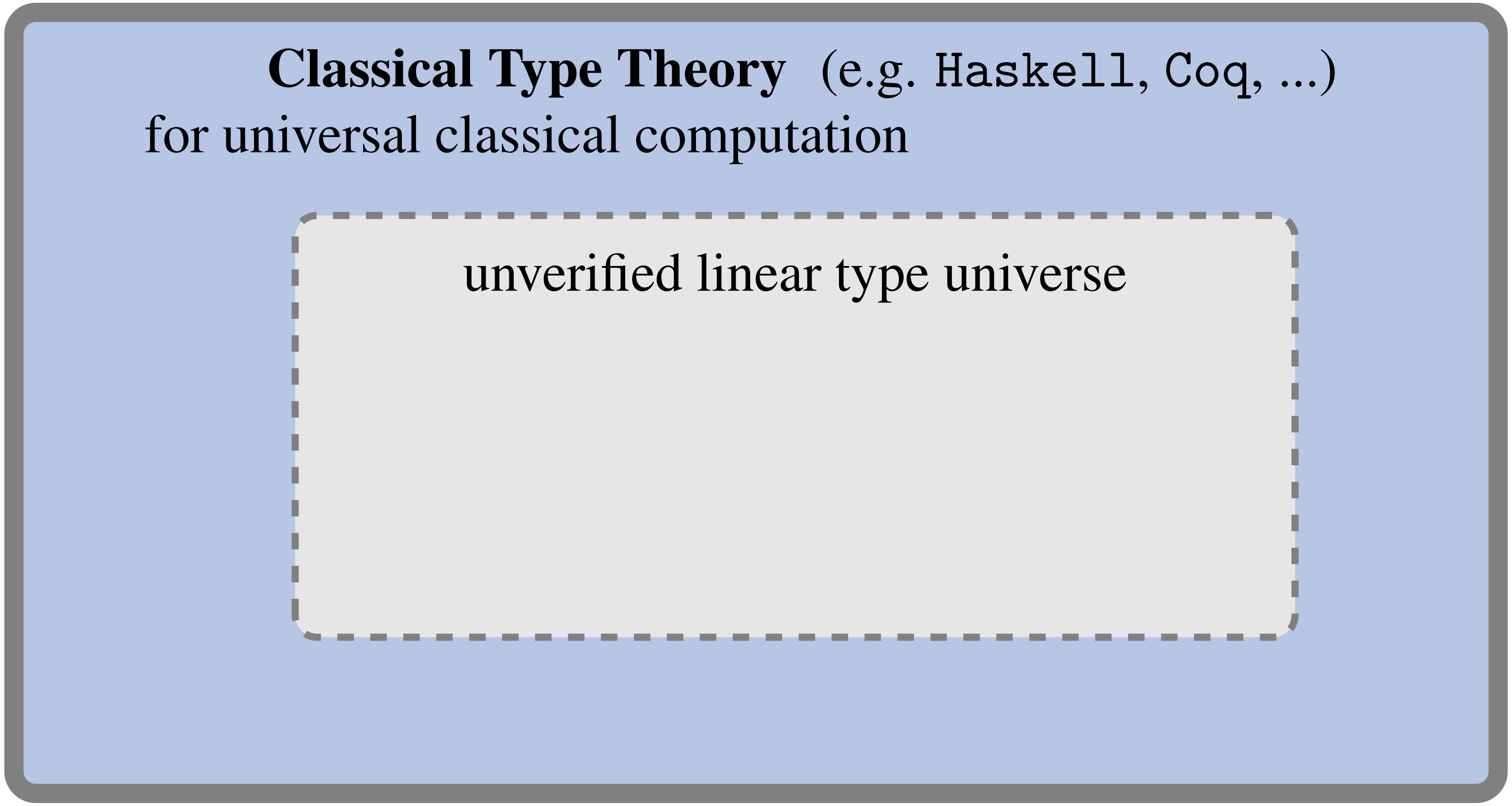
are embedded inside *classical* type theories:

Classical Type Theory (e.g. `Haskell`, `Coq`, ...)

for universal classical computation

Existing quantum typed circuit languages

are embedded inside *classical* type theories:



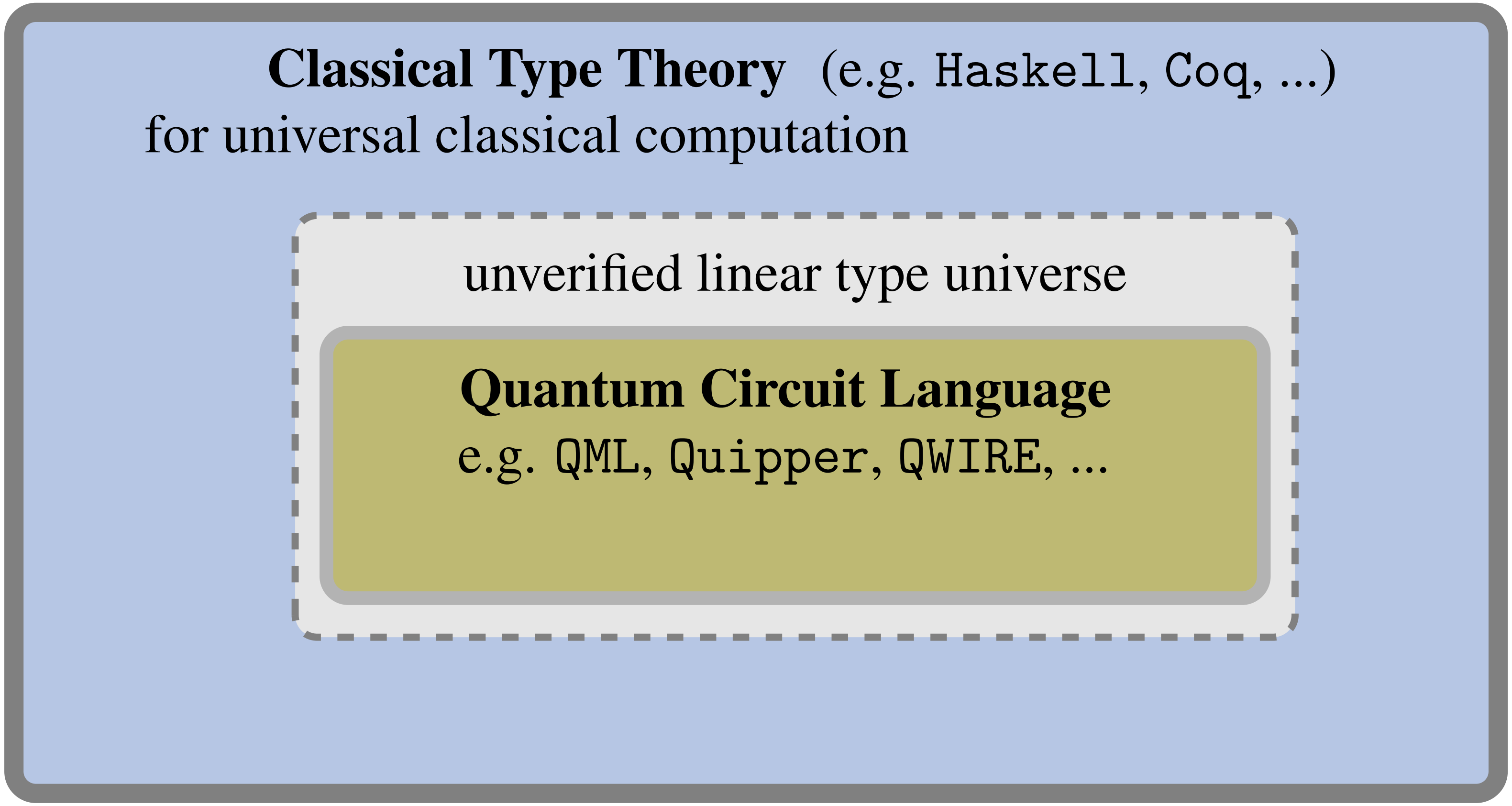
Classical Type Theory (e.g. `Haskell`, `Coq`, ...)

for universal classical computation

unverified linear type universe

Existing quantum typed circuit languages

are embedded inside *classical* type theories:



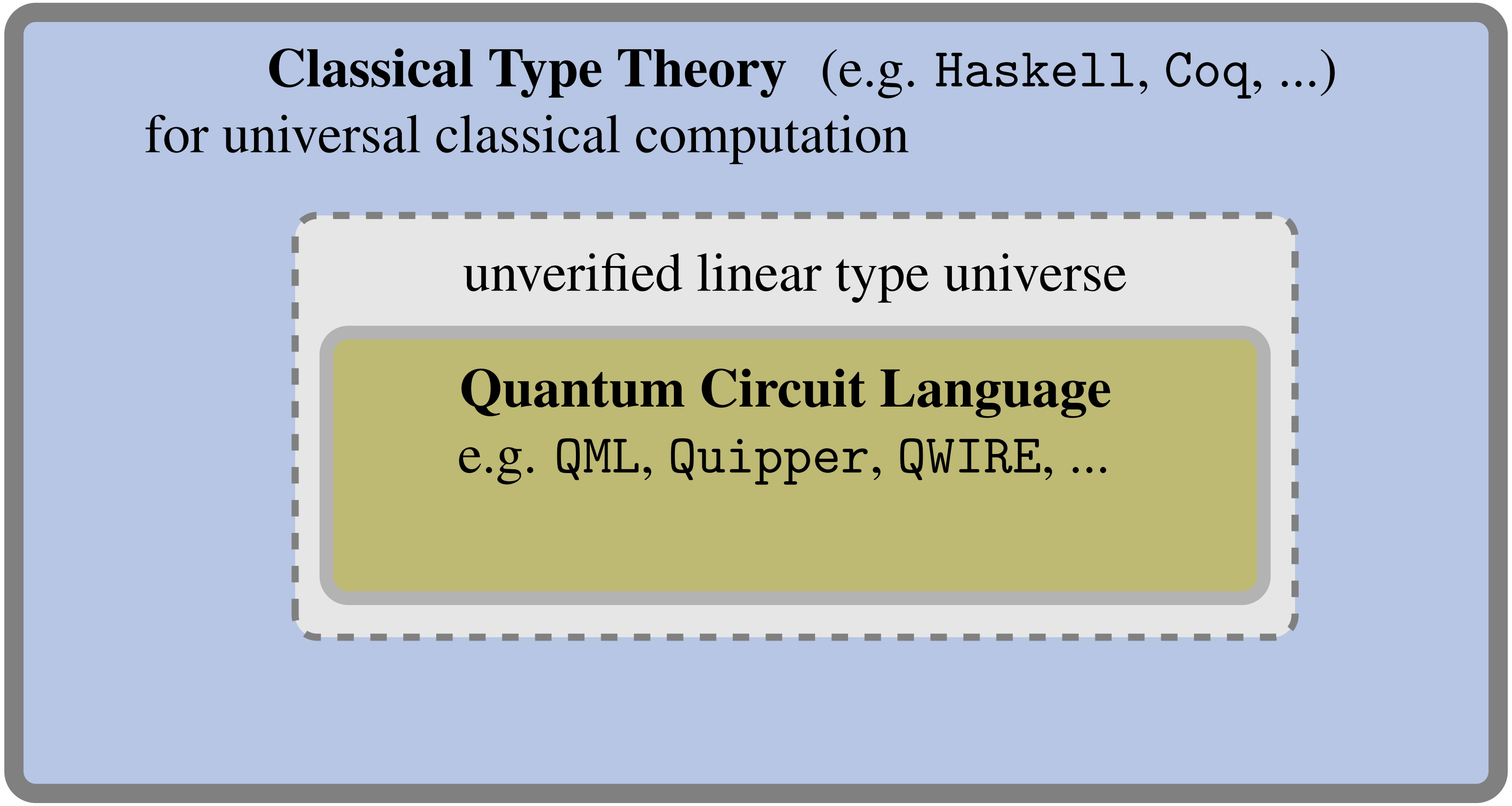
Classical Type Theory (e.g. `Haskell`, `Coq`, ...)
for universal classical computation

unverified linear type universe

Quantum Circuit Language
e.g. `QML`, `Quipper`, `QWIRE`, ...

Existing quantum typed circuit languages

are embedded inside *classical* type theories:



Classical Type Theory (e.g. `Haskell`, `Coq`, ...)
for universal classical computation

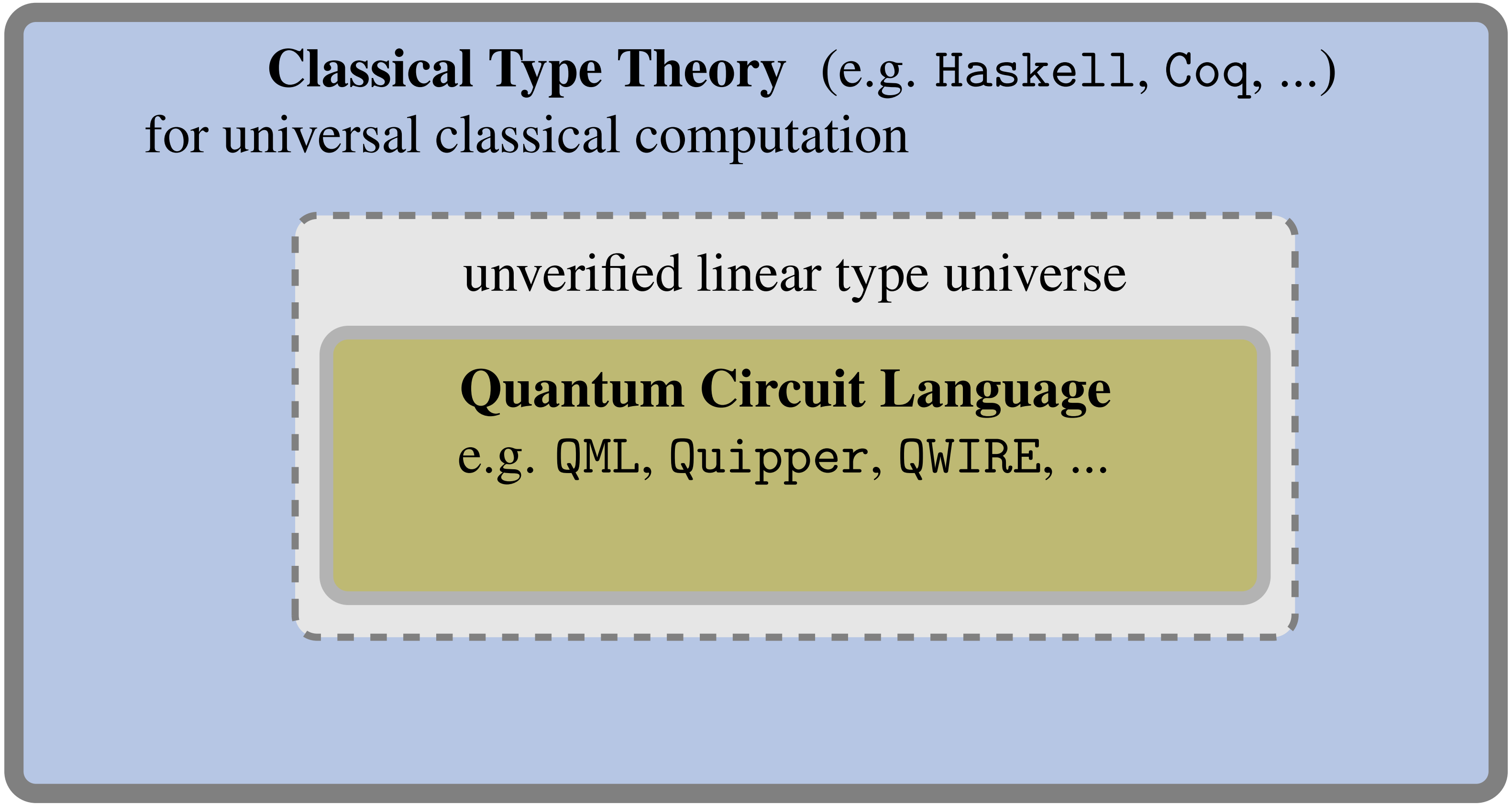
unverified linear type universe

Quantum Circuit Language
e.g. `QML`, `Quipper`, `QWIRE`, ...

for lack of a universal linear type theory.

Existing quantum typed circuit languages

are embedded inside *classical* type theories:



Classical Type Theory (e.g. `Haskell`, `Coq`, ...)
for universal classical computation

unverified linear type universe

Quantum Circuit Language
e.g. `QML`, `Quipper`, `QWIRE`, ...

for lack of a universal linear type theory.

Until now...

Our Solution

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

$\exists$ *universal* quantum+classical specification language

conservative over classical *Homotopy Type Theory* (`HoTT`)

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

∃ *universal* quantum+classical specification language

conservative over classical *Homotopy Type Theory* (`HoTT`)

and

verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]

(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

∃ *universal* quantum+classical specification language

conservative over classical *Homotopy Type Theory* (`HoTT`)

and

verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]

(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

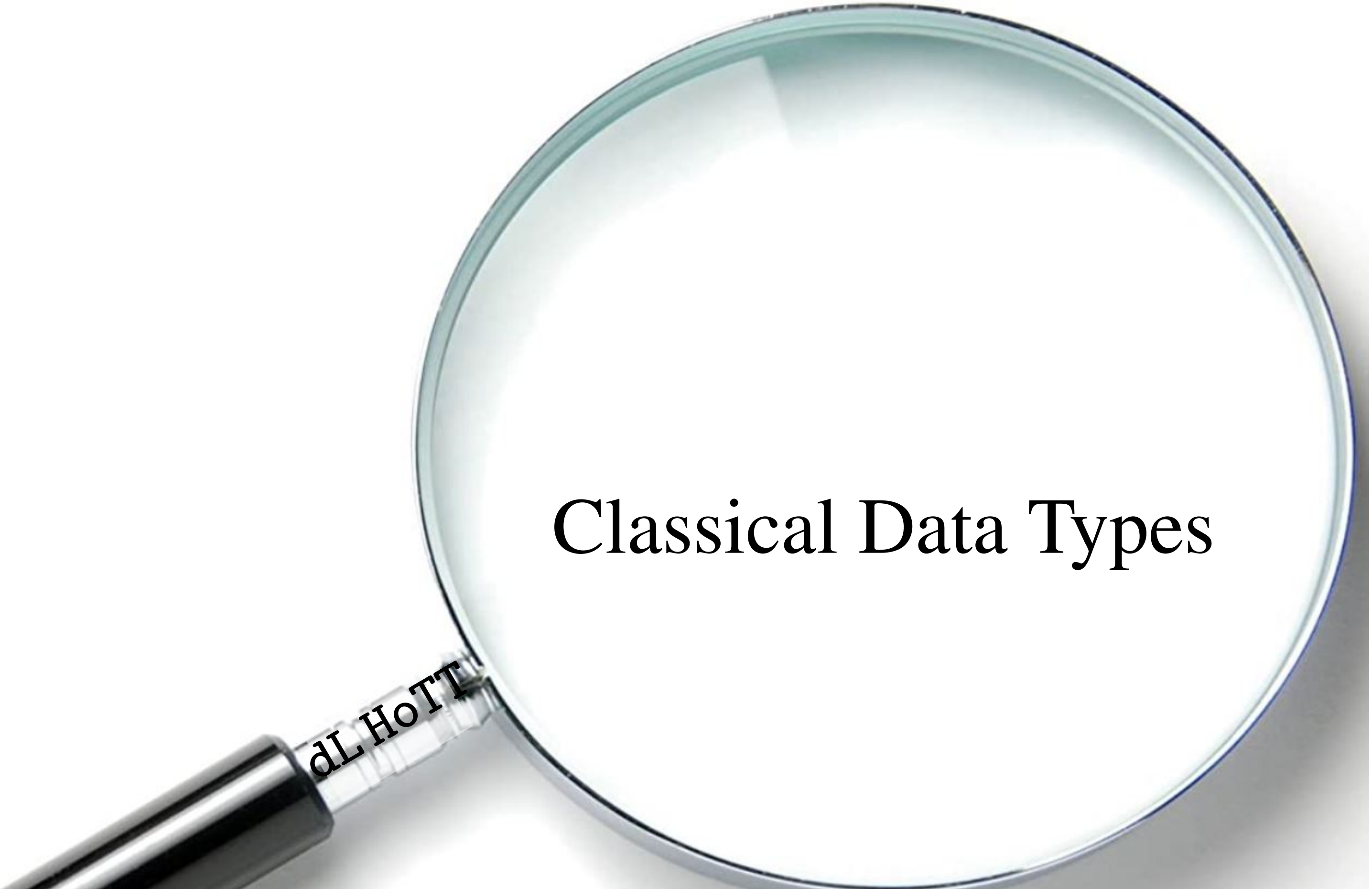
Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

`LHoTT` is like a quantum microscope for Classical Data Types

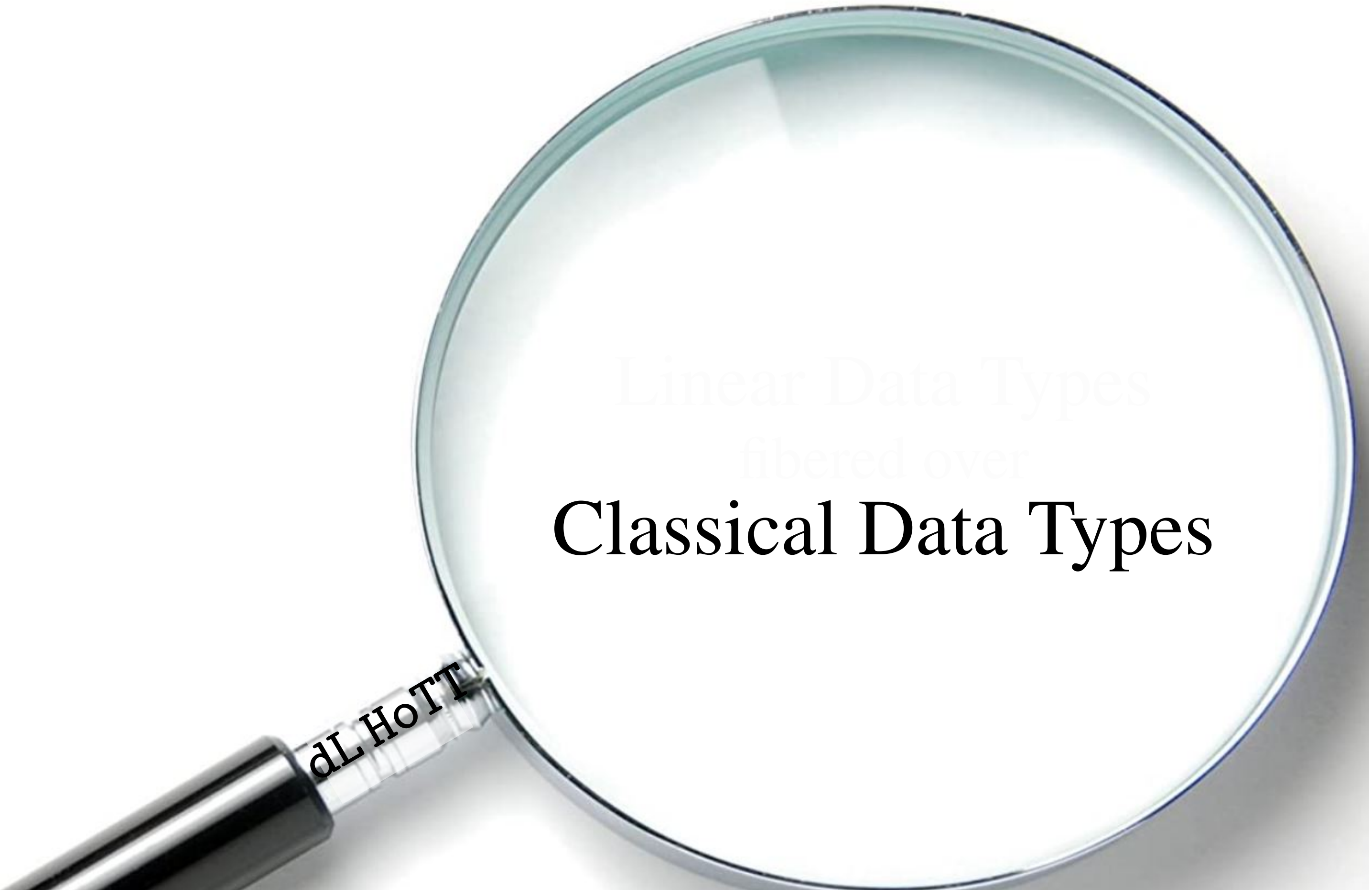
Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Dependent Linear Homotopy Type Theory (`LHoTT`)

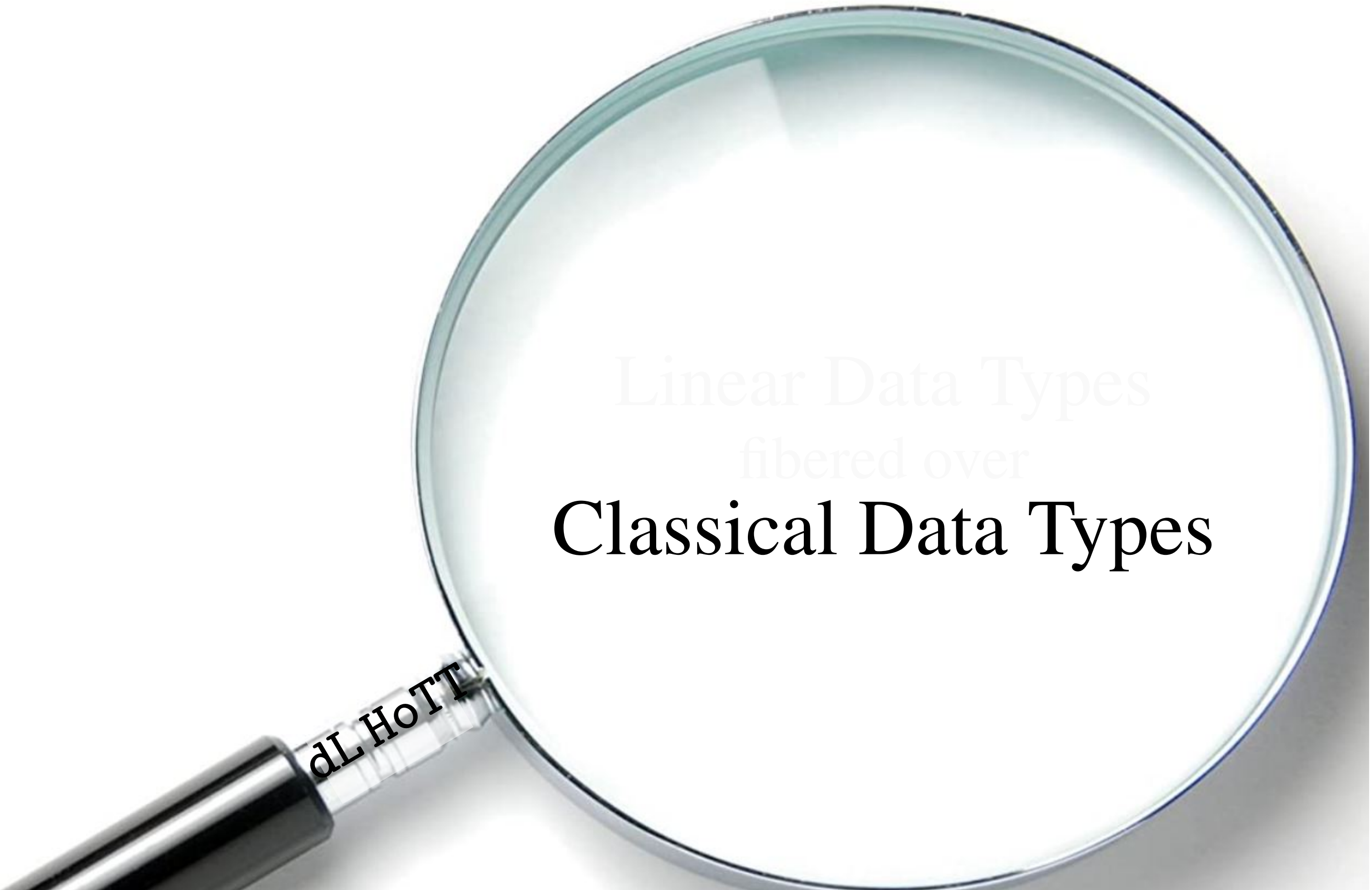
Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Dependent Linear Homotopy Type Theory (`LHoTT`)

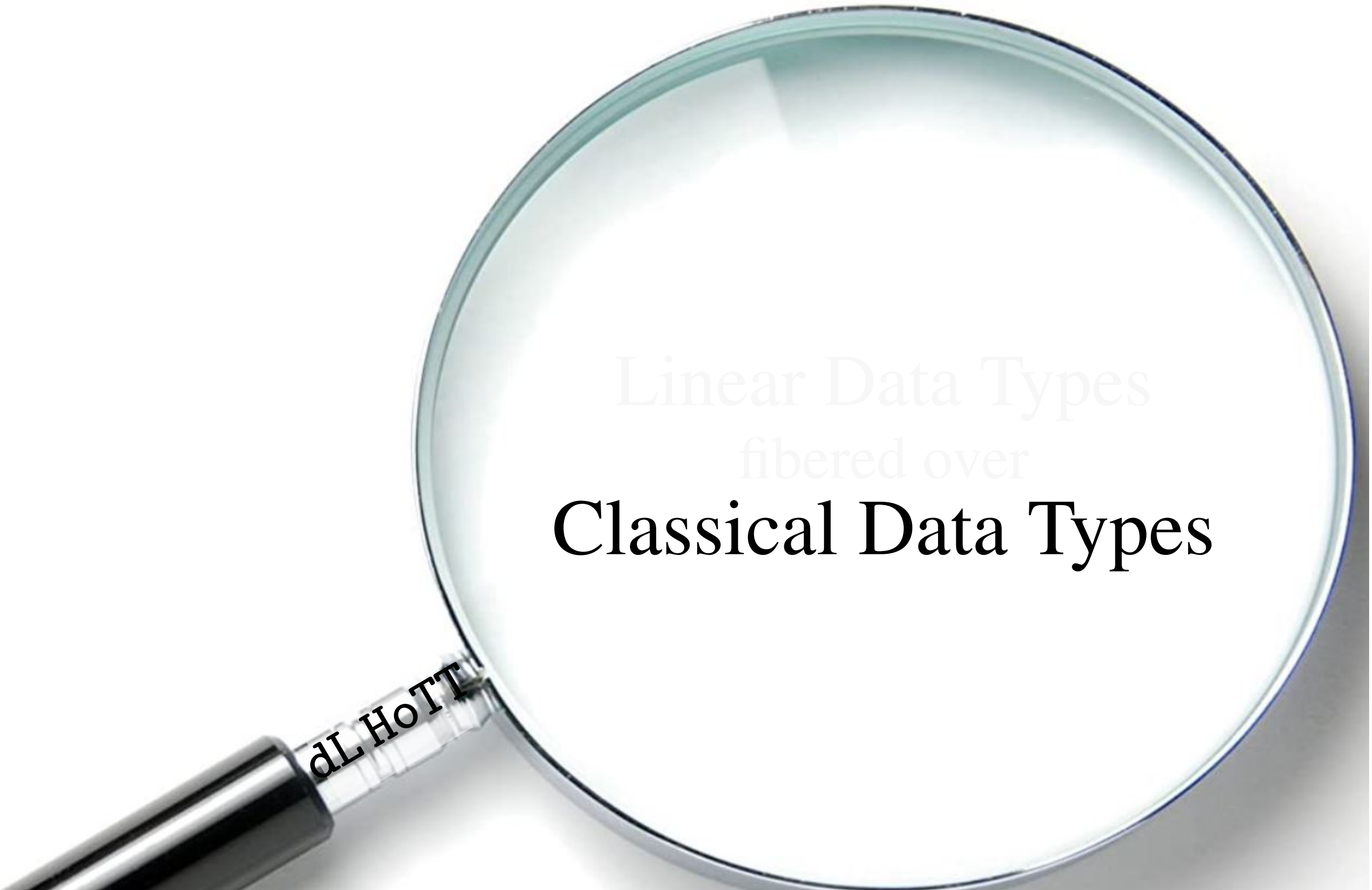
Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme “Motivic Yoga” [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck’s six operations *à la* Wirthmüller — more on all this below)



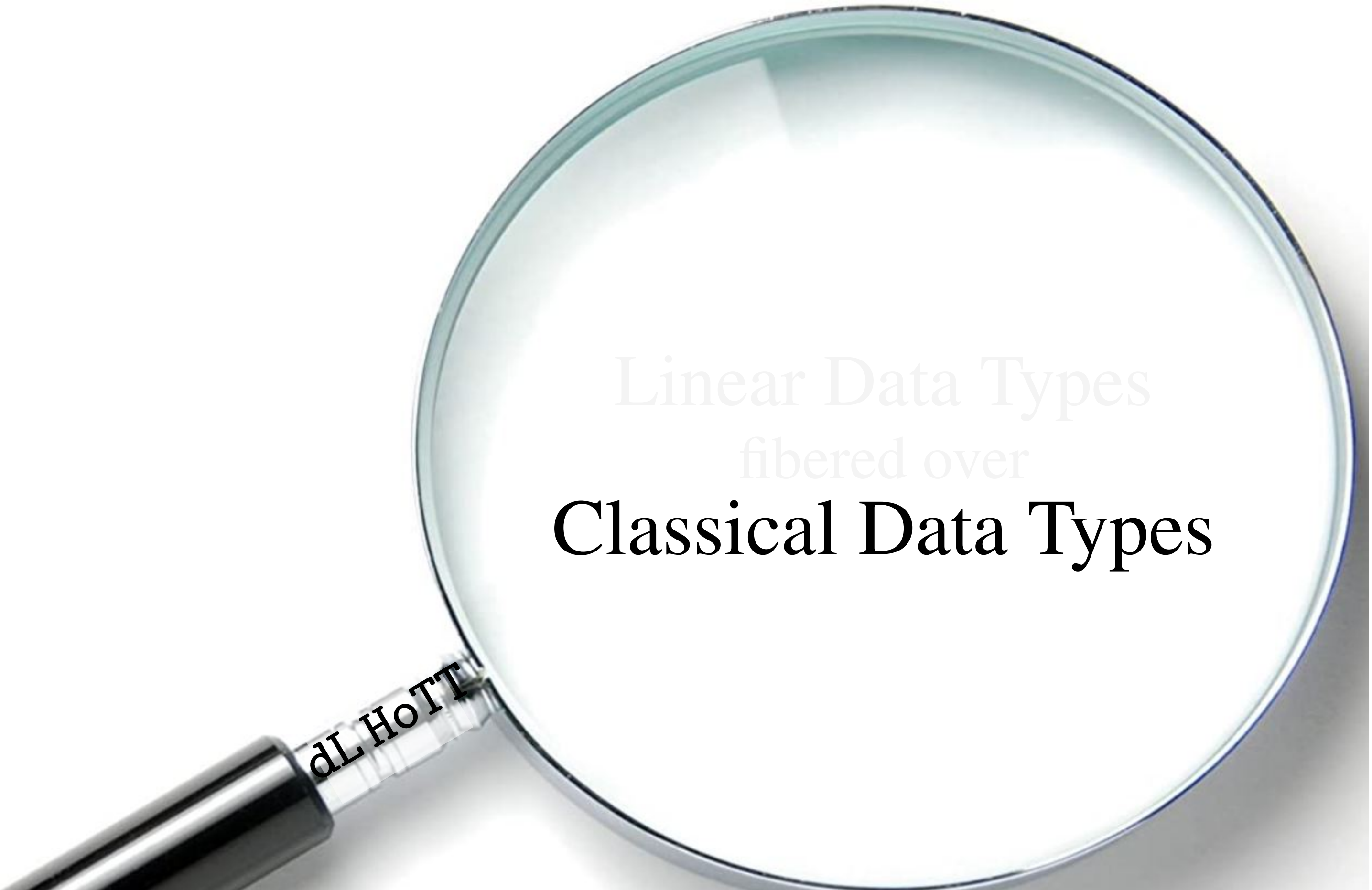
Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

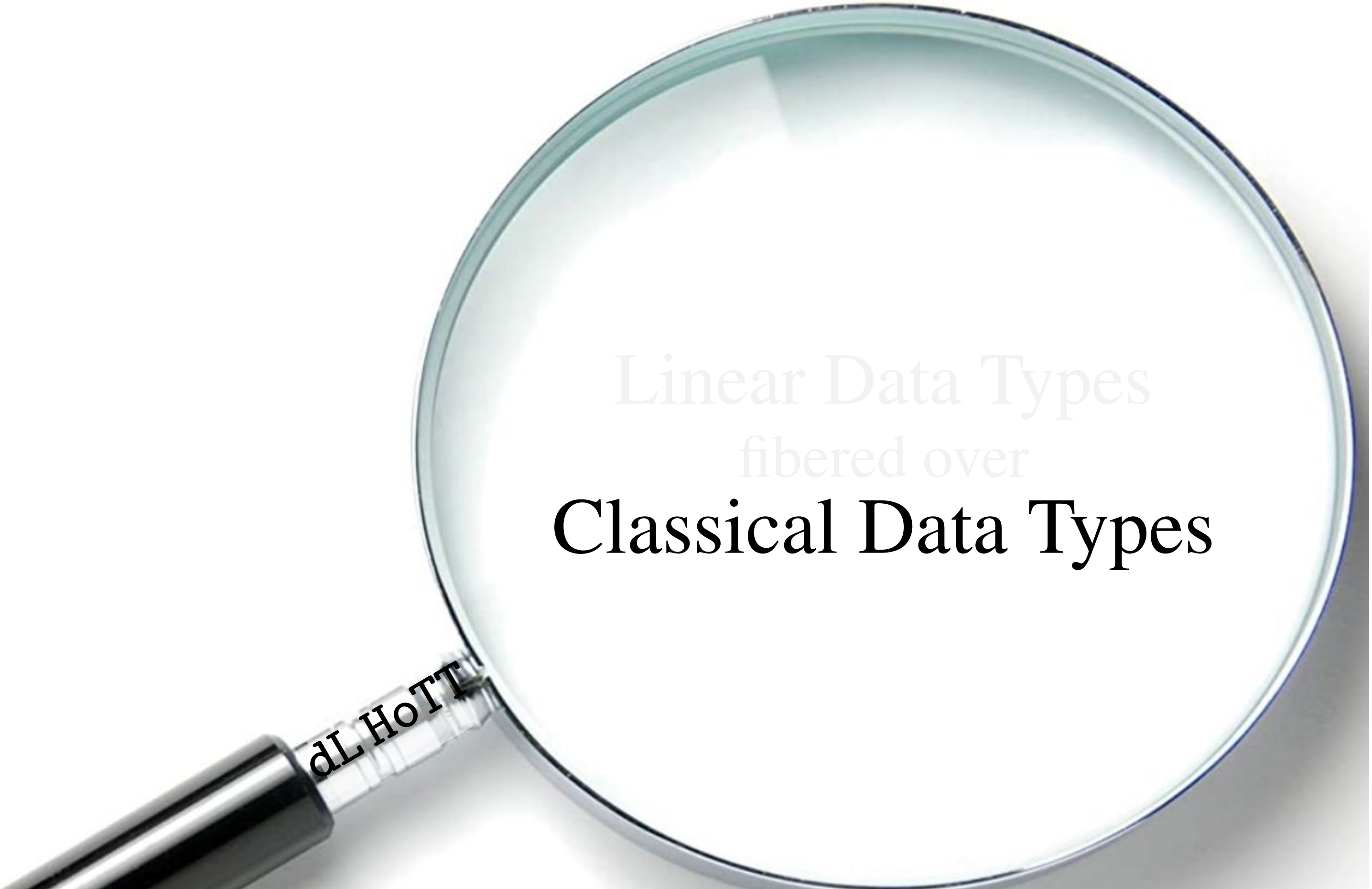
$\exists$ *universal* quantum+classical specification language

conservative over classical *Homotopy Type Theory* (`HoTT`)

and

verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]

(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

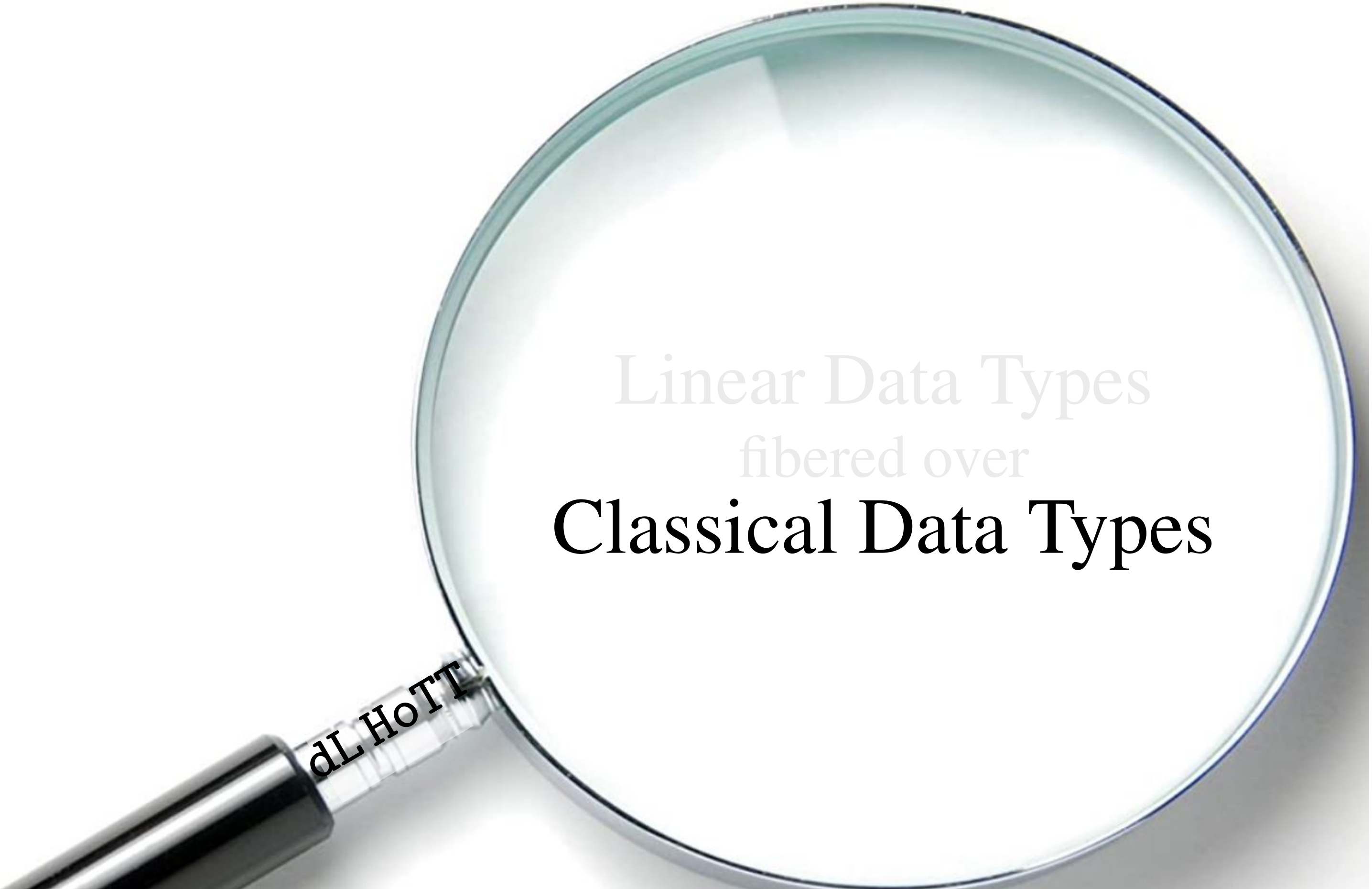
∃ *universal* quantum+classical specification language

conservative over classical *Homotopy Type Theory* (`HoTT`)

and

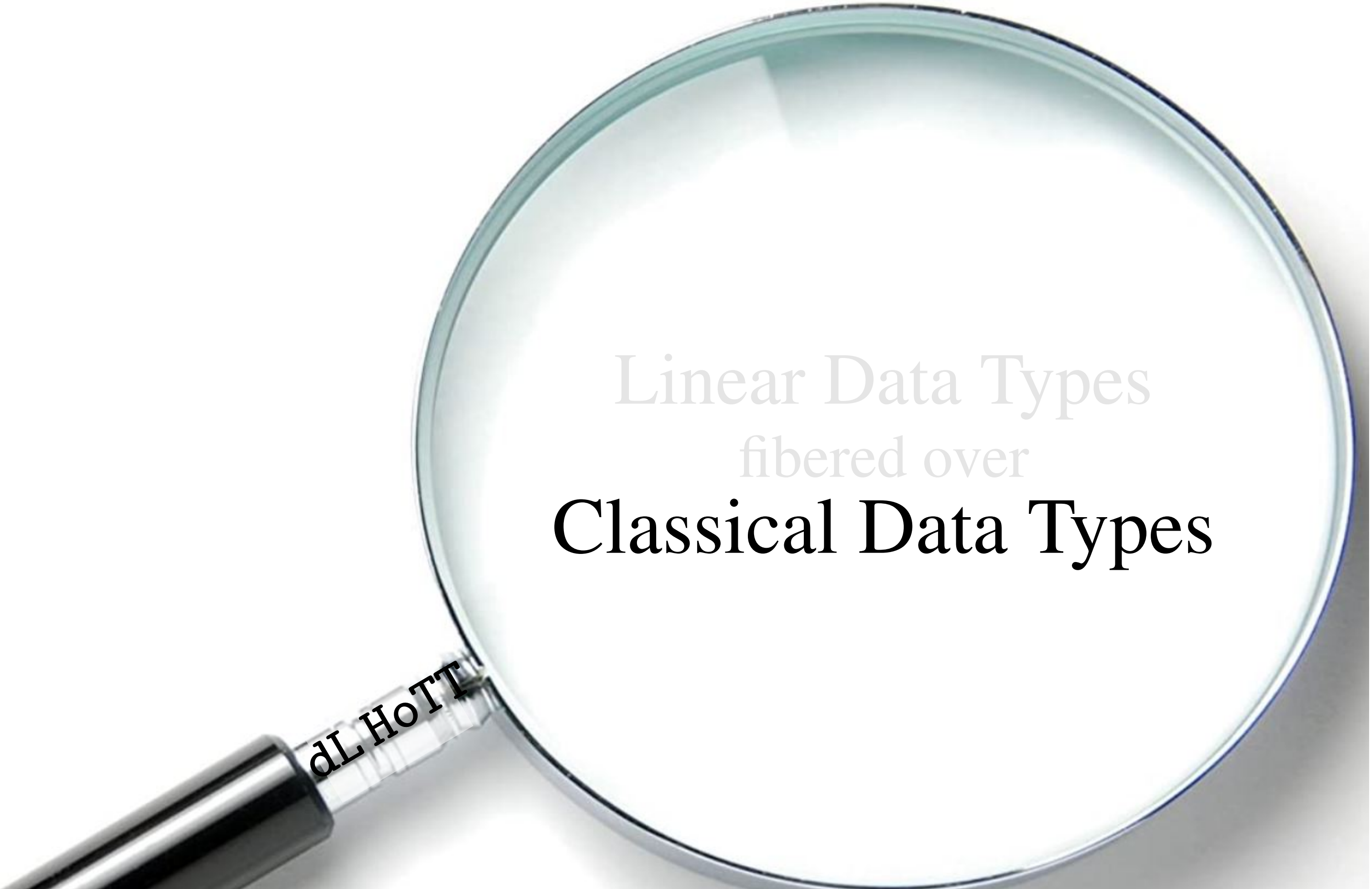
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]

(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



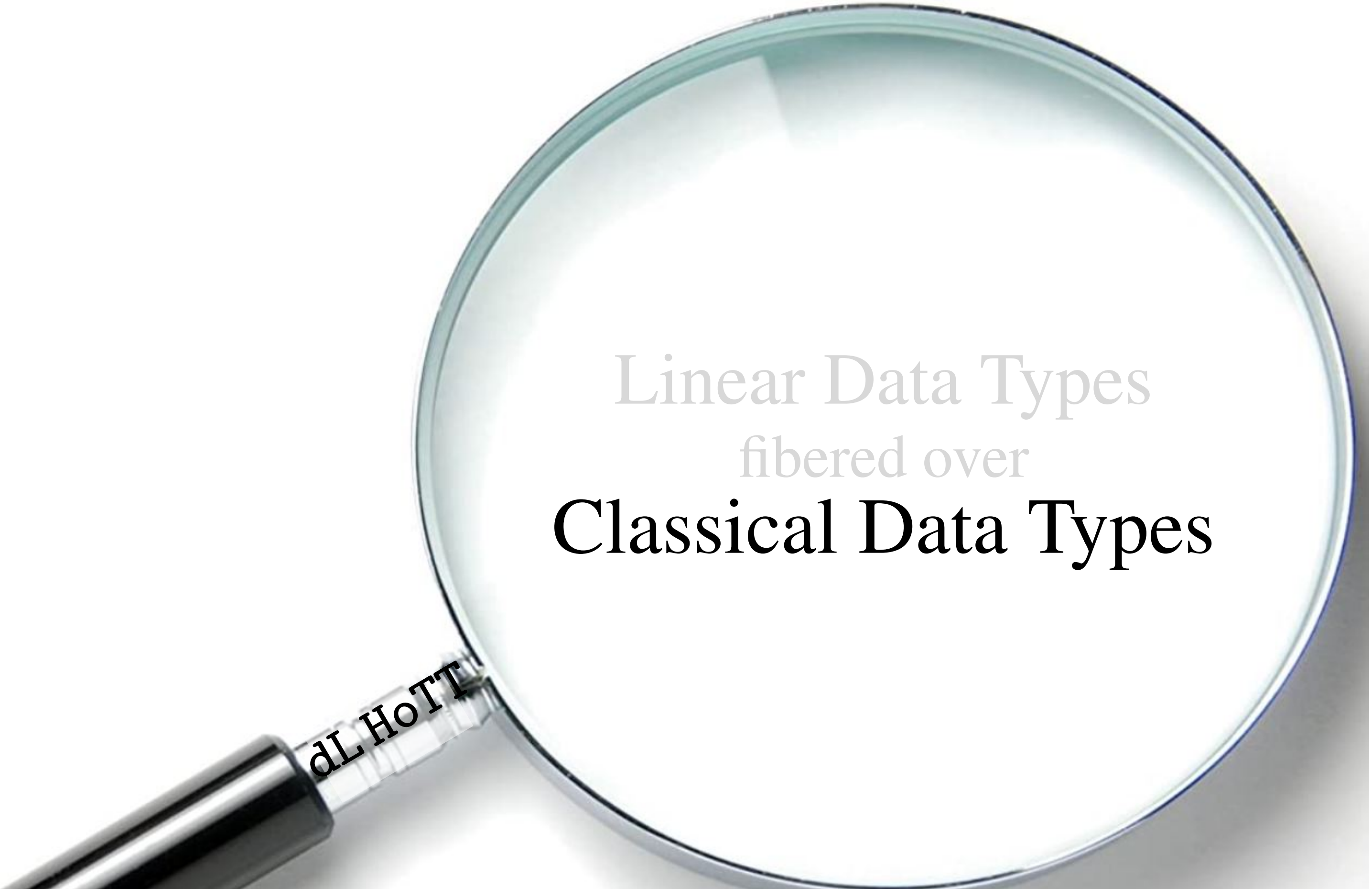
Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme “Motivic Yoga” [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck’s six operations *à la* Wirthmüller — more on all this below)



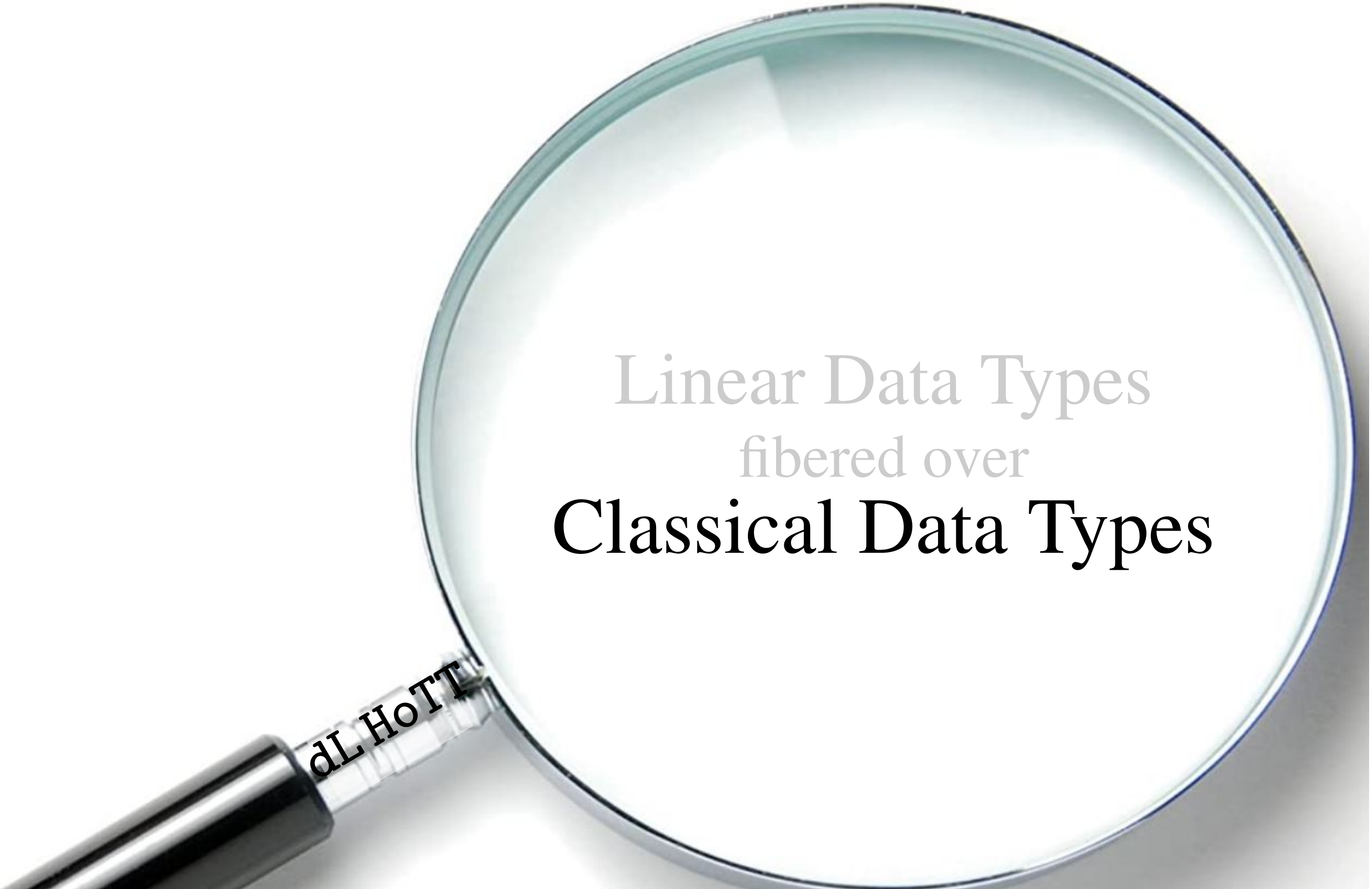
Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

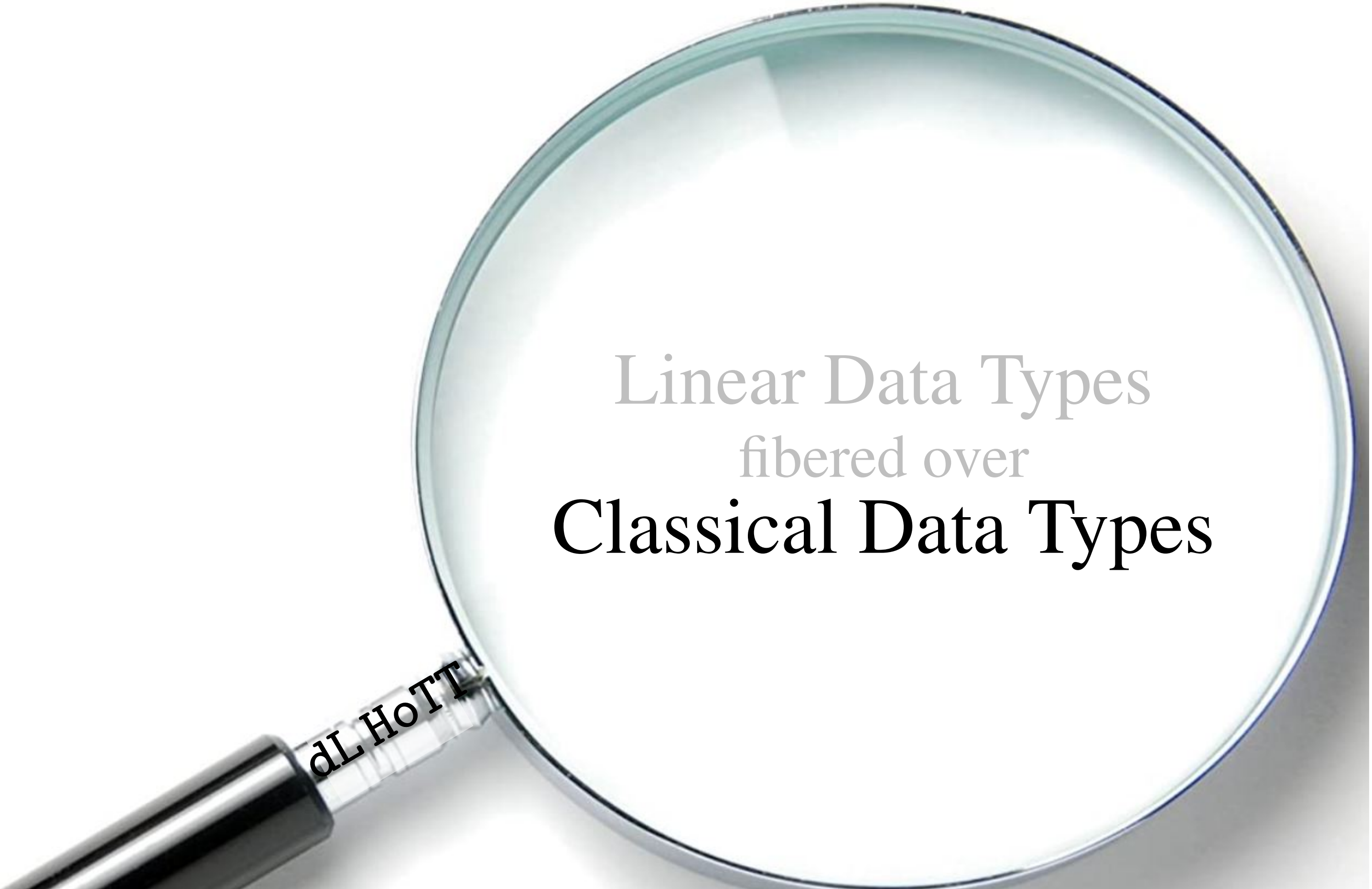
$\exists$ *universal* quantum+classical specification language

conservative over classical *Homotopy Type Theory* (`HoTT`)

and

verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]

(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Linear Data Types

fibered over

Classical Data Types

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

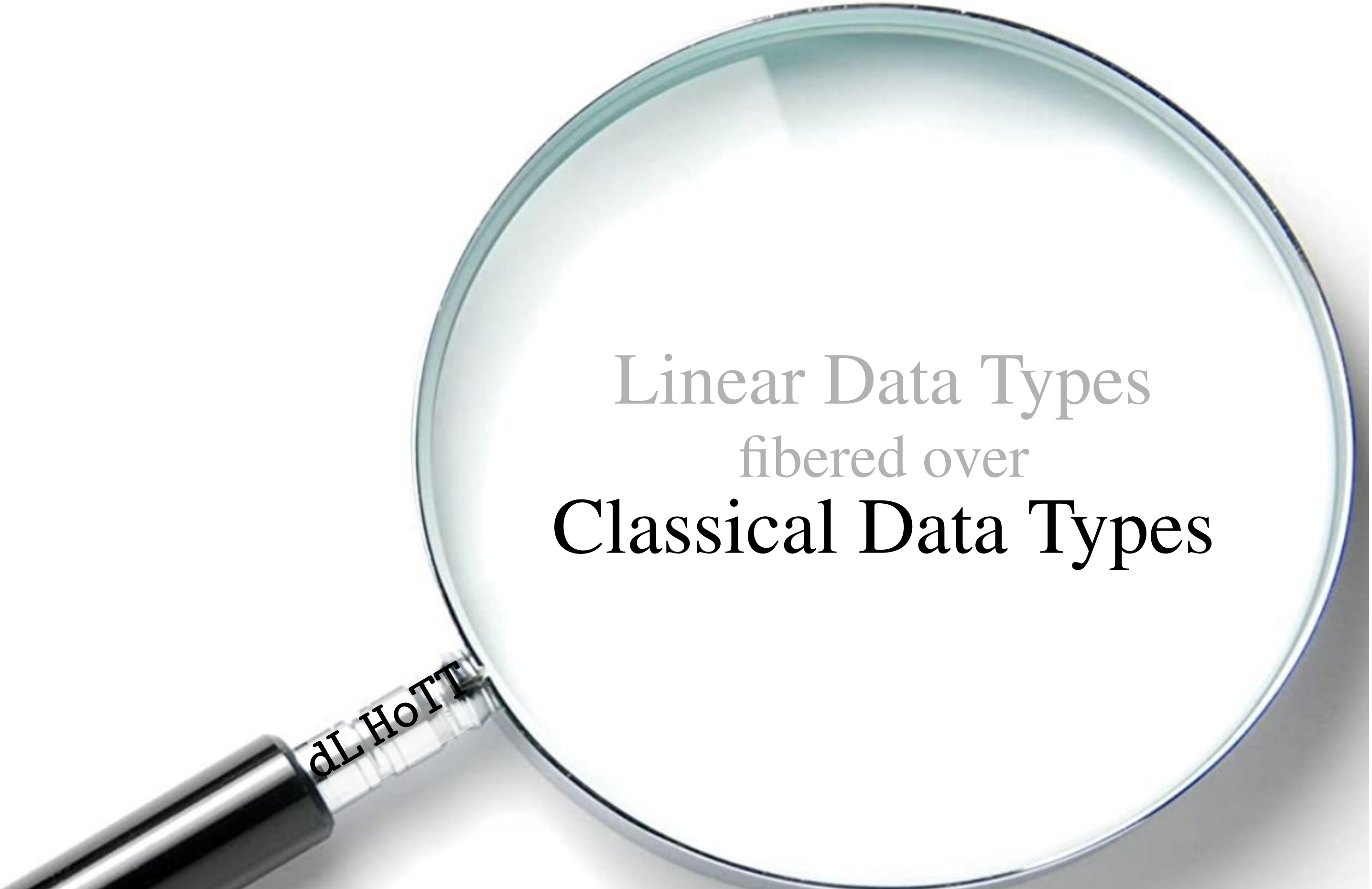
$\exists$ *universal* quantum+classical specification language

conservative over classical *Homotopy Type Theory* (`HoTT`)

and

verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]

(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



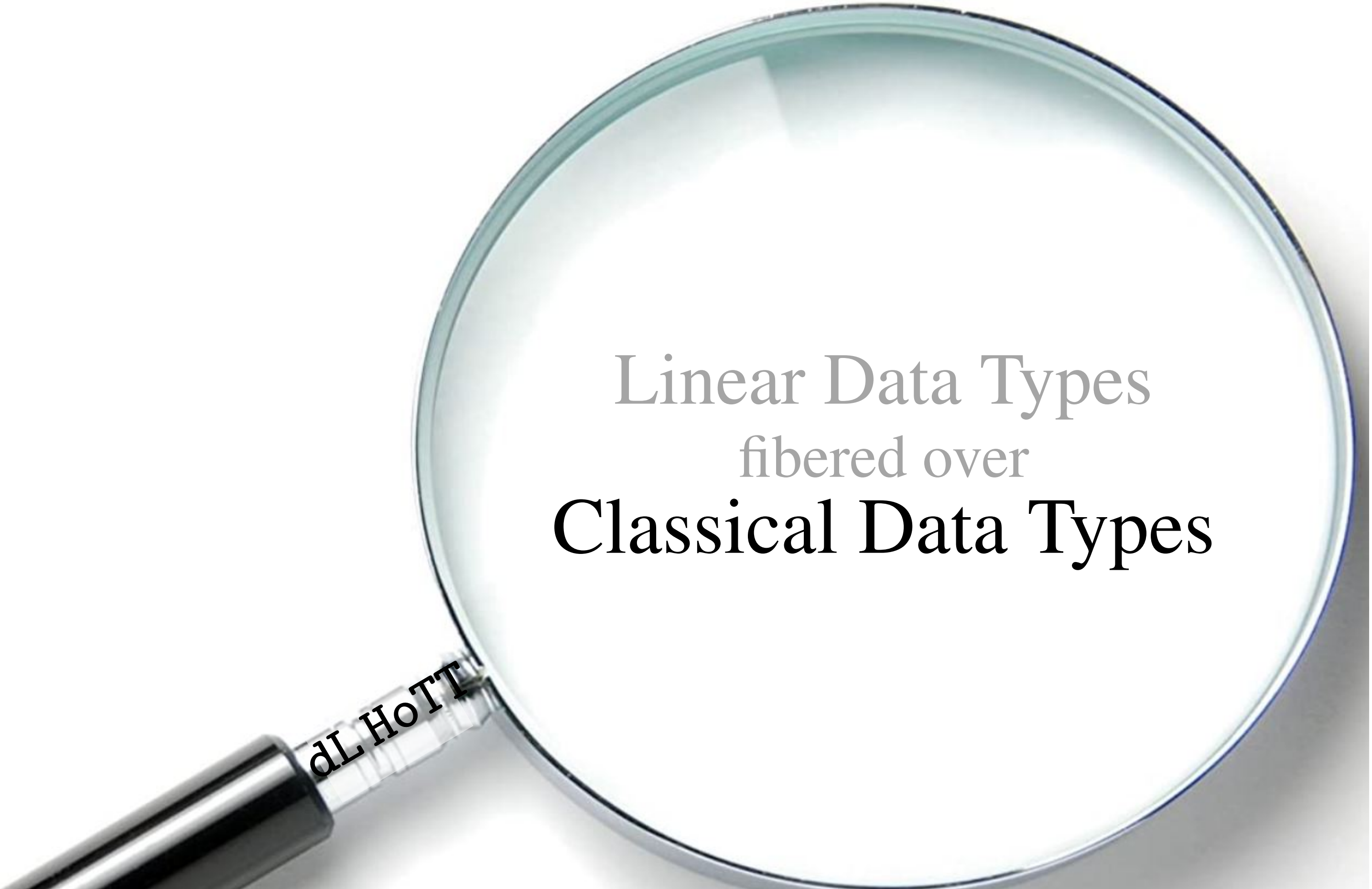
Linear Data Types

fibered over

Classical Data Types

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

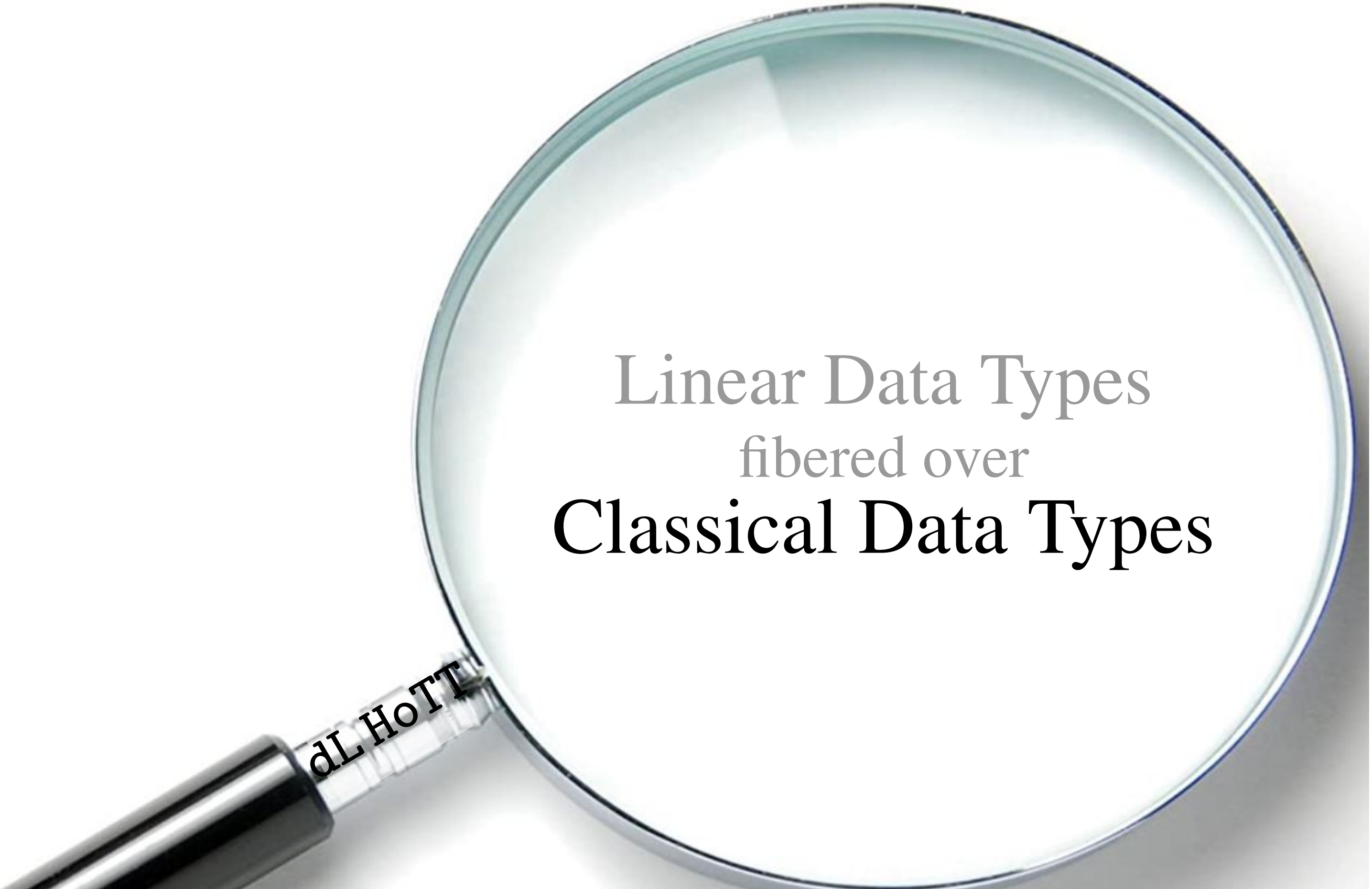
$\exists$ *universal* quantum+classical specification language

conservative over classical *Homotopy Type Theory* (`HoTT`)

and

verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]

(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



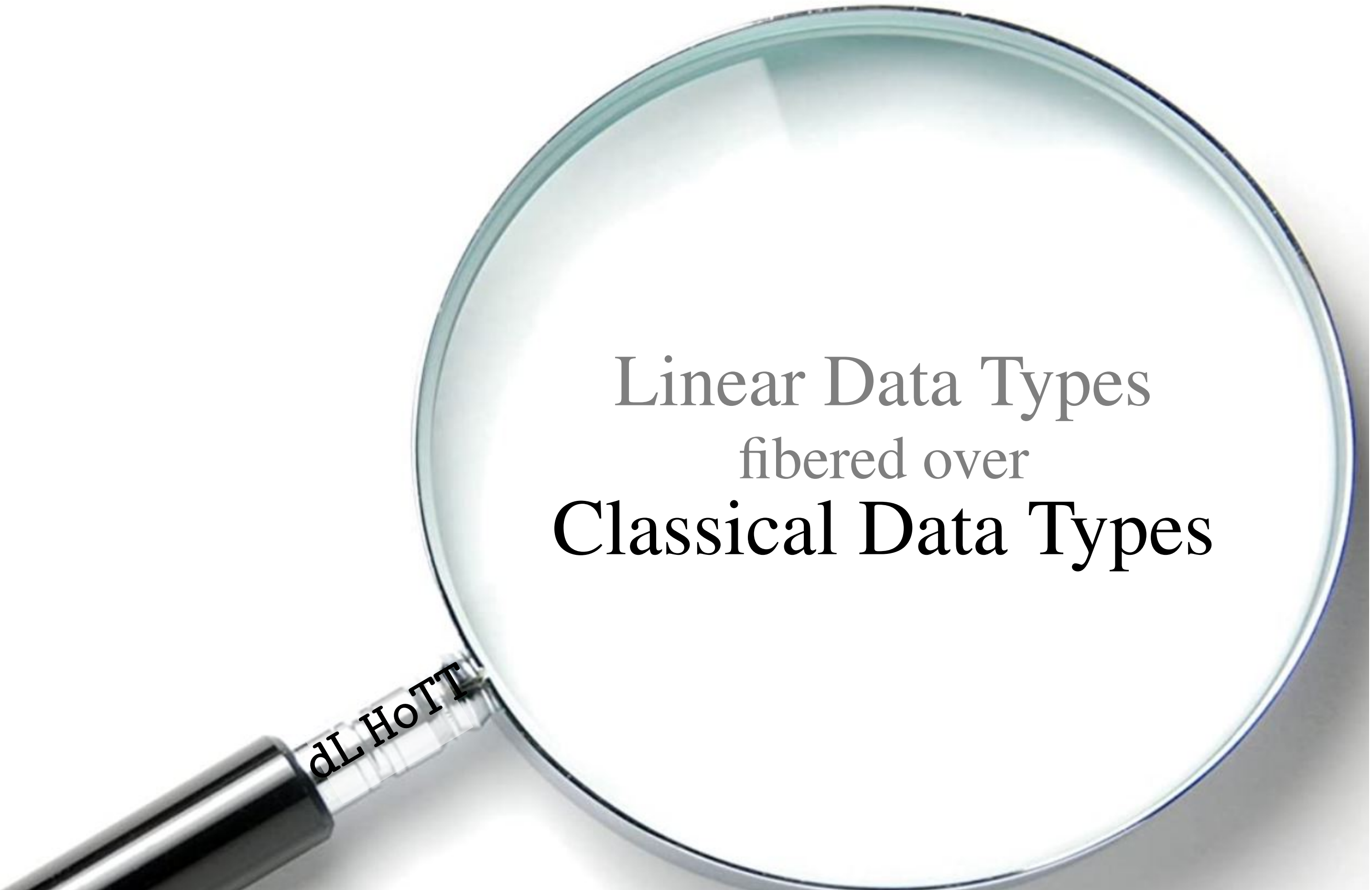
Linear Data Types

fibered over

Classical Data Types

Dependent Linear Homotopy Type Theory (`LHoTT`)

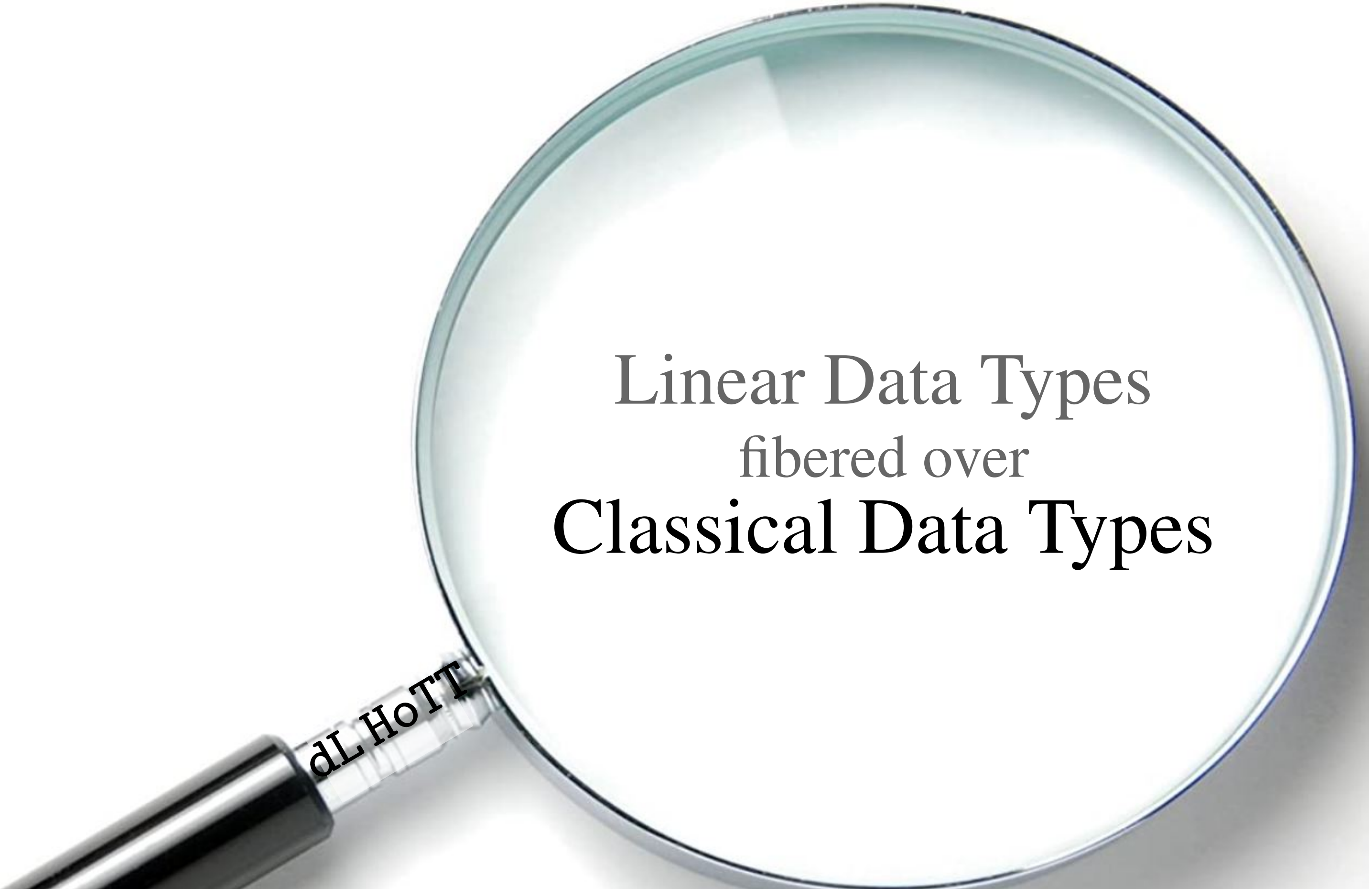
Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)



Linear Data Types
fibered over
Classical Data Types

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

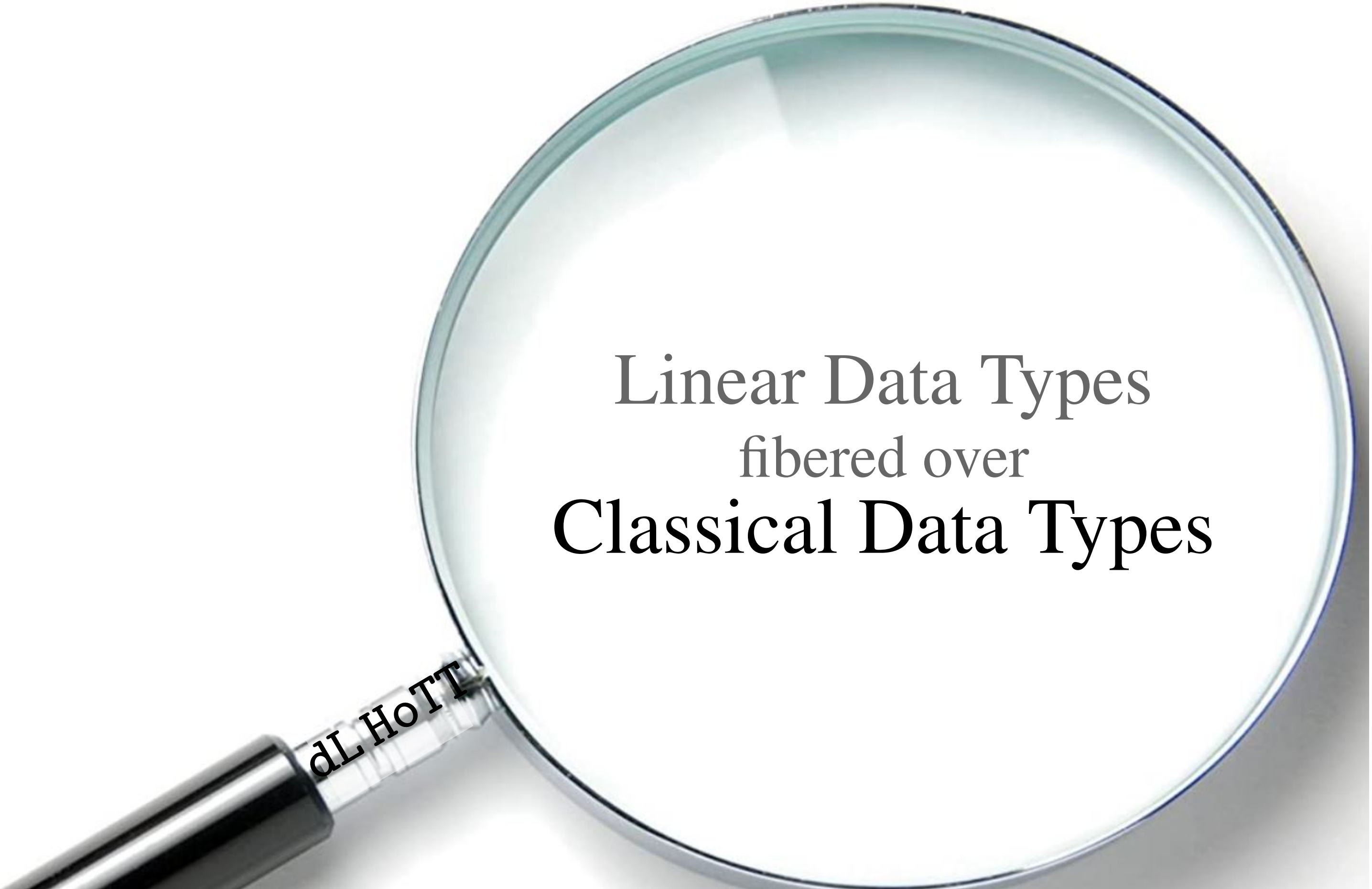


Linear Data Types
fibered over
Classical Data Types

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

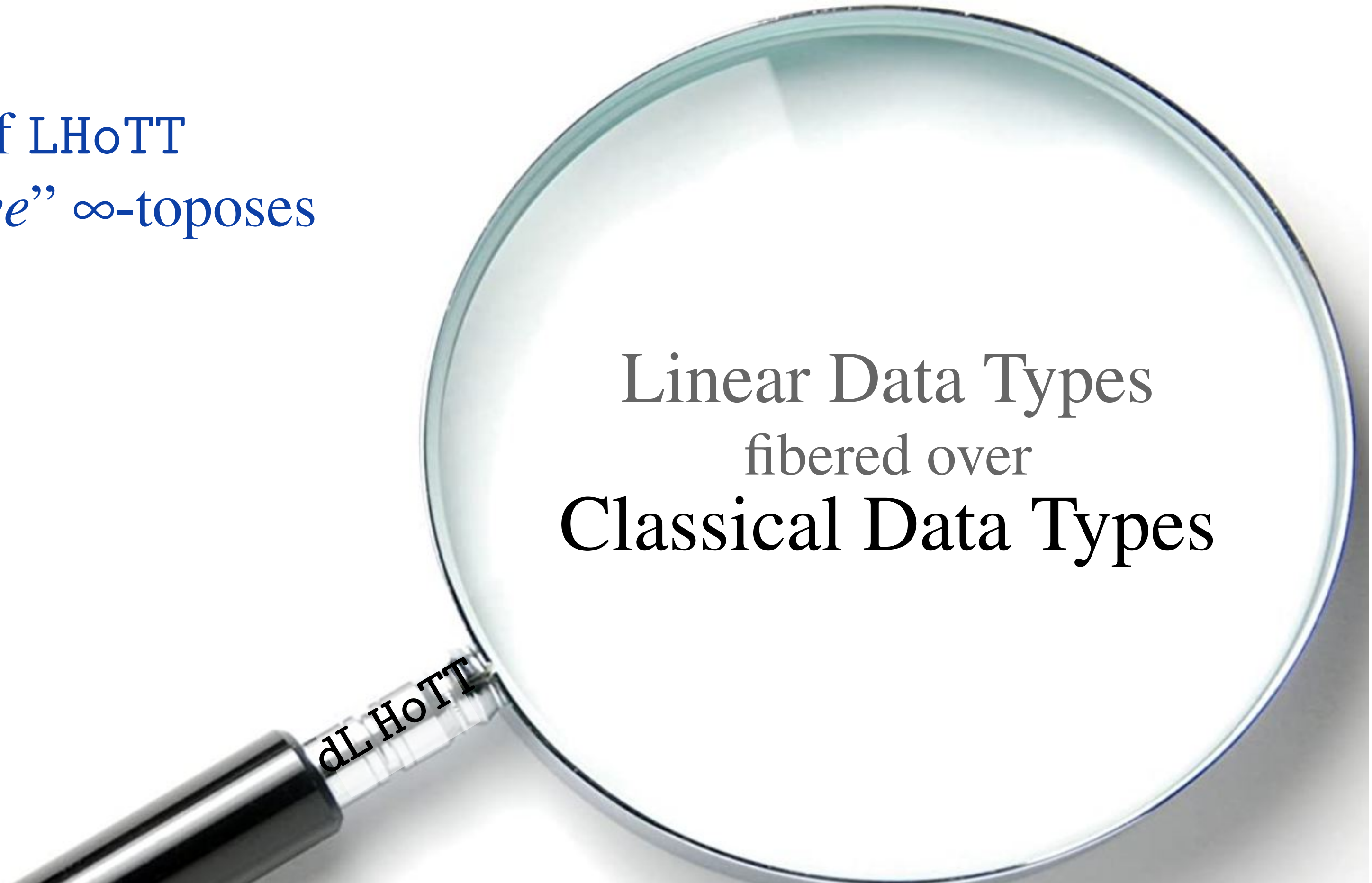
More technically:



Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

More technically:
The categorical semantics of `LHoTT`
is in "*infinitesimally cohesive*" ∞-toposes



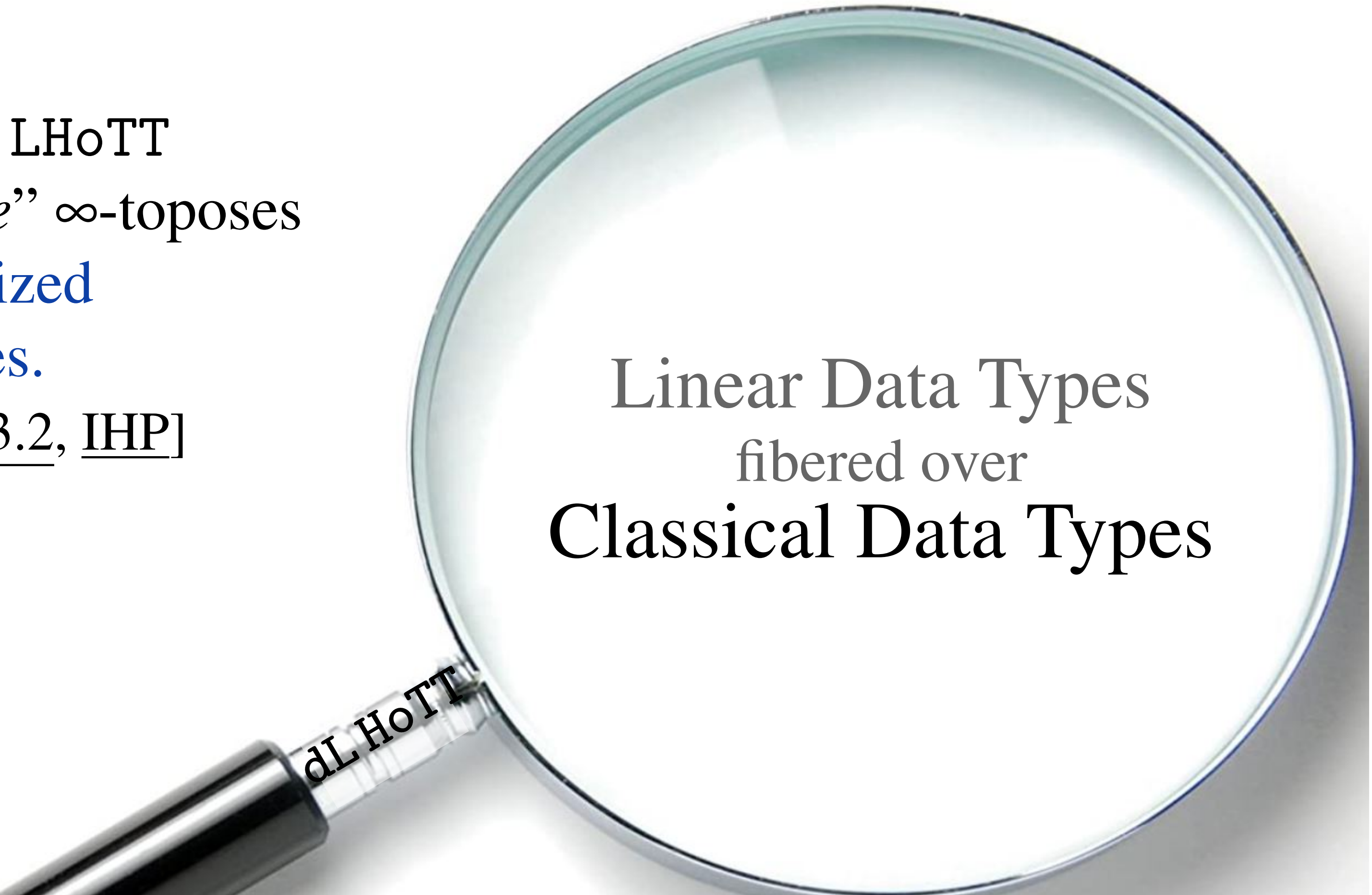
Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

More technically:

The categorical semantics of `LHoTT`
is in "*infinitesimally cohesive*" ∞-toposes
of module spectra parameterized
over classical homotopy types.
[S. (2013), §4.1.2] [S. (2014), §3.2, IHP]
[Riley, Finster & Licata (2021)].



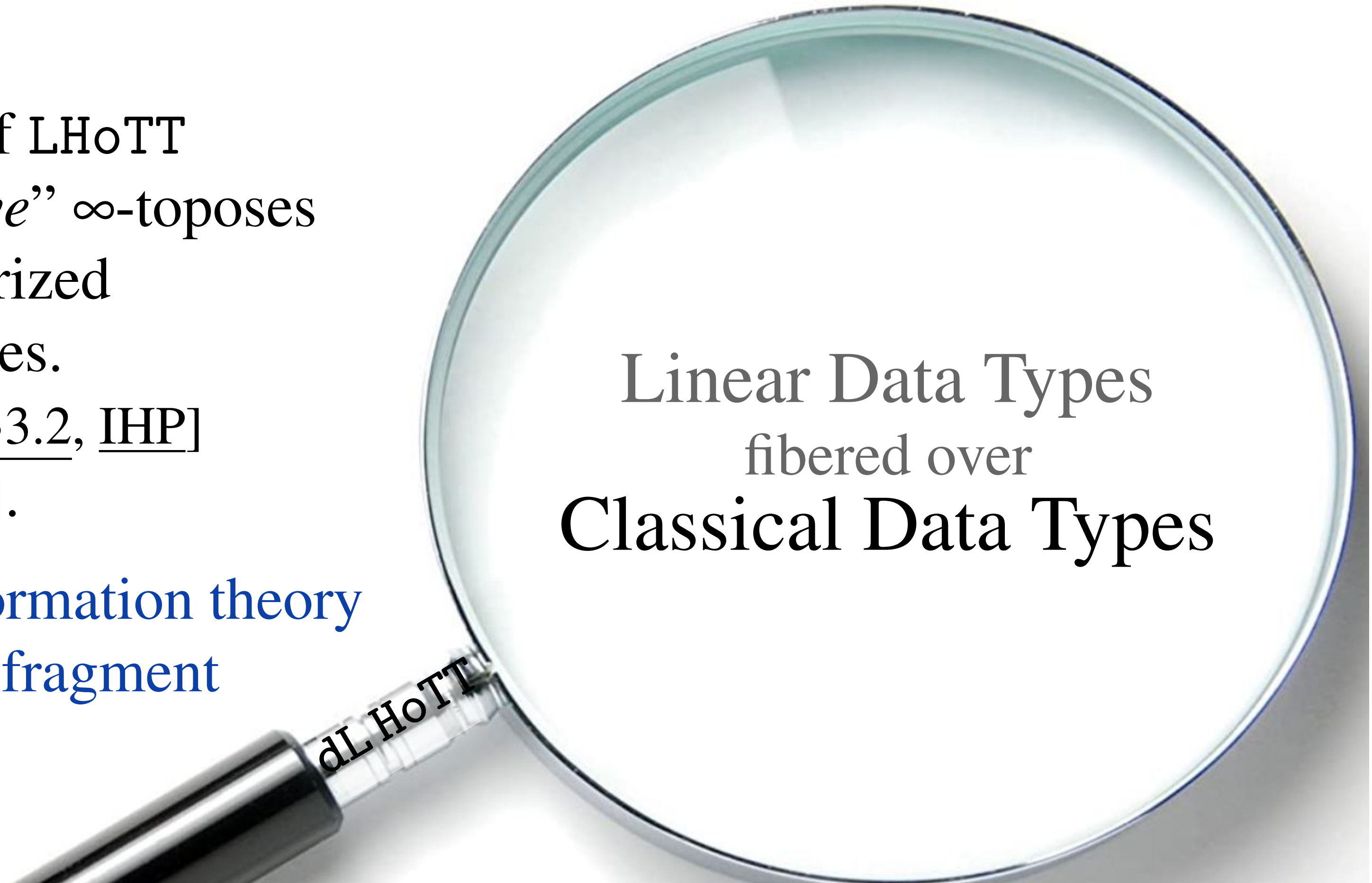
Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

More technically:

The categorical semantics of `LHoTT`
is in "*infinitesimally cohesive*" ∞-toposes
of module spectra parameterized
over classical homotopy types.
[S. (2013), §4.1.2] [S. (2014), §3.2, IHP]
[Riley, Finster & Licata (2021)].

For traditional quantum information theory
this faithfully subsumes the fragment



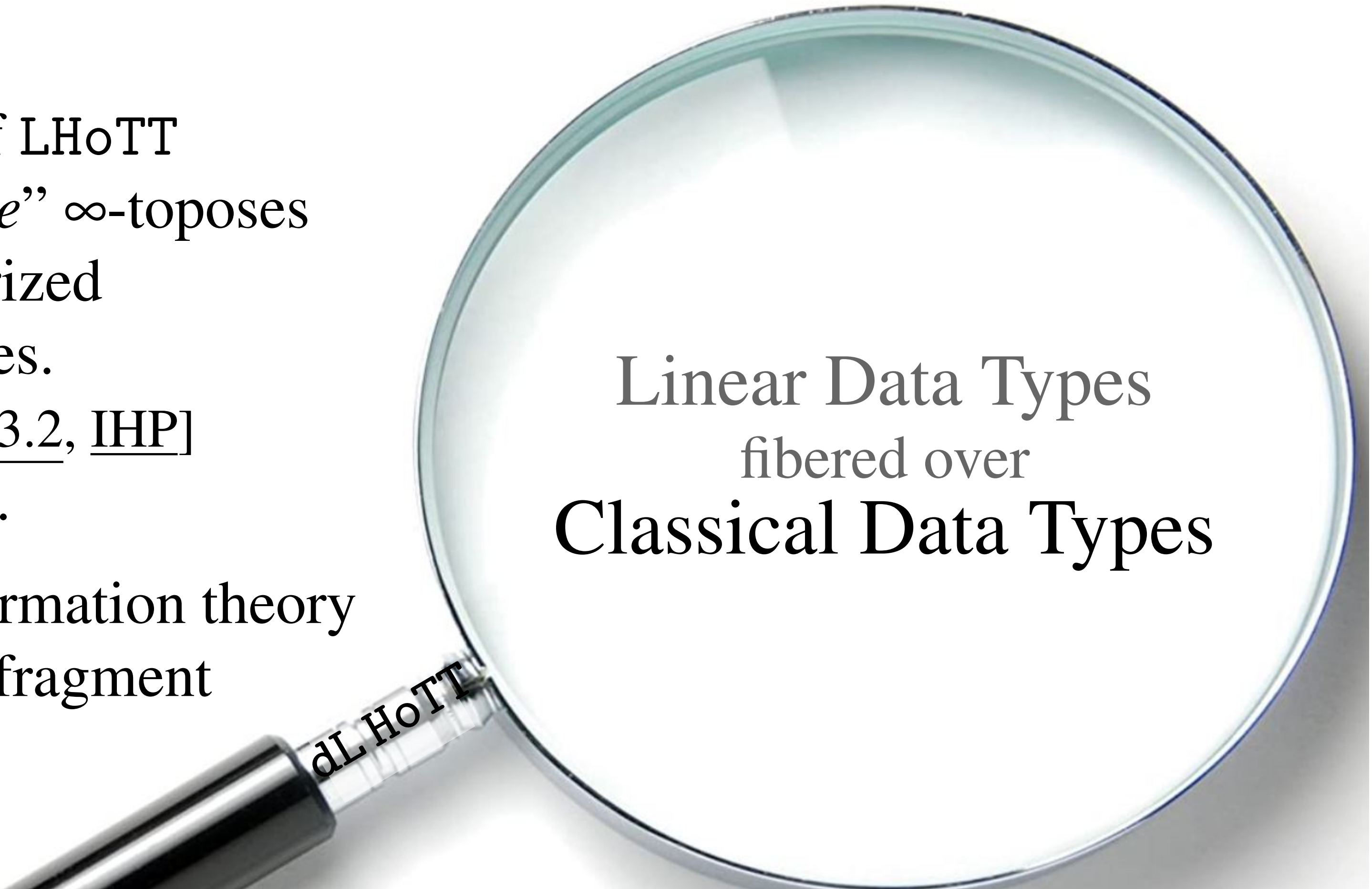
Linear Data Types
fibered over
Classical Data Types

Dependent Linear Homotopy Type Theory (LHoTT)

Theorem [M. Riley (2022), doi:10.14418/wes01.3.139]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (HoTT)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

More technically:
The categorical semantics of LHoTT
is in "*infinitesimally cohesive*" ∞-toposes
of module spectra parameterized
over classical homotopy types.
[S. (2013), §4.1.2] [S. (2014), §3.2, IHP]
[Riley, Finster & Licata (2021)].

For traditional quantum information theory
this faithfully subsumes the fragment
of complex vector bundles
over finite sets.

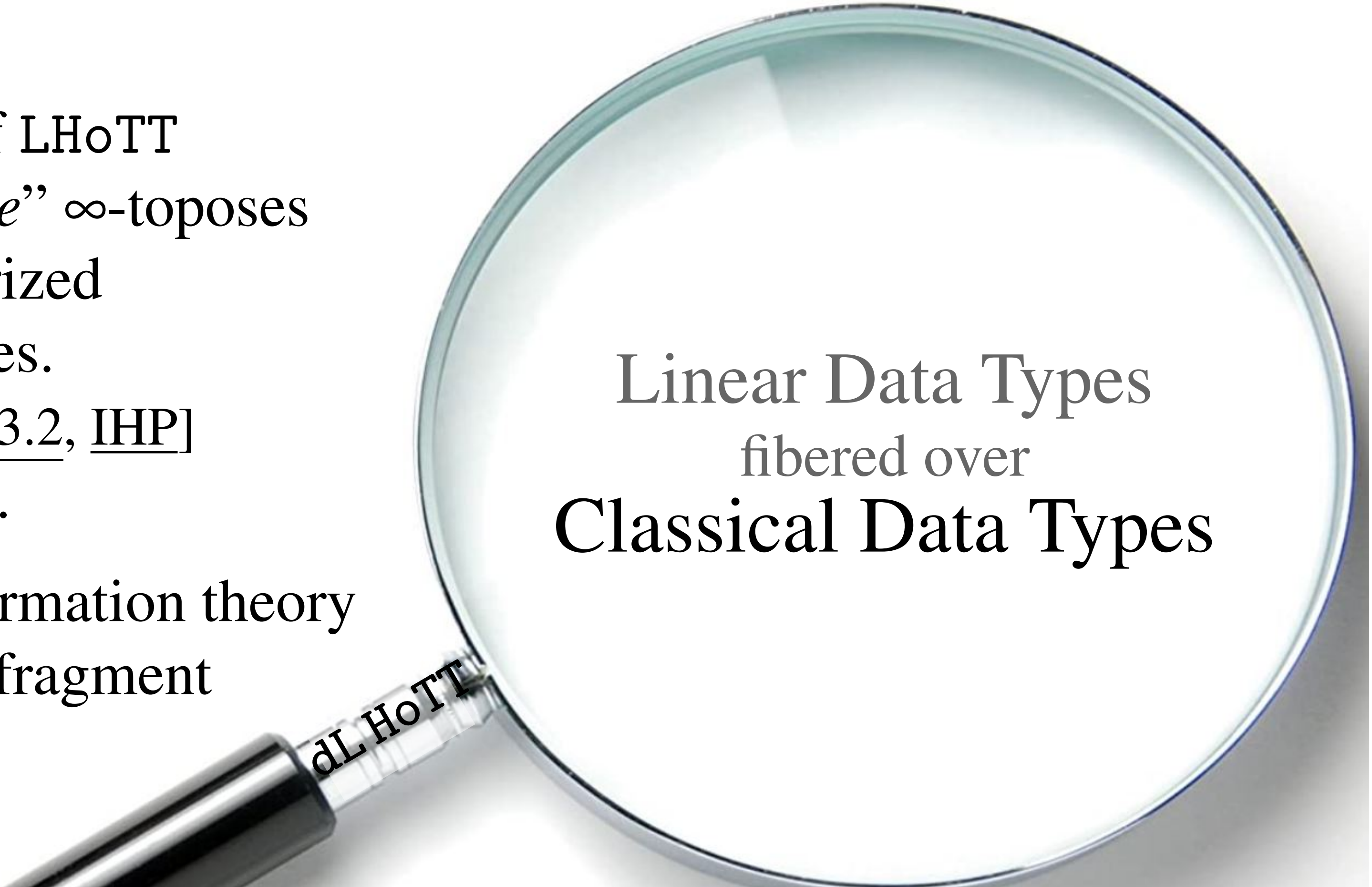

Linear Data Types
fibered over
Classical Data Types

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

More technically:
The categorical semantics of `LHoTT`
is in "*infinitesimally cohesive*" ∞-toposes
of module spectra parameterized
over classical homotopy types.
[S. (2013), §4.1.2] [S. (2014), §3.2, IHP]
[Riley, Finster & Licata (2021)].

For traditional quantum information theory
this faithfully subsumes the fragment
of complex vector bundles
over finite sets.



Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

$\exists$ *universal* quantum+classical specification language

conservative over classical *Homotopy Type Theory* (`HoTT`)

and

verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]

(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Theorem [CQTS (2022)]:

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Theorem [CQTS (2022)]:
Motivic Yoga induces a system of monadic computational effects
constituting
linear modalities of actuality and potentiality

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Theorem [CQTS (2022)]:
Motivic Yoga induces a system of monadic computational effects
constituting
linear modalities of actuality and potentiality
which happen to
know all about quantum information theory:

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Theorem [CQTS (2022)]:
Motivic Yoga induces a system of monadic computational effects
constituting
linear modalities of actuality and potentiality
which happen to
know all about quantum information theory:

quantum measurement is handling of linear indefiniteness effects

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:

∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Theorem [CQTS (2022)]:

Motivic Yoga induces a system of monadic computational effects
constituting
linear modalities of actuality and potentiality
which happen to
know all about quantum information theory:

quantum measurement	is	handling of linear indefiniteness effects
quantum state preparation	is	handling of linear randomness co-effects

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Theorem [CQTS (2022)]:
Motivic Yoga induces a system of monadic computational effects
constituting
linear modalities of actuality and potentiality
which happen to
know all about quantum information theory:

quantum measurement	is	handling of linear indefiniteness effects
quantum state preparation	is	handling of linear randomness co-effects
quantum+classical circuits	are	the effectful string diagrams

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
$\exists$ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Theorem [CQTS (2022)]:
Motivic Yoga induces a system of monadic computational effects
constituting
linear modalities of actuality and potentiality
which happen to
know all about quantum information theory:

quantum measurement	is	handling of linear indefiniteness effects
quantum state preparation	is	handling of linear randomness co-effects
quantum+classical circuits	are	the effectful string diagrams
quantum dynamic lifting	is	comparison functor of monadicity theorem

Dependent Linear Homotopy Type Theory (`LHoTT`)

Theorem [M. Riley (2022), `doi:10.14418/wes01.3.139`]:
∃ *universal* quantum+classical specification language
conservative over classical *Homotopy Type Theory* (`HoTT`)
and
verifying axiom scheme "Motivic Yoga" [Riley, §2.4, anticipated in S. (2014), §3.2]
(i.e. Grothendieck's six operations *à la* Wirthmüller — more on all this below)

Theorem [CQTS (2022)]:
Motivic Yoga induces a system of monadic computational effects
constituting
linear modalities of actuality and potentiality
which happen to
know all about quantum information theory:

quantum measurement	is	handling of linear indefiniteness effects
quantum state preparation	is	handling of linear randomness co-effects
quantum+classical circuits	are	the effectful string diagrams
quantum dynamic lifting	is	comparison functor of monadicity theorem

`QS` – Quantum Systems language @ CQTS

$\rightsquigarrow$ full-blown Quantum Systems language emerges embedded in `LHoTT`

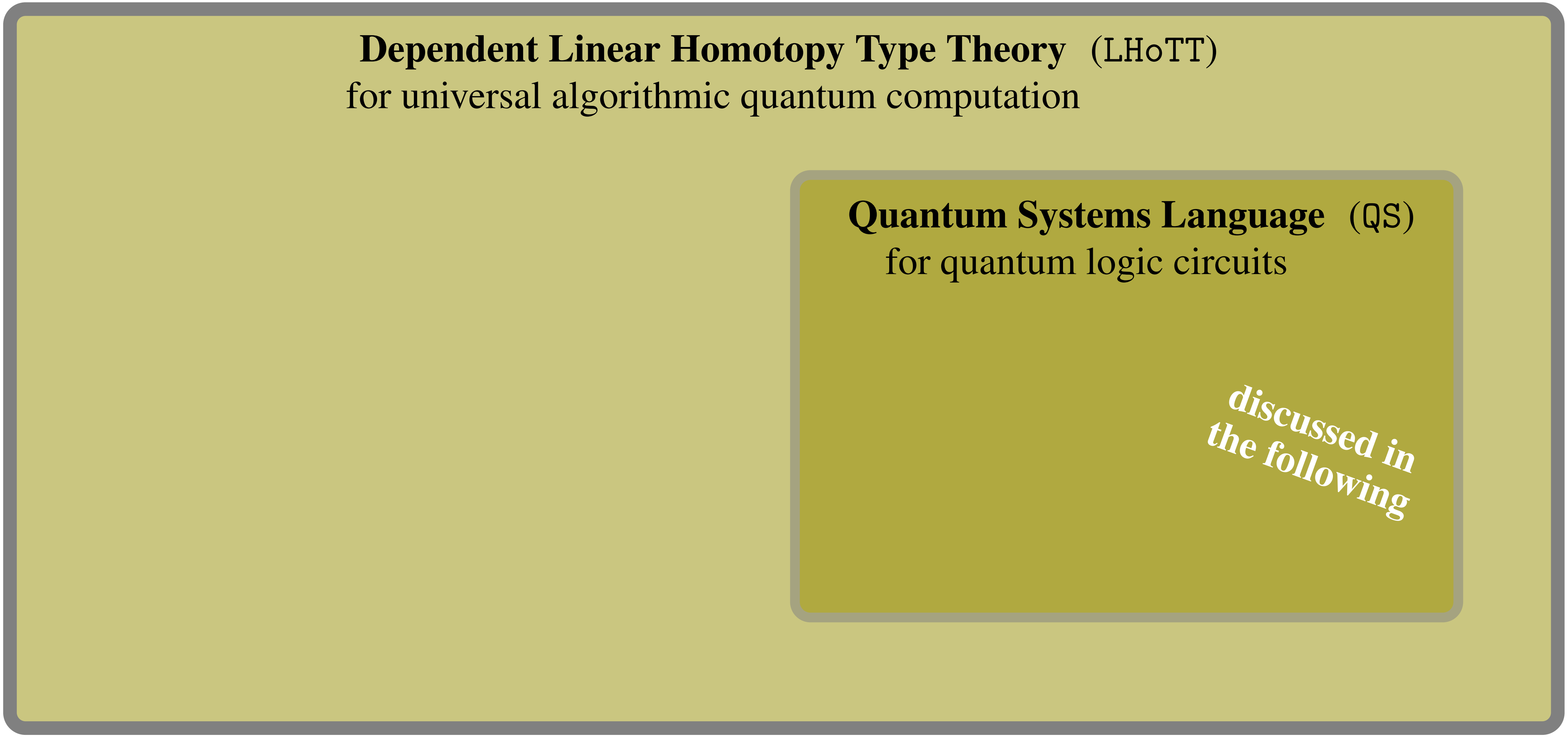
`QS` – Quantum Systems language @ CQTS

$\rightsquigarrow$ full-blown Quantum Systems language emerges embedded in `LHoTT`

Dependent Linear Homotopy Type Theory (`LHoTT`)
for universal algorithmic quantum computation

QS – Quantum Systems language @ CQTS

⇝ full-blown Quantum Systems language emerges embedded in LHoTT



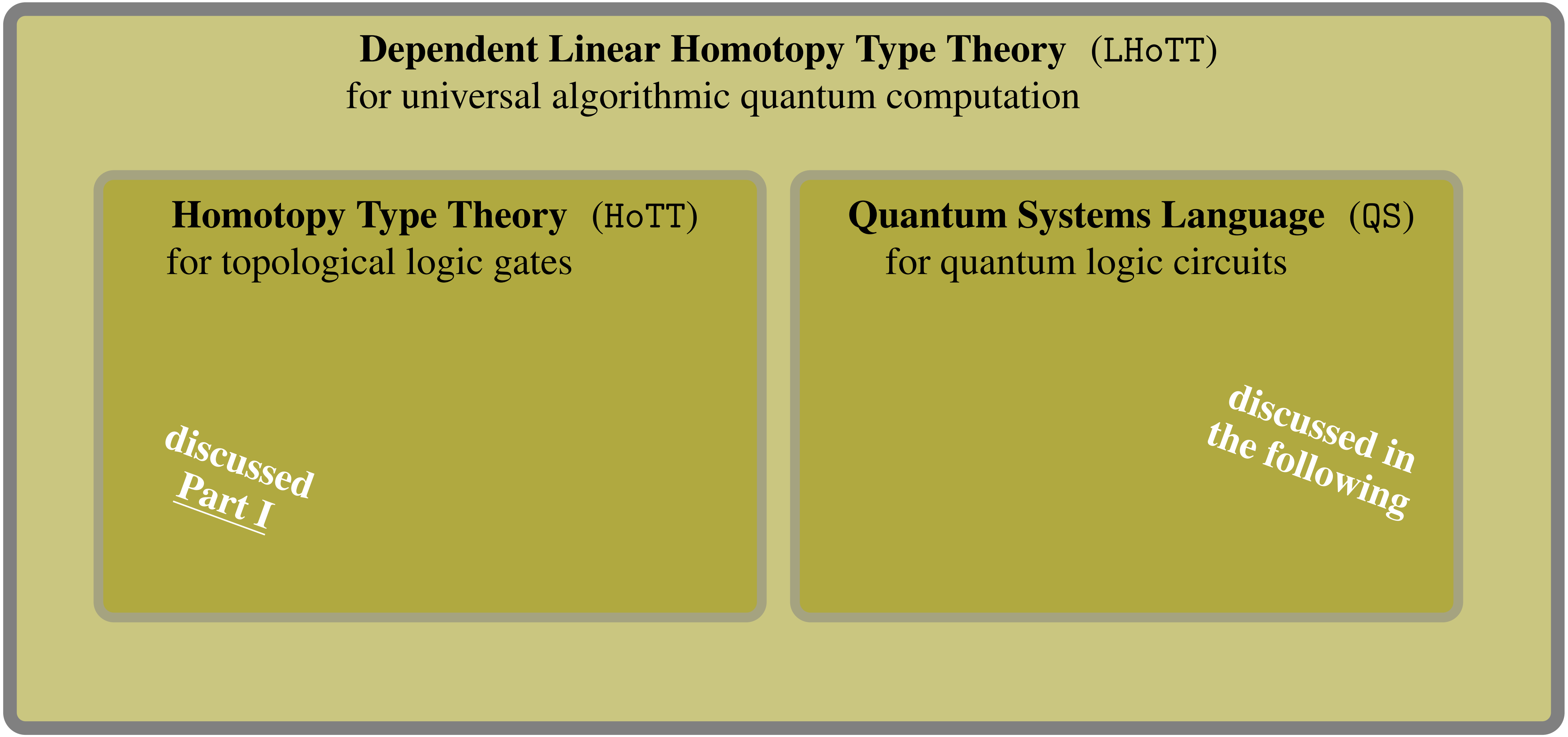
Dependent Linear Homotopy Type Theory (LHoTT)
for universal algorithmic quantum computation

Quantum Systems Language (QS)
for quantum logic circuits

discussed in the following

QS – Quantum Systems language @ CQTS

⇝ full-blown Quantum Systems language emerges embedded in LHoTT



Dependent Linear Homotopy Type Theory (LHoTT)
for universal algorithmic quantum computation

Homotopy Type Theory (HoTT)
for topological logic gates

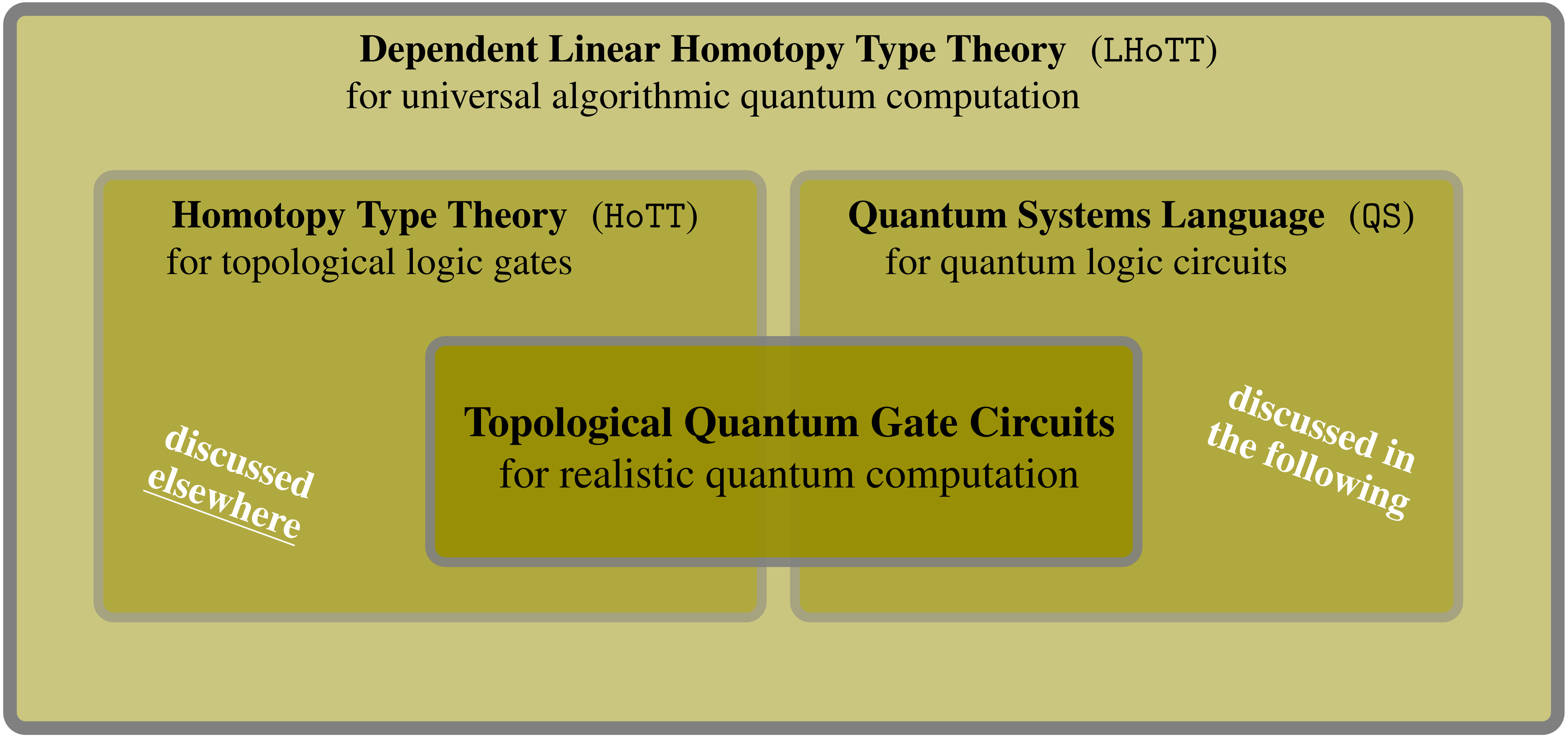
discussed Part I

Quantum Systems Language (QS)
for quantum logic circuits

discussed in the following

`QS` – Quantum Systems language @ CQTS

⇝ full-blown Quantum Systems language emerges embedded in `LHoTT`



Dependent Linear Homotopy Type Theory (`LHoTT`)
for universal algorithmic quantum computation

Homotopy Type Theory (`HoTT`)
for topological logic gates

discussed elsewhere

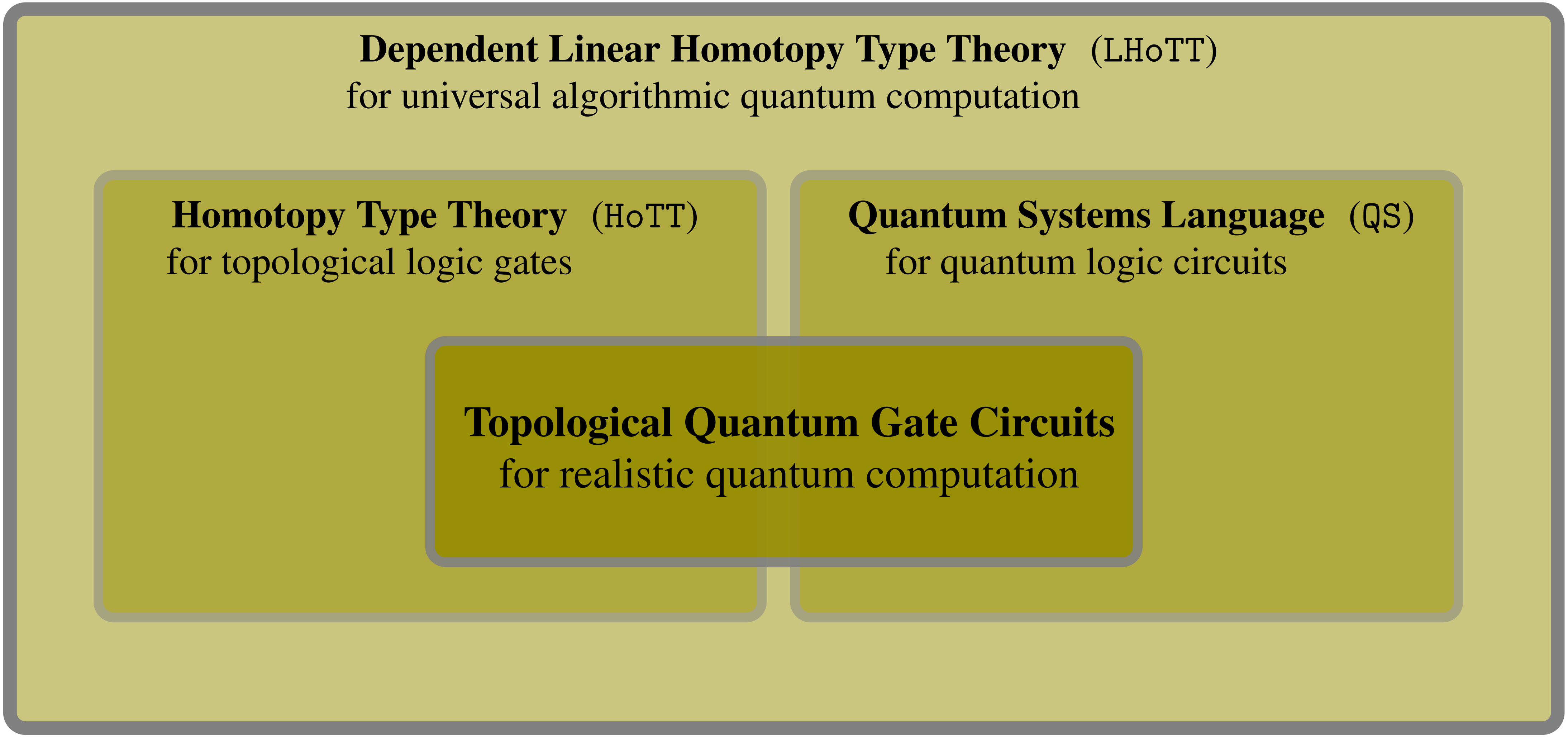
Quantum Systems Language (`QS`)
for quantum logic circuits

discussed in the following

Topological Quantum Gate Circuits
for realistic quantum computation

QS – Quantum Systems language @ CQTS

⇝ full-blown Quantum Systems language emerges embedded in LHoTT



Dependent Linear Homotopy Type Theory (LHoTT)
for universal algorithmic quantum computation

Homotopy Type Theory (HoTT)
for topological logic gates

Quantum Systems Language (QS)
for quantum logic circuits

Topological Quantum Gate Circuits
for realistic quantum computation

ambient LHoTT	verifies	classically dependent quantum linear types
ambient HoTT	provides	specification of topological quantum gates
ambient dTT	provides	full verified classical control

Quantum Data Types

Linear/Quantum Data Types

<table>
<tr><td>Characteristic Property</td><td></td><td></td><td></td></tr>
<tr><td>Symbol</td><td></td><td></td><td></td></tr>
<tr><td>Formula
(for B : FinType)</td><td></td><td></td><td></td></tr>
<tr><td rowspan="2">AlgTop Jargon</td><td rowspan="2"></td><td></td><td></td></tr>
<tr><td colspan="2"></td></tr>
<tr><td>Linear Logic</td><td></td><td></td><td></td></tr>
<tr><td>Physics Meaning</td><td></td><td></td><td></td></tr>
</table>

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:		
Symbol			
Formula (for B : FinType)			
AlgTop Jargon			
Linear Logic			
Physics Meaning			

Linear/Quantum Data Types

<table>
<tr><td>Characteristic Property</td><td>1. their cartesian product blends into the co-product:</td><td></td><td></td></tr>
<tr><td>Symbol</td><td>$\oplus$ direct sum</td><td></td><td></td></tr>
<tr><td>Formula
(for B : FinType)</td><td></td><td></td><td></td></tr>
<tr><td rowspan="2">AlgTop Jargon</td><td rowspan="2"></td><td></td><td></td></tr>
<tr><td colspan="2"></td></tr>
<tr><td>Linear Logic</td><td></td><td></td><td></td></tr>
<tr><td>Physics Meaning</td><td></td><td></td><td></td></tr>
</table>

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	
Symbol	$\oplus$ direct sum		
Formula (for B : FinType)			
AlgTop Jargon			
Linear Logic			
Physics Meaning			

Linear/Quantum Data Types

<table>
<tr><td>Characteristic Property</td><td>1. their cartesian product blends into the co-product:</td><td>2. a tensor product appears & distributes over direct sum</td><td></td></tr>
<tr><td>Symbol</td><td>$\oplus$ direct sum</td><td>$\otimes$ tensor product</td><td></td></tr>
<tr><td>Formula
(for B : FinType)</td><td></td><td></td><td></td></tr>
<tr><td rowspan="2">AlgTop Jargon</td><td rowspan="2"></td><td></td><td></td></tr>
<tr><td colspan="2"></td></tr>
<tr><td>Linear Logic</td><td></td><td></td><td></td></tr>
<tr><td>Physics Meaning</td><td></td><td></td><td></td></tr>
</table>

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	
Formula (for B : FinType)			
AlgTop Jargon			
Linear Logic			
Physics Meaning			

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)			
AlgTop Jargon			
Linear Logic			
Physics Meaning			

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\textbf{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\textbf{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\textbf{co-product}}{\coprod_B \mathcal{H}_b}$		
AlgTop Jargon			
Linear Logic			
Physics Meaning			

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\textbf{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\textbf{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\textbf{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	
AlgTop Jargon			
Linear Logic			
Physics Meaning			

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$
AlgTop Jargon			
Linear Logic			
Physics Meaning			

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\bigoplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\textbf{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\textbf{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\textbf{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$
AlgTop Jargon	biproduct, stability, ambidexterity	Frobenius reciprocity	mapping spectrum
Linear Logic			
Physics Meaning			

Linear/Quantum Data Types

<table>
<tr><th>Characteristic Property</th><th>1. their cartesian product blends into the co-product:</th><th>2. a tensor product appears & distributes over direct sum</th><th>3. a linear function type appears adjoint to tensor</th></tr>
<tr><td>Symbol</td><td>$\bigoplus$ direct sum</td><td>$\bigotimes$ tensor product</td><td>$\multimap$ linear function type</td></tr>
<tr><td>Formula
(for B : FinType)</td><td>$\overset{\textbf{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\textbf{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\textbf{co-product}}{\coprod_B \mathcal{H}_b}$</td><td>$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} \big(\mathcal{V} \otimes \mathcal{H}_b \big)$</td><td>$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$
$\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$</td></tr>
<tr><td rowspan="2">AlgTop Jargon</td><td rowspan="2">biproduct,
stability, ambidexterity</td><td>Frobenius reciprocity</td><td>mapping spectrum</td></tr>
<tr><td colspan="2">Grothendieck's Motivic Yoga of 6 oper. (Wirthmüller form)</td></tr>
<tr><td>Linear Logic</td><td></td><td></td><td></td></tr>
<tr><td>Physics Meaning</td><td></td><td></td><td></td></tr>
</table>

Linear/Quantum Data Types

<table>
<tr><th>Characteristic Property</th><th>1. their cartesian product blends into the co-product:</th><th>2. a tensor product appears & distributes over direct sum</th><th>3. a linear function type appears adjoint to tensor</th></tr>
<tr><td>Symbol</td><td>$\oplus$ direct sum</td><td>$\otimes$ tensor product</td><td>$\multimap$ linear function type</td></tr>
<tr><td>Formula
(for B : FinType)</td><td>$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$</td><td>$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$</td><td>$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$
$\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$</td></tr>
<tr><td rowspan="2">AlgTop Jargon</td><td rowspan="2">biproduct,
stability, ambidexterity</td><td>Frobenius reciprocity</td><td>mapping spectrum</td></tr>
<tr><td colspan="2">Grothendieck's Motivic Yoga of 6 oper. (Wirthmüller form)</td></tr>
<tr><td>Linear Logic</td><td>additive disjunction</td><td>multiplicative conjunction</td><td>linear implication</td></tr>
<tr><td>Physics Meaning</td><td></td><td></td><td></td></tr>
</table>

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$
AlgTop Jargon	biproduct, stability, ambidexterity	Frobenius reciprocity Grothendieck's Motivic Yoga of 6 oper. (Wirthmüller form)	mapping spectrum
Linear Logic	additive disjunction	multiplicative conjunction	linear implication
Physics Meaning	superselection sectors / quantum parallelism	compound quantum systems / quantum entanglement	QRAM systems

Linear/Quantum Data Types

<table>
<tr><th>Characteristic Property</th><th>**1.** their cartesian product blends into the co-product:</th><th>**2.** a tensor product appears & distributes over direct sum</th><th>**3.** a linear function type appears adjoint to tensor</th></tr>
<tr><td>**Symbol**</td><td>$\bigoplus$ direct sum</td><td>$\bigotimes$ tensor product</td><td>$\multimap$ linear function type</td></tr>
<tr><td>**Formula**
(for B : FinType)</td><td>$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$</td><td>$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$</td><td>$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$
$\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$</td></tr>
<tr><td rowspan="2">**AlgTop Jargon**</td><td rowspan="2">biproduct,
stability, ambidexterity</td><td>Frobenius reciprocity</td><td>mapping spectrum</td></tr>
<tr><td colspan="2">Grothendieck's Motivic Yoga of 6 oper. (Wirthmüller form)</td></tr>
<tr><td>**Linear Logic**</td><td>additive disjunction</td><td>multiplicative conjunction</td><td>linear implication</td></tr>
<tr><td>**Physics Meaning**</td><td>superselection sectors
/ quantum parallelism</td><td>compound quantum systems
/ quantum entanglement</td><td>QRAM systems</td></tr>
</table>

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd}
B \arrow[r, "p_B"] & * \\
\mathrm{BType}_B & \mathrm{BType}
\end{tikzcd}$$

classical type system dependent on context

classical type system

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	**cart. product** **co-product** $\prod_B \mathcal{H}_b \simeq \bigoplus_B \mathcal{H}_b \simeq \coprod_B \mathcal{H}_b$ **direct sum**	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd}
B \arrow[r, "p_B"] & *
\end{tikzcd}$$

classical context extension

classical type system dependent on context

$$\begin{tikzcd}
\mathrm{BType}_B & \mathrm{BType} \arrow[l, "*_B \times"']
\end{tikzcd}$$

classical type system

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd} B \arrow[r, "p_B"] & * \end{tikzcd}$$

co-product

$$\begin{tikzcd} \mathrm{BType}_B \arrow[r, bend left, "\coprod_{b:B}"] \arrow[r, phantom, "\perp"] & \mathrm{BType} \arrow[l, bend left, "*_B \times"] \end{tikzcd}$$

classical type system dependent on context

classical type system

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\bigoplus$ direct sum	$\bigotimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd}
B \arrow[r, "p_B"] & *
\end{tikzcd}$$

classical type system dependent on context

$$\begin{tikzcd}[column sep=huge]
\mathrm{BType}_B \arrow[r, bend left=50, "\overset{\text{co-product}}{\coprod_{b:B}}"] \arrow[r, bend right=50, "\underset{\text{product}}{\prod_{b:B}}"'] \arrow[r, phantom, shift left=3ex, "\perp"] \arrow[r, phantom, shift right=3ex, "\perp"] & \mathrm{BType} \arrow[l, "*_B \times"']
\end{tikzcd}$$

classical type system

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\bigoplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd}[column sep=huge]
B \arrow[r, "p_B"] & *
\end{tikzcd}$$

co-product

classical type system dependent on context

$$\begin{tikzcd}[column sep=huge, row sep=large]
\mathrm{BType}_B \arrow[r, shift left=6, "\coprod_{b:B}"] \arrow[r, phantom, shift left=3, "\perp"] \arrow[r, phantom, shift right=3, "\perp"] \arrow[r, shift right=6, "\prod_{b:B}"'] & \mathrm{BType} \arrow[l, "*_B \times"']
\end{tikzcd}$$

classical type system

product

classical base change / classical quantification

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	**cart. product** **co-product** $\prod_B \mathcal{H}_b \simeq \bigoplus_B \mathcal{H}_b \simeq \coprod_B \mathcal{H}_b$ **direct sum**	$\mathcal{V} \otimes \Big(\bigoplus\limits_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus\limits_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd}
B \arrow[r, "p_B"] & \ast
\end{tikzcd}$$

classical type system dependent on context

co-product

$$\begin{tikzcd}[column sep=huge]
\mathrm{BType}_B \arrow[r, shift left=8, "\coprod_{b:B}"] \arrow[r, shift right=8, "\prod_{b:B}"'] \arrow[r, phantom, shift left=4, "\perp"] \arrow[r, phantom, shift right=4, "\perp"] & \mathrm{BType} \arrow[l, "\ast_B \times"']
\end{tikzcd}$$

product

classical type system

classical base change / classical quantification

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\bigoplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd}
B \arrow[rr, "p_B"] && *
\end{tikzcd}$$

classical type system dependent on context

$$\begin{tikzcd}
\mathrm{BType}_B \arrow[rr, shift left=8, "\overset{\text{co-product}}{\coprod_{b:B}}"] \arrow[rr, phantom, shift left=4, "\perp"] \arrow[rr, phantom, shift right=4, "\perp"] \arrow[rr, shift right=8, "\underset{\text{product}}{\prod_{b:B}}"'] && \mathrm{BType} \arrow[ll, "{*_B \times}"']
\end{tikzcd}$$

classical type system

classical base change / classical quantification

linear type system in classical context

$$\Big(\mathrm{LType}_B, \overset{\text{tensor}}{\otimes_B}\Big) \qquad \Big(\mathrm{LType}, \overset{\text{tensor}}{\otimes}\Big)$$

linear type system

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\bigoplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \; \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd} B \arrow[r, "p_B"] & * \end{tikzcd}$$

classical type system dependent on context

$$\begin{tikzcd}[column sep=huge]
\mathrm{BType}_B \arrow[r, bend left=40, "\overset{\text{co-product}}{\coprod_{b:B}}"] \arrow[r, phantom, shift left=3, "\perp"] \arrow[r, phantom, shift right=3, "\perp"] \arrow[r, bend right=40, "\underset{\text{product}}{\prod_{b:B}}"'] & \mathrm{BType} \arrow[l, "*_B \times"']
\end{tikzcd}$$

classical type system

linear type system in classical context

$$\begin{tikzcd}[column sep=huge]
\Big(\mathrm{LType}_B, \overset{\text{tensor}}{\otimes_B}\Big) & \Big(\mathrm{LType}, \overset{\text{tensor}}{\otimes}\Big) \arrow[l, "\overset{\text{linear context extension}}{\mathbb{1}_B \otimes}"']
\end{tikzcd}$$

linear type system

classical base change / classical quantification

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd}[column sep=huge]
B \arrow[r, "p_B"] & *
\end{tikzcd}$$

classical type system dependent on context

$$\begin{tikzcd}[column sep=huge, row sep=large]
\mathrm{BType}_B \arrow[r, shift left=8, "\overset{\text{co-product}}{\coprod_{b:B}}"] \arrow[r, phantom, shift left=4, "\perp"] \arrow[r, phantom, shift right=4, "\perp"] \arrow[r, shift right=8, "\underset{\text{product}}{\prod_{b:B}}"'] & \mathrm{BType} \arrow[l, "*_B \times"{description}]
\end{tikzcd}$$

classical type system

classical base change / classical quantification

linear type system in classical context

$$\begin{tikzcd}[column sep=huge, row sep=large]
\Big(\mathrm{LType}_B, \overset{\text{tensor}}{\otimes_B}\Big) \arrow[r, shift left=8, "\overset{\text{direct sum}}{\bigoplus_{b:B}}"] \arrow[r, phantom, shift left=4, "\perp"] \arrow[r, phantom, shift right=4, "\perp"] \arrow[r, shift right=8, "\bigoplus_{b:B}"'] & \Big(\mathrm{LType}, \overset{\text{tensor}}{\otimes}\Big) \arrow[l, "\mathbb{1}_B \otimes"{description}]
\end{tikzcd}$$

linear type system

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\textbf{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\textbf{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\textbf{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...) **reference context**

$$\begin{tikzcd}[column sep=huge] B \arrow[r, "p_B"] & * \end{tikzcd}$$

classical type system dependent on context

$$\begin{tikzcd}[column sep=huge]
\mathrm{BType}_B
\arrow[r, shift left=14pt, "\overset{\text{co-product}}{\coprod_{b:B}}"]
\arrow[r, phantom, shift left=7pt, "\perp"]
\arrow[r, phantom, shift right=7pt, "\perp"]
\arrow[r, shift right=14pt, "\underset{\text{product}}{\prod_{b:B}}"']
& \mathrm{BType} \arrow[l, "*_B \times" description]
\end{tikzcd}$$

classical type system

classical base change / classical quantification

linear type system in classical context

$$\begin{tikzcd}[column sep=huge]
\big(\mathrm{LType}_B, \overset{\text{tensor}}{\otimes_B}\big)
\arrow[r, shift left=14pt, "\overset{\text{direct sum}}{\bigoplus_{b:B}}"]
\arrow[r, phantom, shift left=7pt, "\perp"]
\arrow[r, phantom, shift right=7pt, "\perp"]
\arrow[r, shift right=14pt, "\bigoplus_{b:B}"']
& \big(\mathrm{LType}, \overset{\text{tensor}}{\otimes}\big) \arrow[l, "\mathbb{1}_B \otimes" description]
\end{tikzcd}$$

linear type system

quantum base change / Motivic Yoga

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\bigoplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\textbf{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\textbf{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\textbf{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd} B \arrow[rr, "p_B"] & & * \end{tikzcd}$$

classical type system dependent on context

$$\begin{tikzcd}[column sep=huge] \mathrm{BType}_B \arrow[r, shift left=8, "\overset{\text{co-product}}{\coprod_{b:B}}"] \arrow[r, phantom, shift left=4, "\perp"] \arrow[r, phantom, shift right=4, "\perp"] \arrow[r, shift right=8, "\underset{\text{product}}{\prod_{b:B}}"'] & \mathrm{BType} \arrow[l, "*_B \times" description] \end{tikzcd}$$

classical type system

classical base change / classical quantification

linear type system in classical context

$$\begin{tikzcd}[column sep=huge] \big(\mathrm{LType}_B, \overset{\text{tensor}}{\otimes_B}\big) \arrow[r, shift left=8, "\overset{\text{direct sum}}{\bigoplus_{b:B}}"] \arrow[r, phantom, shift left=4, "\perp"] \arrow[r, phantom, shift right=4, "\perp"] \arrow[r, shift right=8, "\bigoplus_{b:B}"'] & \big(\mathrm{LType}, \overset{\text{tensor}}{\otimes}\big) \arrow[l, "\mathbb{1}_B \otimes" description] \end{tikzcd}$$

linear type system

quantum base change / Motivic Yoga

Linear/Quantum Data Types

Characteristic Property	**1.** their cartesian product blends into the co-product:	**2.** a tensor product appears & distributes over direct sum	**3.** a linear function type appears adjoint to tensor
Symbol	$\oplus$ direct sum	$\otimes$ tensor product	$\multimap$ linear function type
Formula (for B : FinType)	$\overset{\text{cart. product}}{\prod_B \mathcal{H}_b} \simeq \underset{\text{direct sum}}{\bigoplus_B \mathcal{H}_b} \simeq \overset{\text{co-product}}{\coprod_B \mathcal{H}_b}$	$\mathcal{V} \otimes \Big(\bigoplus_{b:B} \mathcal{H}_b \Big) \simeq \bigoplus_{b:B} (\mathcal{V} \otimes \mathcal{H}_b)$	$(\mathcal{V} \otimes \mathcal{H}) \multimap \mathcal{K}$ $\simeq \mathcal{V} \multimap (\mathcal{H} \multimap \mathcal{K})$

Dependent linear Type Formers

finite classical context (variables, parameters, ...)

reference context

$$\begin{tikzcd}[column sep=huge]
B \arrow[r, "p_B"] & *
\end{tikzcd}$$

all available in LHoTT

classical type system dependent on context

$$\begin{tikzcd}[column sep=huge]
\mathrm{BType}_B \arrow[r, shift left=6, "\overset{\text{co-product}}{\coprod_{b:B}}" description] \arrow[r, phantom, shift left=3, "\perp"] \arrow[r, shift right=6, "\underset{\text{product}}{\prod_{b:B}}" description] \arrow[r, phantom, shift right=3, "\perp"] & \mathrm{BType} \arrow[l, "{*_B}\times" description]
\end{tikzcd}$$

classical type system

classical base change / classical quantification

linear type system in classical context

$$\begin{tikzcd}[column sep=huge]
\Big(\mathrm{LType}_B, \overset{\text{tensor}}{\otimes_B}\Big) \arrow[r, shift left=6, "\overset{\text{direct sum}}{\bigoplus_{b:B}}" description] \arrow[r, phantom, shift left=3, "\perp"] \arrow[r, shift right=6, "\bigoplus_{b:B}" description] \arrow[r, phantom, shift right=3, "\perp"] & \Big(\mathrm{LType}, \overset{\text{tensor}}{\otimes}\Big) \arrow[l, "\mathbb{1}_B \otimes" description]
\end{tikzcd}$$

linear type system

quantum base change / Motivic Yoga

Quantum Effects

Recall: **Monadic computational effects.**

A *monad* $\mathscr{E}(-)$ on a data type system encodes *computational effects*:

effectful program

$$D_1 \xrightarrow{\mathrm{prog}_{12}} \mathscr{E}(D_2)$$

output data of nominal type D_2
causing effects of type $\mathscr{E}(-)$

Recall: **Monadic computational effects.**

A *monad* $\mathscr{E}(-)$ on a data type system encodes *computational effects*:

first program

$$D_1 \xrightarrow{\mathrm{prog}_{12}} \mathscr{E}(D_2)$$

output data of nominal type D_2
causing effects of type $\mathscr{E}(-)$

second program

$$D_2 \xrightarrow{\mathrm{prog}_{23}} \mathscr{E}(D_3)$$

input data of type D_2
causing effects of type $\mathscr{E}(-)$

Recall: **Monadic computational effects.**

A *monad* $\mathscr{E}(-)$ on a data type system encodes *computational effects*:

first program

second program

$$\begin{tikzcd}
D_1 \arrow[r, "\mathrm{prog}_{12}"] & \mathscr{E}(D_2) & D_2 \arrow[r, "\mathrm{prog}_{23}"] & \mathscr{E}(D_3) \arrow[d, mapsto, bend left] \\
D_1 \arrow[r, "\mathrm{prog}_{12}"] & \mathscr{E}(D_2) & \mathscr{E}(D_2) \arrow[r, "\mathrm{bind}^{\mathscr{E}}\mathrm{prog}_{23}"] & \mathscr{E}(D_3)
\end{tikzcd}$$

output data of nominal type D_2
causing effects of type $\mathscr{E}(-)$

input data of type D_2
causing effects of type $\mathscr{E}(-)$

bind previous effects
into second program

carry any previous $\mathscr{E}(-)$-effects along

Recall: **Monadic computational effects.**

A *monad* $\mathscr{E}(-)$ on a data type system encodes *computational effects*:

first program

$$D_1 \xrightarrow{\;\mathrm{prog}_{12}\;} \mathscr{E}(D_2)$$

output data of nominal type D_2
causing effects of type $\mathscr{E}(-)$

second program

$$D_2 \xrightarrow{\;\mathrm{prog}_{23}\;} \mathscr{E}(D_3)$$

input data of type D_2
causing effects of type $\mathscr{E}(-)$

$\mapsto$ **bind previous effects**
into second program

$$D_1 \xrightarrow{\;\mathrm{prog}_{12}\;} \mathscr{E}(D_2) \qquad \mathscr{E}(D_2) \xrightarrow{\;\mathrm{bind}^{\mathscr{E}}\,\mathrm{prog}_{23}\;} \mathscr{E}(D_3)$$

carry any previous $\mathscr{E}(-)$-effects along

$\mapsto$ **compose**

$\mathscr{E}$-composite program

$$D_1 \xrightarrow{\qquad\;\mathrm{bind}^{\mathscr{E}}\,\mathrm{prog}_{23} \circ \mathrm{prog}_{12}\;\qquad} \mathscr{E}(D_3)$$

causing cumulative $\mathscr{E}(-)$-effects

Recall: **Monadic effect handlers.**

$$D_1 \xrightarrow{\text{prog}_{12}} D_2$$ **data type to absorb $\mathscr{E}$-effects**

in-effectful program

Recall: **Monadic effect handlers.**

$$\begin{tikzcd}
D_1 \arrow[r, "\mathrm{prog}_{12}", "\textbf{in-effectful program}"'] & D_2 \arrow[d, mapsto, bend left, "\begin{array}{c}\textbf{incorporate handling}\\ \textbf{of } \mathscr{E}(-)\textbf{-effects}\end{array}"] \\
\mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\,\mathrm{prog}_{12}", "\begin{array}{c}\textbf{in-effectful program}\\ \textbf{handling effects of type } \mathscr{E}(-)\end{array}"'] & D_2
\end{tikzcd}$$

Recall: **Monadic effect handlers.**

$$\begin{tikzcd}[column sep=huge, row sep=large]
D_1 \arrow[r, "\mathrm{prog}_{12}", "\textbf{in-effectful program}"'] & D_2 \arrow[d, mapsto, bend left, "\begin{array}{c}\textbf{incorporate handling}\\ \textbf{of } \mathscr{E}(-)\textbf{-effects}\end{array}"] \\
\mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\mathrm{prog}_{12}", "\begin{array}{c}\textbf{in-effectful program}\\ \textbf{handling effects of type } \mathscr{E}(-)\end{array}"'] & D_2
\end{tikzcd}$$

$$\begin{tikzcd}[column sep=huge]
D_1 \arrow[r, "\mathrm{ret}^{\mathscr{E}}_{D_1}", "\begin{array}{c}\textbf{produce}\\ \textbf{trivial effect}\end{array}"'] \arrow[rr, bend right, "\begin{array}{c}\mathrm{prog}_{12}\\ \textbf{no effect}\end{array}"'] & \mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\mathrm{prog}_{12}", "\begin{array}{c}\textbf{handle effects}\\ \textbf{running program}\end{array}"'] & D_2
\end{tikzcd}$$

consistency conditions

Recall: **Monadic effect handlers.**

$$\begin{tikzcd}
D_1 \arrow[r, "\mathrm{prog}_{12}", "\text{in-effectful program}"'] & D_2 \arrow[d, mapsto, bend left, "\text{incorporate handling of } \mathscr{E}(-)\text{-effects}"] \\
\mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\,\mathrm{prog}_{12}", "\text{in-effectful program handling effects of type } \mathscr{E}(-)"'] & D_2
\end{tikzcd}$$

$$\begin{tikzcd}
D_1 \arrow[r, "\mathrm{ret}^{\mathscr{E}}_{D_1}", "\text{produce trivial effect}"'] \arrow[rr, bend right, "\mathrm{prog}_{12}\ \text{no effect}"'] & \mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\,\mathrm{prog}_{12}", "\text{handle effects running program}"'] & D_2
\end{tikzcd}$$

$$\begin{tikzcd}
\mathscr{E}(D_0) \arrow[r, "\mathrm{bind}^{\mathscr{E}}\,\mathrm{prog}_{01}", "\text{carry effects along}"'] \arrow[rr, bend right, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\left(D_0 \xrightarrow{\mathrm{prog}_{01}} \mathscr{E}(D_1) \xrightarrow{\mathrm{hndl}^{\mathscr{E}}_{D_2}\mathrm{prog}_{12}} D_2\right)"'] & \mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\,\mathrm{prog}_{12}", "\text{handle cumulative effects}"'] & D_2
\end{tikzcd}$$

handle effects... **consecutively**

consistency conditions

Recall: **Data type system of Monadic effect handlers.**

$$\begin{tikzcd}[column sep=huge, row sep=huge]
D_1 \arrow[r, "\mathrm{prog}_{12}", "\textbf{in-effectful program}"'] & D_2 \arrow[d, mapsto, bend left=60, "\textbf{incorporate handling of } \mathscr{E}(-)\textbf{-effects}"] \\
\mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\,\mathrm{prog}_{12}", "{\textbf{in-effectful program handling effects of type } \mathscr{E}(-)}"'] & D_2
\end{tikzcd}$$

“$\mathscr{E}$-modal data type”

Monadicity:

$\mathscr{E}$**-modales in** Type
(“EM-category”)

$\mathrm{Type}^{\mathscr{E}}$

Recall: **Data type system of Monadic effect handlers.**

$$\begin{tikzcd} D_1 \arrow[r, "\mathrm{prog}_{12}", "\text{in-effectful program}"'] & D_2 \arrow[d, mapsto, bend left=60, "\text{incorporate handling of } \mathscr{E}(-)\text{-effects}"] \\ \mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\mathrm{prog}_{12}", "\text{in-effectful program handling effects of type } \mathscr{E}(-)"'] & D_2 \end{tikzcd}$$

“$\mathscr{E}$-modal data type”

Monadicity:

$$\begin{tikzcd} \mathrm{Type} \arrow[loop right, "\mathscr{E}"] \end{tikzcd}$$

monad

$\mathscr{E}$-modales in Type (“EM-category”) $\mathrm{Type}^{\mathscr{E}}$

Recall: **Data type system of Monadic effect handlers.**

$$\begin{tikzcd}
D_1 \arrow[r, "\mathrm{prog}_{12}"] & D_2 \arrow[d, mapsto, bend left] \\
\mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\,\mathrm{prog}_{12}"] & D_2
\end{tikzcd}$$

in-effectful program

incorporate handling of $\mathscr{E}(-)$-effects

“$\mathscr{E}$-modal data type”

in-effectful program handling effects of type $\mathscr{E}(-)$

Monadicity:

$$\begin{tikzcd}
& \mathrm{Type} \arrow[loop right, "\mathscr{E}"] \arrow[dl, shift right=2, "F^{\mathscr{E}}"'] \\
\mathrm{Type}^{\mathscr{E}} \arrow[ur, shift right=2, "U^{\mathscr{E}}"'] \arrow[ur, phantom, "\perp"] &
\end{tikzcd}$$

induced monad

monadic adjunction

$\mathscr{E}$**-modales in** Type

(“EM-category”)

Recall: **Data type system of Monadic effect handlers.**

$$\begin{tikzcd}
D_1 \arrow[r, "\mathrm{prog}_{12}"] & D_2 \arrow[d, mapsto, bend left, "\text{incorporate handling of } \mathscr{E}(-)\text{-effects}"] \\
\mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\mathrm{prog}_{12}"] & D_2
\end{tikzcd}$$

in-effectful program

“$\mathscr{E}$-modal data type”

in-effectful program
handling effects of type $\mathscr{E}(-)$

Monadicity:

$$\begin{tikzcd}
\mathrm{Type}' \arrow[d, "K^{UF}"] \arrow[rr, shift right, "U"'] \arrow[rr, phantom, "\perp"] & & \mathrm{Type} \arrow[ll, shift right, "F"'] \arrow[loop right, "\mathscr{E}"] \arrow[dll, shift right, "F^{\mathscr{E}}"'] \\
\mathrm{Type}^{\mathscr{E}} \arrow[urr, shift right, "U^{\mathscr{E}}"'] \arrow[urr, phantom, "\perp"] & &
\end{tikzcd}$$

any adjunction
on type system

induced monad

terminal comparison functor

monadic adjunction

$\mathscr{E}$-modales in Type
(“EM-category”)

Recall: **Data type system of Monadic effect handlers.**

$$\begin{tikzcd}
D_1 \arrow[r, "\mathrm{prog}_{12}", "\text{in-effectful program}"'] & D_2 \arrow[d, mapsto, bend left, "\text{incorporate handling of } \mathscr{E}(-)\text{-effects}"] \\
\mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\,\mathrm{prog}_{12}", "\text{in-effectful program handling effects of type } \mathscr{E}(-)"'] & D_2 \;\; \text{“}\mathscr{E}\text{-modal data type”}
\end{tikzcd}$$

Monadicity:

$$\begin{tikzcd}[column sep=huge, row sep=huge]
\mathrm{Type}_{\mathscr{E}} \arrow[d, hook, "K_{UF}"] \arrow[dr, shift right, "U_{\mathscr{E}}"'] & & \\
\mathrm{Type}' \arrow[d, "K^{UF}"] \arrow[r, shift right, "U"'] \arrow[r, phantom, "\perp"] & \mathrm{Type} \arrow[l, shift right, "F"'] \arrow[ul, two heads, shift right, "F_{\mathscr{E}}"'] \arrow[ul, phantom, "\perp"] \arrow[dl, shift right, "F^{\mathscr{E}}"'] \arrow[dl, phantom, "\perp"] \arrow[loop right, "\mathscr{E}"] \\
\mathrm{Type}^{\mathscr{E}} \arrow[ur, shift right, "U^{\mathscr{E}}"'] & &
\end{tikzcd}$$

free $\mathscr{E}$-modales in Type
(“Kleisli category”)

initial comparison functor

any adjunction
on type system

terminal comparison functor

$\mathscr{E}$-modales in Type
(“EM-category”)

monadic adjunction

induced monad

Recall: **Data type system of Monadic effect handlers.**

$$\begin{tikzcd} D_1 \arrow[r, "\mathrm{prog}_{12}"] & D_2 & {} \arrow[d, mapsto, bend left, "\text{incorporate handling of } \mathscr{E}(-)\text{-effects}"] \\ \mathscr{E}(D_1) \arrow[r, "\mathrm{hndl}^{\mathscr{E}}_{D_2}\mathrm{prog}_{12}"] & D_2 & {} \end{tikzcd}$$

in-effectful program

incorporate handling of $\mathscr{E}(-)$-effects

“$\mathscr{E}$-modal data type”

in-effectful program
handling effects of type $\mathscr{E}(-)$

Monadicity:

$$\begin{tikzcd} \mathrm{Type}_{\mathscr{E}} \arrow[d, hook, "K_{UF}"] \arrow[dr, shift left, "U_{\mathscr{E}}"'] & \\ \mathrm{Type}' \arrow[d, "K^{UF}"] \arrow[r, shift right, "U"'] \arrow[r, phantom, "\perp"] & \mathrm{Type} \arrow[l, shift right, "F"'] \arrow[ul, two heads, shift left, "F_{\mathscr{E}}"'] \arrow[dl, shift right, "F^{\mathscr{E}}"] \arrow[loop right, "\mathscr{E}"] \\ \mathrm{Type}^{\mathscr{E}} \arrow[ur, shift right, "U^{\mathscr{E}}"'] & \end{tikzcd}$$

free $\mathscr{E}$-modales in Type
(“Kleisli category”)

initial comparison functor

any adjunction
on type system

terminal comparison functor

$\mathscr{E}$-modales in Type
(“EM-category”)

induced monad

monadic adjunction

Now just to work this out
for the **effects induced by**
dependent data type formers
in LHoTT →

Given $B : \mathrm{BType}$ of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent data type formers** constitute modalities of **actual and potential B-measurements**:

classical data types

possibly **randomly**

$$\begin{tikzcd}[column sep=huge]
\mathrm{Type}_B \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"'] \arrow[r, shift left=8, "\coprod_B"] \arrow[r, phantom, shift left=4, "\perp"] \arrow[r, phantom, shift right=4, "\perp"] \arrow[r, shift right=8, "\prod_B"'] & \mathrm{Type} \arrow[l, "\times B" description] \arrow[loop above, "\star_B"'] \arrow[loop below, "\bigcirc_B"]
\end{tikzcd}$$

actual data $\perp$ $\perp$ **potential data**

necessarily **indefinitely**

Given $B : \mathrm{BType}$ of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent data type formers** constitute modalities of **actual and potential B-measurements**:

classical
data types

possibly **randomly**

$$\begin{tikzcd}
\mathrm{Type}_B \arrow[loop above, "\lozenge_B"] \arrow[loop below, "\square_B"'] \arrow[r, shift left=3, "\coprod_B"] \arrow[r, shift right=3, "\prod_B"'] & \mathrm{Type} \arrow[l, "\times B" description] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"']
\end{tikzcd}$$

actual data $\perp$ $\quad$ $\perp$ **potential data**

necessarily **indefinitely**

---

necessarily $P_\bullet$

$$\square_B\, P_\bullet$$

$$b : B \;\vdash\; \prod_{b' : B} P_{b'}$$

Given B : BType of possible measurement outcomes ("possible worlds")
the monadic effects of B-dependent data type formers constitute modalities of **actual and potential B-measurements**:

classical data types

possibly **randomly**

$$\begin{tikzcd}
\mathrm{Type}_B \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"'] \arrow[r, shift left=4, "\coprod_B"] \arrow[r, phantom, shift left=2, "\perp"] \arrow[r, phantom, shift right=2, "\perp"] \arrow[r, shift right=4, "\prod_B"'] & \mathrm{Type} \arrow[l, "\times B"'] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"']
\end{tikzcd}$$

actual data $\perp$ $\perp$ **potential data**

necessarily **indefinitely**

---

necessarily $P_\bullet$ **entails** **actually $P_\bullet$**

$$\Box_B P_\bullet \xrightarrow{\;\epsilon^{\Box_B}_{P_\bullet}\;} P_\bullet$$

$$b : B \;\vdash\; \prod_{b':B} P_{b'} \xrightarrow{(p_{b'})_{b':B} \,\mapsto\, p_b} P_b$$

Given B : BType of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent data type formers** constitute modalities of **actual and potential B-measurements**:

classical data types

possibly **randomly**

$$\begin{tikzcd}
\mathrm{Type}_B \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"'] \arrow[r, shift left=4, "\coprod_B"] \arrow[r, phantom, shift left=2, "\perp"] \arrow[r, phantom, shift right=2, "\perp"] \arrow[r, shift right=4, "\prod_B"'] & \mathrm{Type} \arrow[l, "\times B" description] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"']
\end{tikzcd}$$

actual data $\perp$ $\perp$ **potential data**

necessarily **indefinitely**

---

necessarily $P_\bullet$ **entails** **actually $P_\bullet$** **entails** **possibly $P_\bullet$**

$$\Box_B P_\bullet \xrightarrow{\;\epsilon^{\Box_B}_{P_\bullet}\;} P_\bullet \xrightarrow{\;\eta^{\Diamond_B}_{P_\bullet}\;} \Diamond_B P_\bullet$$

$$b : B \;\vdash\; \prod_{b':B} P_{b'} \xrightarrow{\;(p_{b'})_{b':B} \;\mapsto\; p_b\;} P_b \xrightarrow{\;p_b \;\mapsto\; (p_b)_b\;} \coprod_{b':B} P_{b'}$$

Given B : BType of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent data type formers** constitute modalities of **actual and potential B-measurements**:

classical data types

possibly **randomly**

actual data $\perp$ **potential data**

necessarily **indefinitely**

$$\begin{tikzcd}
\mathrm{Type}_B \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"] \arrow[r, shift left=6, "\coprod_B"] \arrow[r, shift right=6, "\prod_B"'] & \mathrm{Type} \arrow[l, "\times B"' description] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"]
\end{tikzcd}$$

$$\begin{tikzcd}
\coprod_B \arrow[d, phantom, "\perp"] \\
\times B \arrow[d, phantom, "\perp"] \\
\prod_B
\end{tikzcd}$$

---

necessarily $P_\bullet$ **entails** **actually $P_\bullet$** **entails** **possibly $P_\bullet$**

$$\begin{tikzcd}
\Box_B P_\bullet \arrow[r, "\epsilon^{\Box_B}_{P_\bullet}" description] & P_\bullet \arrow[r, "\eta^{\Diamond_B}_{P_\bullet}" description] & \Diamond_B P_\bullet
\end{tikzcd}$$

$$b : B \ \vdash \ \begin{tikzcd}
\prod\limits_{b':B} P_{b'} \arrow[r, "(p_{b'})_{b':B} \,\mapsto\, p_b"] & P_b \arrow[r, "p_b \,\mapsto\, (p_b)_b"] & \coprod\limits_{b':B} P_{b'}
\end{tikzcd}$$

randomly P

$$\star_B P$$

$$\coprod_{b:B} P$$

Given $B :$ BType of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent data type formers** constitute modalities of **actual and potential B-measurements**:

classical data types

possibly **randomly**

actual data **potential data**

$$\begin{tikzcd}
\perp & \mathrm{Type}_B \arrow[loop above, "\lozenge_B"] \arrow[loop below, "\Box_B"'] \arrow[rr, shift left=4, "\coprod_B"] \arrow[rr, shift left=2, phantom, "\perp"] \arrow[rr, shift right=2, phantom, "\perp"] \arrow[rr, shift right=4, "\prod_B"'] & & \mathrm{Type} \arrow[ll, "\times B" description] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"'] & \perp
\end{tikzcd}$$

necessarily **indefinitely**

necessarily $P_\bullet$ **entails** **actually $P_\bullet$** **entails** **possibly $P_\bullet$**

$$\Box_B P_\bullet \xrightarrow{\epsilon^{\Box_B}_{P_\bullet}} P_\bullet \xrightarrow{\eta^{\lozenge_B}_{P_\bullet}} \lozenge_B P_\bullet$$

$$b : B \;\vdash\; \prod_{b':B} P_{b'} \xrightarrow{(p_{b'})_{b':B} \;\mapsto\; p_b} P_b \xrightarrow{p_b \;\mapsto\; (p_b)_b} \coprod_{b':B} P_{b'}$$

randomly P **entails** **potentially P**

$$\star_B P \xrightarrow{\epsilon^{\star_B}_{P}} P$$

$$\coprod_{b:B} P \xrightarrow{(p)_b \;\mapsto\; p} P$$

Given B : BType of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent data type formers** constitute modalities of **actual and potential B-measurements**:

classical data types

possibly **randomly**

actual data $\perp$ **potential data**

necessarily **indefinitely**

$$\begin{tikzcd}
\mathrm{Type}_B \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"] \arrow[r, shift left=4, "\coprod_B"] \arrow[r, shift right=4, "\prod_B"'] & \mathrm{Type} \arrow[l, "\times B" description] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"]
\end{tikzcd}$$

$$\coprod_B \dashv \times B \dashv \prod_B \qquad \Box_B \perp \Diamond_B \qquad \star_B \perp \bigcirc_B$$

necessarily $P_\bullet$ **entails** **actually $P_\bullet$** **entails** **possibly $P_\bullet$**

$$\Box_B P_\bullet \xrightarrow{\;\varepsilon^{\Box_B}_{P_\bullet}\;} P_\bullet \xrightarrow{\;\eta^{\Diamond_B}_{P_\bullet}\;} \Diamond_B P_\bullet$$

$$b : B \;\vdash\; \prod_{b':B} P_{b'} \xrightarrow{(p_{b'})_{b':B} \,\mapsto\, p_b} P_b \xrightarrow{p_b \,\mapsto\, (p_b)_b} \coprod_{b':B} P_{b'}$$

randomly P **entails** **potentially P** **entails** **indefinitely P**

$$\star_B P \xrightarrow{\;\varepsilon^{\star_B}_{P}\;} P \xrightarrow{\;\eta^{\bigcirc_B}_{P}\;} \bigcirc_B P$$

$$\coprod_{b:B} P \xrightarrow{(p)_b \,\mapsto\, p} P \xrightarrow{p \,\mapsto\, (p)_{b:B}} \prod_{b:B} P$$

Given B : BType of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent linear data type formers** constitute modalities of **actual and potential quantum B-measurements**.

quantum data types

$$\begin{tikzcd}
& \text{possibly} &&&& \text{randomly} \\
&& (\mathrm{LType}_B)_{\Diamond_B} \arrow[r, hook, "\sim"] & (\mathrm{LType}_B)^{\Diamond_B} \arrow[dr, phantom, "\simeq"] && \\
\text{actual data } \perp & \mathrm{LType}_B \simeq \mathrm{LType}^{\bigcirc_B} \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"'] \arrow[ur, two heads, "\oplus_B"] \arrow[dr, two heads, "\oplus_B"'] & \mathrm{LType}_{\bigcirc_B} \arrow[l, hook] \arrow[u, phantom, "\perp"] \arrow[d, phantom, "\perp"] && \mathrm{LType} \arrow[ll, two heads, "\mathbb{1}_B \otimes"'] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"'] & \perp \text{ potential data} \\
&& (\mathrm{LType}_B)_{\Box_B} \arrow[r, hook, "\sim"'] & (\mathrm{LType}_B)^{\Box_B} \arrow[ur, phantom, "\simeq"] && \\
& \text{necessarily} &&&& \text{indefinitely}
\end{tikzcd}$$

Given $B : \mathrm{BType}$ of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent linear data type formers** constitute modalities of **actual and potential quantum B-measurements**.

quantum data types

possibly **randomly**

actual data $\perp$ $\perp$ **potential data**

$$\begin{tikzcd}
& (\mathrm{LType}_B)_{\Diamond_B} \arrow[r, hook, "\sim"] \arrow[d, phantom, "\perp"] & (\mathrm{LType}_B)^{\Diamond_B} \\
\mathrm{LType}_B \simeq \mathrm{LType}^{\bigcirc_B} \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"] \arrow[ur, two heads, "\oplus_B"] \arrow[dr, two heads, "\oplus_B"'] & \mathrm{LType}_{\bigcirc_B} \arrow[l, hook'] \arrow[d, phantom, "\perp"] & \mathrm{LType} \arrow[l, two heads, "\mathbb{1}_B \otimes"'] \arrow[u, phantom, "\simeq"] \arrow[d, phantom, "\simeq"] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"] \\
& (\mathrm{LType}_B)_{\Box_B} \arrow[r, hook, "\sim"'] & (\mathrm{LType}_B)^{\Box_B}
\end{tikzcd}$$

necessarily **indefinitely**

necessarily $\mathcal{H}_\bullet$

$\Box_B \, \mathcal{H}_\bullet$

Given... **obtain...**

$b : B \;\vdash\; \mathcal{H}$

measurement result

where $\mathcal{H} \;:=\; \bigoplus_{b' : B} \mathcal{H}_{b'}$

Given B : BType of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent linear data type formers** constitute modalities of **actual and potential quantum B-measurements**.

quantum
data types

possibly **randomly**

$$\begin{tikzcd}
{} & {} & (\mathrm{LType}_B)_{\Diamond_B} \arrow[r, hook, "\sim"] \arrow[d, phantom, "\perp"] & (\mathrm{LType}_B)^{\Diamond_B} & {} \\
\mathrm{LType}_B \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"] \arrow[r, phantom, "\simeq"] \arrow[urr, two heads, "\oplus_B"] \arrow[drr, two heads, "\oplus_B"'] & \mathrm{LType}^{\bigcirc_B} & \mathrm{LType}_{\bigcirc_B} \arrow[l, hook'] \arrow[d, phantom, "\perp"] & {} & \mathrm{LType} \arrow[ll, two heads, "\mathbb{1}_B\otimes"'] \arrow[ul, dash, "\wr"'] \arrow[dl, dash, "\wr"] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"] \\
{} & {} & (\mathrm{LType}_B)_{\Box_B} \arrow[r, hook, "\sim"'] & (\mathrm{LType}_B)^{\Box_B} & {}
\end{tikzcd}$$

actual data $\perp$ $\perp$ **potential data**

necessarily **indefinitely**

necessarily $\mathcal{H}_\bullet$ **entails** **actually $\mathcal{H}_\bullet$**

$$\begin{tikzcd}
\Box_B \mathcal{H}_\bullet \arrow[rr, "\varepsilon^{\Box_B}_{\mathcal{H}_\bullet}"] & & \mathcal{H}_\bullet
\end{tikzcd}$$

Given... **obtain...**

$$b : B \;\vdash\; \mathcal{H} \xrightarrow[\text{measurement collapse}]{\sum_{b'} |\psi_{b'}\rangle \;\mapsto\; |\psi_b\rangle} \mathcal{H}_b \qquad \text{where } \mathcal{H} := \bigoplus_{b':B} \mathcal{H}_{b'}$$

measurement result

Given $B : \mathrm{BType}$ of possible measurement outcomes ("possible worlds") **the monadic effects of B-dependent linear data type formers** constitute modalities of **actual and potential quantum B-measurements**.

quantum
data types

possibly

actual data $\perp$

necessarily

randomly

$\perp$ **potential data**

indefinitely

$$\begin{tikzcd}
& (\mathrm{LType}_B)_{\lozenge_B} \arrow[r, hook, "\sim"] \arrow[d, phantom, "\perp"] & (\mathrm{LType}_B)^{\lozenge_B} \arrow[dr, dash, "\sim"] & \\
\mathrm{LType}_B \simeq \mathrm{LType}^{\bigcirc_B} \arrow[loop above, "\lozenge_B"] \arrow[loop below, "\square_B"] \arrow[ur, two heads, "\oplus_B"] \arrow[dr, two heads, "\oplus_B"'] & \mathrm{LType}_{\bigcirc_B} \arrow[l, hook'] \arrow[d, phantom, "\perp"] & & \mathrm{LType} \arrow[ll, two heads, "\mathbb{1}_B \otimes"'] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"] \\
& (\mathrm{LType}_B)_{\square_B} \arrow[r, hook, "\sim"'] & (\mathrm{LType}_B)^{\square_B} \arrow[ur, dash, "\sim"'] &
\end{tikzcd}$$

necessarily $\mathcal{H}_\bullet$ **entails** **actually $\mathcal{H}_\bullet$** **entails** **possibly $\mathcal{H}_\bullet$**

$$\begin{tikzcd}
\square_B \mathcal{H}_\bullet \arrow[r, "\epsilon^{\square_B}_{\mathcal{H}_\bullet}"] & \mathcal{H}_\bullet \arrow[r, "\eta^{\lozenge_B}_{\mathcal{H}_\bullet}"] & \lozenge_B \mathcal{H}_\bullet
\end{tikzcd}$$

Given... **obtain...**

$b : B \vdash$

measurement result

$$\begin{tikzcd}
\mathcal{H} \arrow[r, two heads, "\sum_{b'} |\psi_{b'}\rangle \;\mapsto\; |\psi_b\rangle", "\text{measurement collapse}"'] \arrow[rr, bend right, "\text{linear projector onto sub-Hilbert space } \mathcal{H}_b"'] & \mathcal{H}_b \arrow[r, hook, "|\psi_b\rangle \mapsto \oplus_{b'} \begin{cases} |\psi_b\rangle & \text{if } b'=b \\ 0 & \text{else} \end{cases}", "\text{state preparation}"'] & \mathcal{H},
\end{tikzcd}$$

where $\mathcal{H} := \bigoplus\limits_{b':B} \mathcal{H}_{b'}$

Given $B : \mathrm{BType}$ of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent linear data type formers** constitute modalities of **actual and potential quantum B-measurements**.

quantum data types

possibly **randomly**

actual data $\perp$ $\quad$ $\perp$ **potential data**

$$\begin{tikzcd}
 & & (\mathrm{LType}_B)_{\Diamond_B} \arrow[r, hook, "\sim"] & (\mathrm{LType}_B)^{\Diamond_B} & \\
\mathrm{LType}_B \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"'] \arrow[urr, two heads, "\oplus_B"] \arrow[drr, two heads, "\oplus_B"'] \arrow[r, phantom, "\simeq"] & \mathrm{LType}^{\bigcirc_B} & \mathrm{LType}_{\bigcirc_B} \arrow[l, hook'] \arrow[u, phantom, "\perp"] \arrow[d, phantom, "\perp"] & & \mathrm{LType} \arrow[ll, two heads, "\mathbb{1}_B \otimes"'] \arrow[ul, phantom, "\simeq"] \arrow[dl, phantom, "\simeq"] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"'] \\
 & & (\mathrm{LType}_B)_{\Box_B} \arrow[r, hook, "\sim"'] & (\mathrm{LType}_B)^{\Box_B} &
\end{tikzcd}$$

necessarily **indefinitely**

---

principle of quantum compulsion:

necessarily $\mathcal{H}_\bullet$ **entails** **actually $\mathcal{H}_\bullet$** **entails** **possibly $\mathcal{H}_\bullet$** **is** **necessarily $\mathcal{H}_\bullet$**

$$\begin{tikzcd}
\Box_B \mathcal{H}_\bullet \arrow[r, "\varepsilon^{\Box_B}_{\mathcal{H}_\bullet}"] & \mathcal{H}_\bullet \arrow[r, "\eta^{\Diamond_B}_{\mathcal{H}_\bullet}"] & \Diamond_B \mathcal{H}_\bullet \arrow[r, phantom, "\simeq"'] & \Box_B \mathcal{H}_\bullet
\end{tikzcd}$$

ambidexterity

Given... **obtain...**

$$\begin{tikzcd}
b : B \vdash & \mathcal{H} \arrow[r, two heads, "\Sigma_{b'} |\psi_{b'}\rangle \,\mapsto\, |\psi_b\rangle", "\text{measurement collapse}"'] \arrow[rr, bend right, "\text{linear projector onto sub-Hilbert space } \mathcal{H}_b"'] & \mathcal{H}_b \arrow[r, hook, "|\psi_b\rangle \mapsto \oplus_{b'} \begin{cases} |\psi_b\rangle & \text{if } b'=b \\ 0 & \text{else} \end{cases}", "\text{state preparation}"'] & \mathcal{H},
\end{tikzcd}
\qquad \text{where } \mathcal{H} := \bigoplus_{b' : B} \mathcal{H}_{b'}$$

measurement result

Given $B : \mathrm{BType}$ of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent linear data type formers** constitute modalities of **actual and potential quantum B-measurements**.

quantum data types

possibly **randomly**

actual data $\perp$ $\quad$ $\perp$ **potential data**

$$\begin{tikzcd}
 & (\mathrm{LType}_B)_{\Diamond_B} \arrow[r, hook, "\sim"] \arrow[d, phantom, "\perp"] & (\mathrm{LType}_B)^{\Diamond_B} \arrow[dr, phantom, "\simeq"] & \\
\mathrm{LType}_B \simeq \mathrm{LType}^{\bigcirc_B} \arrow[ur, two heads, "\oplus_B"] \arrow[dr, two heads, "\oplus_B"'] \arrow[loop above, "\Diamond_B"] \arrow[loop below, "\Box_B"'] & \mathrm{LType}_{\bigcirc_B} \arrow[l, hook'] \arrow[d, phantom, "\perp"] & & \mathrm{LType} \arrow[ll, two heads, "\mathbb{1}_B \otimes"'] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"'] \\
 & (\mathrm{LType}_B)_{\Box_B} \arrow[r, hook, "\sim"'] & (\mathrm{LType}_B)^{\Box_B} \arrow[ur, phantom, "\simeq"] &
\end{tikzcd}$$

necessarily **indefinitely**

---

principle of quantum compulsion:

necessarily $\mathcal{H}_\bullet$ **entails** **actually $\mathcal{H}_\bullet$** **entails** **possibly $\mathcal{H}_\bullet$** **is** **necessarily $\mathcal{H}_\bullet$**

$$\Box_B \mathcal{H}_\bullet \xrightarrow{\epsilon^{\Box_B}_{\mathcal{H}_\bullet}} \mathcal{H}_\bullet \xrightarrow{\eta^{\Diamond_B}_{\mathcal{H}_\bullet}} \Diamond_B \mathcal{H}_\bullet \underset{\text{ambidexterity}}{\simeq} \Box_B \mathcal{H}_\bullet$$

Given... **obtain...**

$$\underset{\text{measurement result}}{b : B} \;\vdash\; \begin{tikzcd}
\mathcal{H} \arrow[r, two heads, "\sum_{b'} |\psi_{b'}\rangle \;\mapsto\; |\psi_b\rangle", "\text{measurement collapse}"'] \arrow[rr, bend right, "\text{linear projector onto sub-Hilbert space } \mathcal{H}_b"'] & \mathcal{H}_b \arrow[r, hook, "|\psi_b\rangle \mapsto \oplus_{b'} \begin{cases} |\psi_b\rangle & \text{if } b'=b \\ 0 & \text{else} \end{cases}", "\text{state preparation}"'] & \mathcal{H},
\end{tikzcd} \quad \text{where } \mathcal{H} := \bigoplus_{b':B} \mathcal{H}_{b'}$$

---

randomly $\mathcal{H}$

$$\star_B \mathcal{H}$$

$$\bigoplus_{b:B} \mathcal{H}$$

Given B : BType of possible measurement outcomes (“possible worlds”) **the monadic effects of B-dependent linear data type formers** constitute modalities of **actual and potential quantum B-measurements**.

quantum data types

possibly **randomly**

actual data $\perp$ $\perp$ **potential data**

$$\begin{tikzcd}
& (\mathrm{LType}_B)_{\lozenge_B} \arrow[r, hook, "\sim"] \arrow[d, phantom, "\perp"] & (\mathrm{LType}_B)^{\lozenge_B} \arrow[dr, phantom, "\simeq"] & \\
\mathrm{LType}_B \simeq \mathrm{LType}^{\bigcirc_B} \arrow[ur, two heads, "\bigoplus_B"] \arrow[dr, two heads, "\bigoplus_B"'] \arrow[loop above, "\lozenge_B"] \arrow[loop below, "\Box_B"] & \mathrm{LType}_{\bigcirc_B} \arrow[l, hook'] \arrow[d, phantom, "\perp"] & & \mathrm{LType} \arrow[ll, two heads, "\mathbb{1}_B \otimes"'] \arrow[loop above, "\star_B"] \arrow[loop below, "\bigcirc_B"] \\
& (\mathrm{LType}_B)_{\Box_B} \arrow[r, hook, "\sim"'] & (\mathrm{LType}_B)^{\Box_B} \arrow[ur, phantom, "\simeq"] &
\end{tikzcd}$$

necessarily **indefinitely**

---

principle of quantum compulsion:

necessarily $\mathcal{H}_\bullet$ **entails** **actually $\mathcal{H}_\bullet$** **entails** **possibly $\mathcal{H}_\bullet$** **is** **necessarily $\mathcal{H}_\bullet$**

$$\Box_B \mathcal{H}_\bullet \xrightarrow{\;\epsilon^{\Box_B}_{\mathcal{H}_\bullet}\;} \mathcal{H}_\bullet \xrightarrow{\;\eta^{\lozenge_B}_{\mathcal{H}_\bullet}\;} \lozenge_B \mathcal{H}_\bullet \;\simeq\; \Box_B \mathcal{H}_\bullet$$

ambidexterity

Given... **obtain...**

$$b : B \;\vdash\; \begin{tikzcd}
\mathcal{H} \arrow[r, two heads, "\sum_{b'} |\psi_{b'}\rangle \;\mapsto\; |\psi_b\rangle", "\text{measurement collapse}"'] \arrow[rr, bend right, "\text{linear projector onto sub-Hilbert space } \mathcal{H}_b"'] & \mathcal{H}_b \arrow[r, hook, "|\psi_b\rangle \mapsto \oplus_{b'} \left\{ \begin{array}{ll} |\psi_b\rangle & \text{if } b'{=}b \\ 0 & \text{else} \end{array} \right.", "\text{state preparation}"'] & \mathcal{H},
\end{tikzcd} \quad \text{where } \mathcal{H} := \bigoplus_{b' : B} \mathcal{H}_{b'}$$

measurement result

measurement collapse **state preparation**

linear projector onto sub-Hilbert space $\mathcal{H}_b$

---

randomly $\mathcal{H}$ **entails** **potentially $\mathcal{H}$**

$$\star_B \mathcal{H} \xrightarrow{\;\epsilon^{\star_B}_{\mathcal{H}}\;} \mathcal{H}$$

$$\bigoplus_{b:B} \mathcal{H} \xrightarrow[\text{quantum superposition}]{\;\oplus_b |\psi_b\rangle \,\mapsto\, \sum_b |\psi_b\rangle\;} \mathcal{H}$$

quantum superposition

Given B : BType of possible measurement outcomes (“possible worlds”)
the monadic effects of B-dependent linear data type formers constitute modalities of **actual and potential quantum B-measurements.**

quantum
data types

$$\begin{tikzcd}
 & & (\mathrm{LType}_B)_{\Diamond_B} \arrow[r, hook, "\sim"] \arrow[d, phantom, "\perp"] & (\mathrm{LType}_B)^{\Diamond_B} \arrow[rd, phantom, "\sim"] & \\
\text{actual data } \perp & \mathrm{LType}_B \simeq \mathrm{LType}^{\bigcirc_B} \arrow[loop above, "\text{possibly } \Diamond_B"] \arrow[loop below, "\text{necessarily } \Box_B"'] \arrow[ru, two heads, "\oplus_B"] \arrow[rd, two heads, "\oplus_B"'] & \mathrm{LType}_{\bigcirc_B} \arrow[l, hook'] \arrow[d, phantom, "\perp"] & & \mathrm{LType} \arrow[ll, two heads, "\mathbb{1}_B\otimes"'] \arrow[loop above, "\text{randomly } \star_B"] \arrow[loop below, "\text{indefinitely } \bigcirc_B"'] \quad \perp \text{ potential data} \\
 & & (\mathrm{LType}_B)_{\Box_B} \arrow[r, hook, "\sim"'] & (\mathrm{LType}_B)^{\Box_B} \arrow[ru, phantom, "\sim"] &
\end{tikzcd}$$

---

principle of quantum compulsion:

necessarily $\mathcal{H}_\bullet$ **entails** **actually $\mathcal{H}_\bullet$** **entails** **possibly $\mathcal{H}_\bullet$** **is** **necessarily $\mathcal{H}_\bullet$**

$$\begin{tikzcd}
\Box_B \mathcal{H}_\bullet \arrow[r, "\epsilon^{\Box_B}_{\mathcal{H}_\bullet}"] & \mathcal{H}_\bullet \arrow[r, "\eta^{\Diamond_B}_{\mathcal{H}_\bullet}"] & \Diamond_B \mathcal{H}_\bullet \arrow[r, phantom, "\simeq"] & \Box_B \mathcal{H}_\bullet
\end{tikzcd}$$

ambidexterity

Given... **obtain...**

$$\begin{tikzcd}
b : B \ \vdash & \mathcal{H} \arrow[r, two heads, "\sum_{b'} |\psi_{b'}\rangle \ \mapsto \ |\psi_b\rangle", "\text{measurement collapse}"'] \arrow[rr, bend right, "\text{linear projector onto sub-Hilbert space } \mathcal{H}_b"'] & \mathcal{H}_b \arrow[r, hook, "|\psi_b\rangle \mapsto \oplus_{b'} \begin{cases} |\psi_b\rangle & \text{if } b'=b \\ 0 & \text{else} \end{cases}", "\text{state preparation}"'] & \mathcal{H},
\end{tikzcd}$$

measurement result

$$\text{where } \mathcal{H} := \bigoplus_{b':B} \mathcal{H}_{b'}$$

- -

randomly $\mathcal{H}$ **entails** **potentially $\mathcal{H}$** **entails** **indefinitely $\mathcal{H}$**

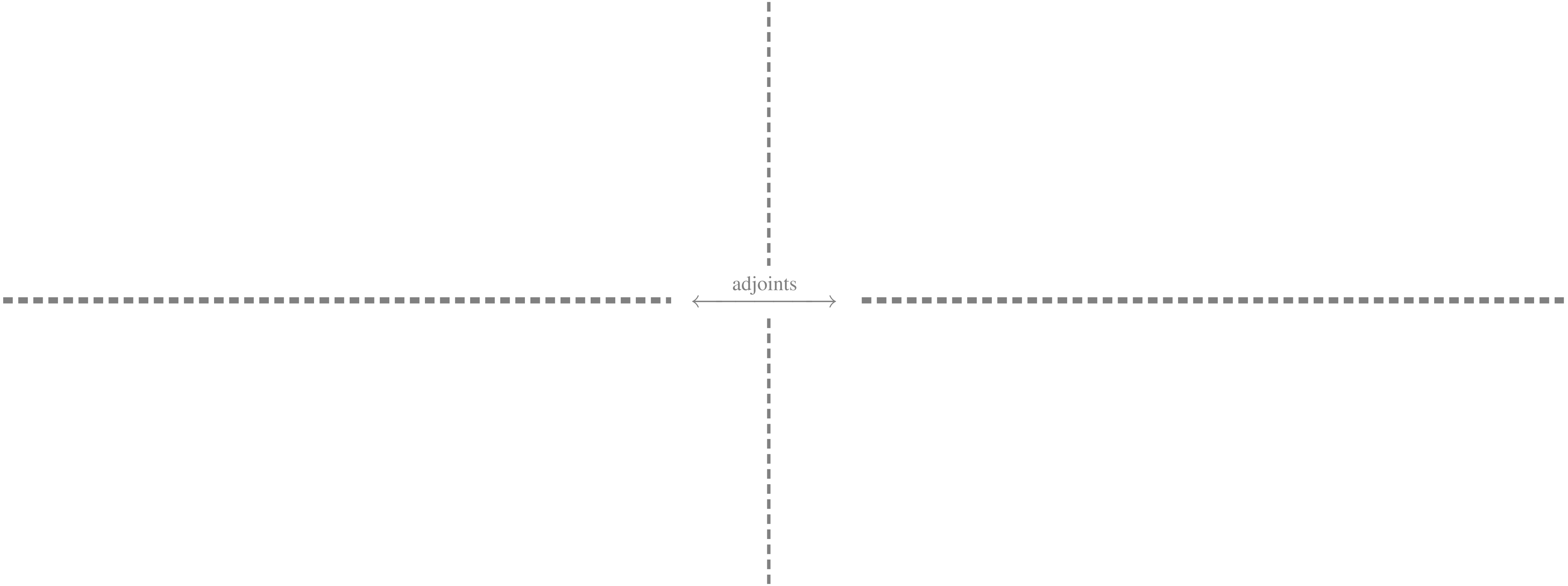
$$\begin{tikzcd}
\star_B \mathcal{H} \arrow[r, "\epsilon^{\star_B}_{\mathcal{H}}"] & \mathcal{H} \arrow[r, "\eta^{\bigcirc_B}_{\mathcal{H}}"] & \bigcirc_B \mathcal{H}
\end{tikzcd}$$

$$\begin{tikzcd}
\bigoplus_{b:B} \mathcal{H} \arrow[r, "\oplus_b |\psi_b\rangle \mapsto \sum_b |\psi_b\rangle", "\text{quantum superposition}"'] & \mathcal{H} \arrow[r, "|\psi\rangle \mapsto \oplus_b |\psi\rangle_b", "\text{quantum parallelization}"'] & \bigoplus_{b:B} \mathcal{H}
\end{tikzcd}$$

The pure effects of these modalities of dependent linear data type formation

are remarkable in their sheer quantum information-theoretic content.

To repeat:



The pure effects of these modalities of dependent linear data type formation

are remarkable in their sheer quantum information-theoretic content.

To repeat:

$$\overbrace{(p_B)^*(p_B)_*}^{\Box_B}\mathcal{H}_\bullet \xrightarrow[\textbf{necessity counit}]{\varepsilon^{\Box_B}_{\mathcal{H}_\bullet}} \mathcal{H}_\bullet$$

$$b : B \vdash \bigoplus_{b':B}\mathcal{H}_{b'} \xrightarrow[\text{quantum }\textbf{measurement}]{\oplus_{b'}|\psi_{b'}\rangle \;\mapsto\; |\psi_b\rangle} \mathcal{H}_b$$

“ the necessary becomes actual ”

adjoints $\longleftrightarrow$

The pure effects of these modalities of dependent linear data type formation

are remarkable in their sheer quantum information-theoretic content.

To repeat:

$$\overbrace{(p_B)^*(p_B)_*}^{\Box_B}\mathcal{H}_\bullet \xrightarrow[\textbf{necessity counit}]{\varepsilon^{\Box_B}_{\mathcal{H}_\bullet}} \mathcal{H}_\bullet$$

$$b : B \vdash \bigoplus_{b':B}\mathcal{H}_{b'} \xrightarrow[\text{quantum }\textbf{measurement}]{\oplus_{b'}|\psi_{b'}\rangle \;\mapsto\; |\psi_b\rangle} \mathcal{H}_b$$

“ the necessary becomes actual ”

adjoints

↔

$$\mathcal{H}_\bullet \xrightarrow[\textbf{possibility unit}]{\eta^{\Diamond_B}_{\mathcal{H}_\bullet}} \overbrace{(p_B)^*(p_B)_!}^{\Diamond_B}\mathcal{H}_\bullet$$

$$b : B \vdash \mathcal{H}_b \xrightarrow[\text{quantum }\textbf{state preparation}]{|\psi_b\rangle \;\mapsto\; \oplus_{b'}\begin{cases}|\psi_b\rangle & \text{if } b'=b \\ 0 & \text{else}\end{cases}} \bigoplus_{b':B}\mathcal{H}_{b'}$$

“ the actual is possible ”

The pure effects of these modalities of dependent linear data type formation

are remarkable in their sheer quantum information-theoretic content.

To repeat:

$$\overbrace{(p_B)^*(p_B)_*}^{\Box_B}\mathcal{H}_\bullet \xrightarrow[\textbf{necessity counit}]{\epsilon^{\Box_B}_{\mathcal{H}_\bullet}} \mathcal{H}_\bullet$$

$$b : B \vdash \bigoplus_{b':B}\mathcal{H}_{b'} \xrightarrow[\text{quantum } \textbf{measurement}]{\oplus_{b'}|\psi_{b'}\rangle \;\mapsto\; |\psi_b\rangle} \mathcal{H}_b$$

“ the necessary becomes actual ”

$$\mathcal{H}_\bullet \xrightarrow[\textbf{possibility unit}]{\eta^{\Diamond_B}_{\mathcal{H}_\bullet}} \overbrace{(p_B)^*(p_B)_!}^{\Diamond_B}\mathcal{H}_\bullet$$

$$b : B \vdash \mathcal{H}_b \xrightarrow[\text{quantum } \textbf{state preparation}]{|\psi_b\rangle \;\mapsto\; \oplus_{b'}\left\{\begin{array}{ll}|\psi_b\rangle & \text{if } b'=b \\ 0 & \text{else}\end{array}\right.} \bigoplus_{b':B}\mathcal{H}_{b'}$$

“ the actual is possible ”

$$\xleftrightarrow{\text{adjoints}}$$

“ the random becomes potential ”

$$\overbrace{(p_B)_!(p_B)^*}^{\star_B}\mathcal{H} \xhookrightarrow[\textbf{randomness counit}]{\epsilon^{\star_B}_{\mathcal{H}}} \mathcal{H}$$

$$\bigoplus_{b:B}\mathcal{H} \xrightarrow[\text{quantum } \textbf{superposition}]{\oplus_b|\psi_b\rangle \;\mapsto\; \Sigma_b|\psi_b\rangle} \mathcal{H}$$

The pure effects of these modalities of dependent linear data type formation

are remarkable in their sheer quantum information-theoretic content.

To repeat:

$$\overbrace{(p_B)^*(p_B)_*}^{\Box_B}\mathcal{H}_\bullet \xrightarrow[\textbf{necessity counit}]{\epsilon^{\Box_B}_{\mathcal{H}_\bullet}} \mathcal{H}_\bullet$$

$$b : B \vdash \bigoplus_{b':B}\mathcal{H}_{b'} \xrightarrow[\text{quantum }\textbf{measurement}]{\oplus_{b'}|\psi_{b'}\rangle \;\mapsto\; |\psi_b\rangle} \mathcal{H}_b$$

“ the necessary becomes actual ”

$$\mathcal{H}_\bullet \xrightarrow[\textbf{possibility unit}]{\eta^{\Diamond_B}_{\mathcal{H}_\bullet}} \overbrace{(p_B)^*(p_B)_!}^{\Diamond_B}\mathcal{H}_\bullet$$

$$b : B \vdash \mathcal{H}_b \xrightarrow[\text{quantum }\textbf{state preparation}]{|\psi_b\rangle \;\mapsto\; \oplus_{b'}\begin{cases}|\psi_b\rangle & \text{if } b'{=}b\\ 0 & \text{else}\end{cases}} \bigoplus_{b':B}\mathcal{H}_{b'}$$

“ the actual is possible ”

$\longleftrightarrow$ adjoints

“ the random becomes potential ”

$$\overbrace{(p_B)_!(p_B)^*}^{\star_B}\mathcal{H} \xhookrightarrow[\textbf{randomness counit}]{\epsilon^{\star_B}_{\mathcal{H}}} \mathcal{H}$$

$$\bigoplus_{b:B}\mathcal{H} \xrightarrow[\text{quantum }\textbf{superposition}]{\oplus_b|\psi_b\rangle \;\mapsto\; \Sigma_b|\psi_b\rangle} \mathcal{H}$$

“ the potential is inderminate ”

$$\mathcal{H} \xrightarrow[\textbf{indeterminacy unit}]{\eta^{\bigcirc_B}_{\mathcal{H}}} \overbrace{(p_B)_*(p_B)^*}^{\bigcirc_B}\mathcal{H}$$

$$\mathcal{H} \xrightarrow[\text{quantum }\textbf{parallelism}]{|\psi\rangle \;\mapsto\; \oplus_b|\psi\rangle_b} \bigoplus_{b:B}\mathcal{H}$$

Q-bits are the free linear indeterminacy-effect handlers over $\mathrm{Bit} = \{0, 1\}$

Coherent q-bits:

$$\begin{tikzcd} \mathrm{QBit} \arrow[d, equal, "{:}"'] & : & \mathrm{LType} \arrow[r, hook, "\mathbb{1}_{\mathrm{Bit}}\otimes"] & \mathrm{LType}_{\mathrm{Bit}} \arrow[r, "\oplus_{\mathrm{Bit}}", "\sim"'] & \mathrm{LType}^{\bigcirc_B} \\ \bigcirc_{\mathrm{Bit}}\mathbb{1} \end{tikzcd}$$

De-cohered (measured) q-bits:

Quantum gate with q-bit output:

Q-bits are the free linear indeterminacy-effect handlers over $\mathrm{Bit} = \{0, 1\}$

Coherent q-bits:

$$\begin{tikzcd}
\mathrm{QBit} \arrow[d, equal, ":"] & : \; \mathrm{LType} \arrow[r, hook, "\mathbb{1}_{\mathrm{Bit}}\otimes"] & \mathrm{LType}_{\mathrm{Bit}} \arrow[r, "\bigoplus_{\mathrm{Bit}}", "\sim"'] & \mathrm{LType}^{\bigcirc_B} \\
\bigcirc_{\mathrm{Bit}}\mathbb{1} \; = \; \bigoplus_{\{0,1\}}\mathbb{C} \; = \; \mathbb{C}\cdot|0\rangle \oplus \mathbb{C}\cdot|1\rangle & & &
\end{tikzcd}$$

Quantum gate with q-bit output:

De-cohered (measured) q-bits:

Q-bits are the free linear indeterminacy-effect handlers over $\mathrm{Bit} = \{0,1\}$

Coherent q-bits:

———

$$\begin{array}{l} \mathrm{QBit} \;:\; \mathrm{LType} \xhookrightarrow{\mathbb{1}_{\mathrm{Bit}}\otimes} \mathrm{LType}_{\mathrm{Bit}} \underset{\sim}{\xrightarrow{\oplus_{\mathrm{Bit}}}} \mathrm{LType}^{\bigcirc_B} \\ \quad\overset{\cdot\cdot}{\|} \\ \bigcirc_{\mathrm{Bit}}\mathbb{1} \;=\; \bigoplus_{\{0,1\}}\mathbb{C} \;=\; \mathbb{C}\cdot|0\rangle \oplus \mathbb{C}\cdot|1\rangle \end{array}$$

Quantum gate with q-bit output:

De-cohered (measured) q-bits:

Q-bits are the free linear indeterminacy-effect handlers over $\mathrm{Bit} = \{0,1\}$

Coherent q-bits:

$$\mathrm{QBit} \;:\; \mathrm{LType} \xhookrightarrow{\mathbb{1}_{\mathrm{Bit}}\otimes} \mathrm{LType}_{\mathrm{Bit}} \xrightarrow[\sim]{\bigoplus_{\mathrm{Bit}}} \mathrm{LType}^{\bigcirc_B}$$

$$\overset{\cdot\cdot}{\Vert}$$

$$\bigcirc_{\mathrm{Bit}}\mathbb{1} \;=\; \bigoplus_{\{0,1\}}\mathbb{C} \;=\; \mathbb{C}\cdot|0\rangle \oplus \mathbb{C}\cdot|1\rangle$$

$$\mathrm{QBit}$$

$$\otimes$$

$$\mathcal{H}$$

$$\Vert$$

$$\bigcirc_{\mathrm{Bit}}\mathcal{H} \;=\; \bigoplus_{\{0,1\}}\mathcal{H} \;=\; \mathcal{H}\otimes|0\rangle \;\oplus\; \mathcal{H}\otimes|0\rangle$$

De-cohered (measured) q-bits:

Quantum gate with q-bit output:

Q-bits are the free linear indeterminacy-effect handlers over $\mathrm{Bit} = \{0, 1\}$

Coherent q-bits:

$$\begin{tikzcd}
\mathrm{QBit} \, : \, \mathrm{LType} \arrow[r, hook, "\mathbb{1}_{\mathrm{Bit}}\otimes"] & \mathrm{LType}_{\mathrm{Bit}} \arrow[r, "\bigoplus_{\mathrm{Bit}}", "\sim"'] & \mathrm{LType}^{\bigcirc_B}
\end{tikzcd}$$

$$\overset{..}{\Vert}$$

$$\bigcirc_{\mathrm{Bit}} \mathbb{1} \;=\; \bigoplus_{\{0,1\}} \mathbb{C} \;=\; \mathbb{C} \cdot |0\rangle \oplus \mathbb{C} \cdot |1\rangle$$

$$\mathrm{QBit}$$

$$\otimes$$

$$\mathcal{H}$$

$$\Vert$$

$$\bigcirc_{\mathrm{Bit}} \mathcal{H} \;=\; \bigoplus_{\{0,1\}} \mathcal{H} \;=\; \mathcal{H} \otimes |0\rangle \;\oplus\; \mathcal{H} \otimes |0\rangle$$

De-cohered (measured) q-bits:

$$\begin{tikzcd}
\mathbb{1}_{\mathrm{Bit}} \, : \, \mathrm{LType}_{\mathrm{Bit}} \arrow[r, "\bigoplus_{\mathrm{Bit}}", "\sim"'] & \mathrm{LType}^{\bigcirc_{\mathrm{Bit}}}
\end{tikzcd}$$

$$b : \mathrm{Bit} \quad \vdash \quad \mathbb{C} \cdot |b\rangle \, : \, \mathrm{LType}$$

Quantum gate with q-bit output:

Q-bits are the free linear indeterminacy-effect handlers over $\mathrm{Bit} = \{0,1\}$

Coherent q-bits:

$$\mathrm{QBit} \;:\; \mathrm{LType} \xhookrightarrow{\mathbb{1}_{\mathrm{Bit}}{}^{\otimes}} \mathrm{LType}_{\mathrm{Bit}} \underset{\sim}{\xrightarrow{\bigoplus_{\mathrm{Bit}}}} \mathrm{LType}^{\bigcirc_B}$$

$$\overset{\cdot\cdot}{\|}$$

$$\bigcirc_{\mathrm{Bit}} \mathbb{1} \;=\; \bigoplus_{\{0,1\}} \mathbb{C} \;=\; \mathbb{C}\cdot|0\rangle \oplus \mathbb{C}\cdot|1\rangle$$

$$\mathrm{QBit}$$

$$\otimes$$

$$\mathcal{H}$$

$$\|$$

$$\bigcirc_{\mathrm{Bit}} \mathcal{H} \;=\; \bigoplus_{\{0,1\}} \mathcal{H} \;=\; \mathcal{H}\otimes|0\rangle \;\oplus\; \mathcal{H}\otimes|0\rangle$$

De-cohered (measured) q-bits:

$$\mathbb{1}_{\mathrm{Bit}} \;:\; \mathrm{LType}_{\mathrm{Bit}} \underset{\sim}{\xrightarrow{\bigoplus_{\mathrm{Bit}}}} \mathrm{LType}^{\bigcirc_{\mathrm{Bit}}}$$

$$b : \mathrm{Bit} \quad \vdash \quad \mathbb{C}\cdot|b\rangle \;:\; \mathrm{LType}$$

$$\mathbb{1}_{\mathrm{Bit}}$$

$$\otimes \qquad b : \mathrm{Bit} \quad \vdash \quad \mathcal{H}\otimes|b\rangle \;:\; \mathrm{LType}$$

$$\mathcal{H}$$

Quantum gate with q-bit output:

Q-bits are the free linear indeterminacy-effect handlers over $\mathrm{Bit} = \{0,1\}$

Coherent q-bits:

$$\mathrm{QBit} : \mathrm{LType} \xhookrightarrow{\mathbb{1}_{\mathrm{Bit}}\otimes} \mathrm{LType}_{\mathrm{Bit}} \xrightarrow[\sim]{\oplus_{\mathrm{Bit}}} \mathrm{LType}^{\bigcirc_B}$$

$$\overset{\cdot\cdot}{\shortparallel}$$

$$\bigcirc_{\mathrm{Bit}}\mathbb{1} = \bigoplus_{\{0,1\}}\mathbb{C} = \mathbb{C}\cdot|0\rangle \oplus \mathbb{C}\cdot|1\rangle$$

$$\mathrm{QBit}$$

$$\otimes$$

$$\mathcal{H}$$

$$\shortparallel$$

$$\bigcirc_{\mathrm{Bit}}\mathcal{H} = \bigoplus_{\{0,1\}}\mathcal{H} = \mathcal{H}\otimes|0\rangle \oplus \mathcal{H}\otimes|0\rangle$$

De-cohered (measured) q-bits:

$$\mathbb{1}_{\mathrm{Bit}} : \mathrm{LType}_{\mathrm{Bit}} \xrightarrow[\sim]{\oplus_{\mathrm{Bit}}} \mathrm{LType}^{\bigcirc_{\mathrm{Bit}}}$$

$$b : \mathrm{Bit} \quad \vdash \quad \mathbb{C}\cdot|b\rangle : \mathrm{LType}$$

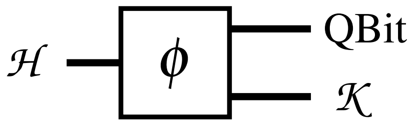
$$\mathbb{1}_{\mathrm{Bit}}$$

$$\otimes \qquad b : \mathrm{Bit} \quad \vdash \quad \mathcal{H}\otimes|b\rangle : \mathrm{LType}$$

$$\mathcal{H}$$

Quantum gate with q-bit output:

A quantum gate which
may handle $\bigcirc_{\mathrm{Bit}}$-effects
is one with a QBit-output:



$$\mathcal{H} \xrightarrow{\phi} \mathrm{QBit}\otimes\mathcal{K} \simeq \bigcirc_{\mathrm{Bit}}\mathcal{K}$$

Q-bits are the free linear indeterminacy-effect handlers over $\mathrm{Bit} = \{0,1\}$

Coherent q-bits:

$$\mathrm{QBit} \;:\; \mathrm{LType} \xhookrightarrow{\mathbb{1}_{\mathrm{Bit}}\otimes} \mathrm{LType}_{\mathrm{Bit}} \xrightarrow[\sim]{\bigoplus_{\mathrm{Bit}}} \mathrm{LType}^{\bigcirc_B}$$

$$\overset{..}{||}$$

$$\bigcirc_{\mathrm{Bit}}\mathbb{1} \;=\; \bigoplus_{\{0,1\}}\mathbb{C} \;=\; \mathbb{C}\cdot|0\rangle \oplus \mathbb{C}\cdot|1\rangle$$

$$\mathrm{QBit}$$
$$\otimes$$
$$\mathcal{H}$$
$$||$$
$$\bigcirc_{\mathrm{Bit}}\mathcal{H} \;=\; \bigoplus_{\{0,1\}}\mathcal{H} \;=\; \mathcal{H}\otimes|0\rangle \;\oplus\; \mathcal{H}\otimes|0\rangle$$

De-cohered (measured) q-bits:

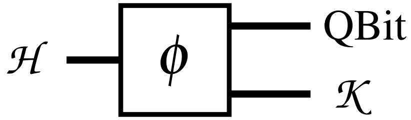
$$\mathbb{1}_{\mathrm{Bit}} \;:\; \mathrm{LType}_{\mathrm{Bit}} \xrightarrow[\sim]{\bigoplus_{\mathrm{Bit}}} \mathrm{LType}^{\bigcirc_{\mathrm{Bit}}}$$

$$b : \mathrm{Bit} \quad\vdash\quad \mathbb{C}\cdot|b\rangle \;:\; \mathrm{LType}$$

$$\mathbb{1}_{\mathrm{Bit}}$$
$$\otimes \qquad b : \mathrm{Bit} \quad\vdash\quad \mathcal{H}\otimes|b\rangle \;:\; \mathrm{LType}$$
$$\mathcal{H}$$

Quantum gate with q-bit output:

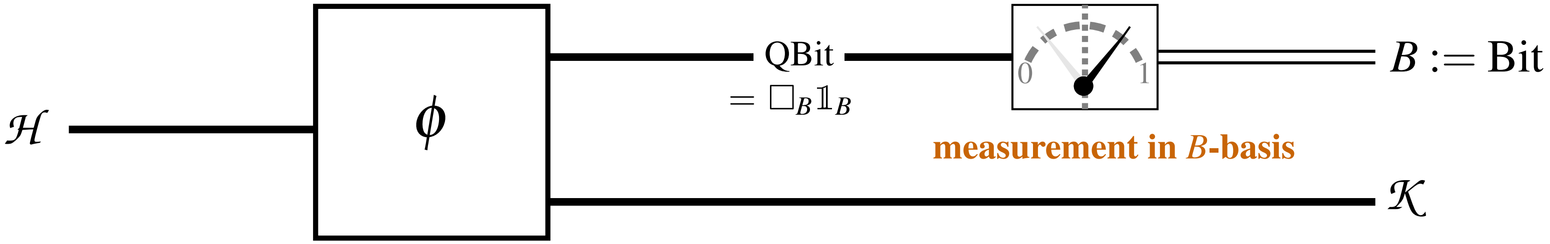
A quantum gate which
may handle $\bigcirc_{\mathrm{Bit}}$-effects
is one with a QBit-output:



$$\mathcal{H} \xrightarrow{\phi} \mathrm{QBit}\otimes\mathcal{K} \simeq \bigcirc_{\mathrm{Bit}}\mathcal{K}$$

Quantum measurement is Linear indefiniteness-effect handling.

quantum circuit



quantum gate

$$\mathcal{H} \xrightarrow{\phi} \mathrm{QBit} \otimes \mathcal{K} \simeq \bigcirc_B \mathcal{K}$$

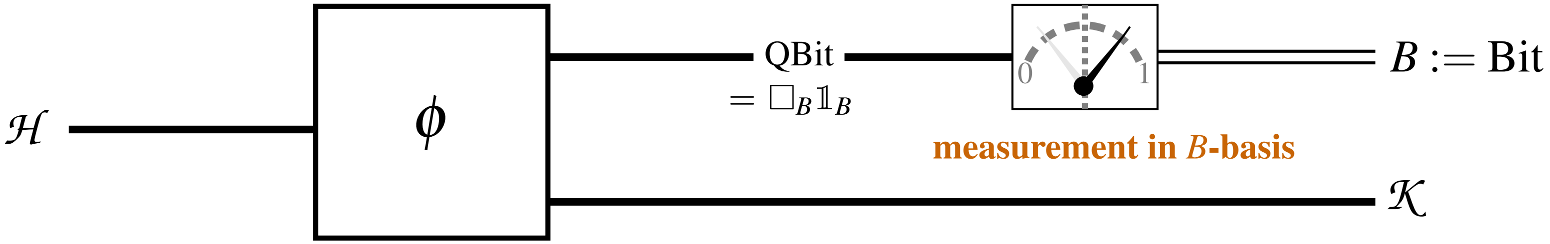
$\bigcirc_B$-effect handling

Quantum measurement is Linear indefiniteness-effect handling.

quantum circuit

$\bigcirc_B$-modal linear types

$\mathrm{LType}_{\bigcirc_B}$



measurement in B-basis

quantum gate

$$\mathcal{H} \xrightarrow{\phi} \mathrm{QBit} \otimes \mathcal{K} \simeq \bigcirc_B \mathcal{K}$$

$\bigcirc_B$-effect handling

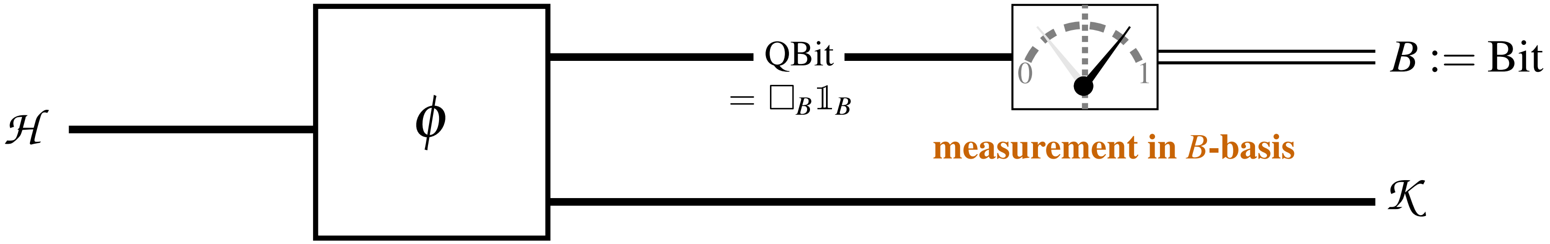
Quantum measurement is Linear indefiniteness-effect handling.

quantum circuit

$$\begin{tikzcd}
\text{quantum circuit} \arrow[d, squiggly, "\text{formalization}"] \\
\mathrm{LType}_{\bigcirc_B} \arrow[d, hook, "K_{(p_B)_*(p_B)^*}"', "\text{comparison functor}"] \\
\mathrm{LType}_B
\end{tikzcd}$$

$\bigcirc_B$-modal linear types

B-dependent linear types



measurement in B-basis

subjection to $\bigcirc_B$-effects

quantum gate

$$\begin{tikzcd}
\bigcirc_B \mathcal{H} \arrow[loop above, "\bigcirc_B"] \arrow[r, "\bigcirc_B\left(\mathcal{H} \xrightarrow{\phi} \mathrm{QBit} \otimes \mathcal{K} \simeq \bigcirc_B \mathcal{K}\right)"] & \bigcirc_B \bigcirc_B \mathcal{K} \arrow[loop above, "\bigcirc_B"] \arrow[r, "\mathrm{hndl}^{\bigcirc_B}_{\bigcirc_B \mathcal{K}}", "\bigcirc_B\text{-effect handling}"'] & \bigcirc_B \mathcal{K} \arrow[loop above, "\bigcirc_B"]
\end{tikzcd}$$

Quantum measurement is Linear indefiniteness-effect handling.

quantum circuit

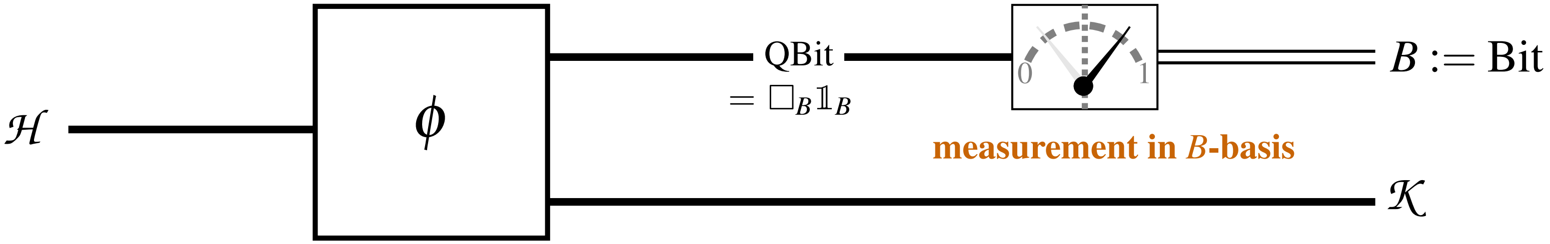
$$\begin{tikzcd} \text{quantum circuit} \arrow[d, squiggly, "\text{formalization}"] \\ \mathrm{LType}_{\bigcirc_B} \arrow[d, "K_{(p_B)^*(p_B)_*}"', "\text{comparison functor}"] \\ \mathrm{LType}_B \end{tikzcd}$$

$\bigcirc_B$-modal linear types

$\mathrm{LType}_{\bigcirc_B}$

LType_B

B-dependent linear types



subjection to $\bigcirc_B$-effects

quantum gate

$$\begin{tikzcd} \bigcirc_B \mathcal{H} \arrow[loop above, "\bigcirc_B"] \arrow[rr, "\bigcirc_B\left(\mathcal{H}\xrightarrow{\phi}\mathrm{QBit}\otimes\mathcal{K}\simeq\bigcirc_B\mathcal{K}\right)"] && \bigcirc_B\bigcirc_B\mathcal{K} \arrow[loop above, "\bigcirc_B"] \arrow[rr, "\mathrm{hndl}^{\bigcirc_B}_{\bigcirc_B\mathcal{K}}", "\bigcirc_B\text{-effect handling}"'] && \bigcirc_B\mathcal{K} \arrow[loop above, "\bigcirc_B"] \end{tikzcd}$$

$$\begin{tikzcd} \mathbb{1}_B \otimes \mathcal{H} \arrow[rr, "\mathbb{1}_B\otimes\phi"] && \square_B \mathbb{1}_B \otimes \mathcal{K} \arrow[rr, "\epsilon^{\square_B}_{\mathbb{1}_B}\otimes\mathcal{K}", "\square_B\text{-counit}"'] && \mathbb{1}_B \otimes \mathcal{K} \end{tikzcd}$$

$$\begin{tikzcd} b : B \vdash \mathcal{H} \arrow[rr, "\phi"] && \mathrm{QBit}\otimes\mathcal{K} \arrow[rr, "\text{quantum state collapse}"] && \mathcal{K} \end{tikzcd}$$

measurement result

$$\begin{array}{c} |0\rangle\otimes|\psi_0\rangle \\ +|1\rangle\otimes|\psi_1\rangle \end{array} \mapsto |\psi_b\rangle$$

Quantum measurement is Linear indefiniteness-effect handling.

quantum circuit

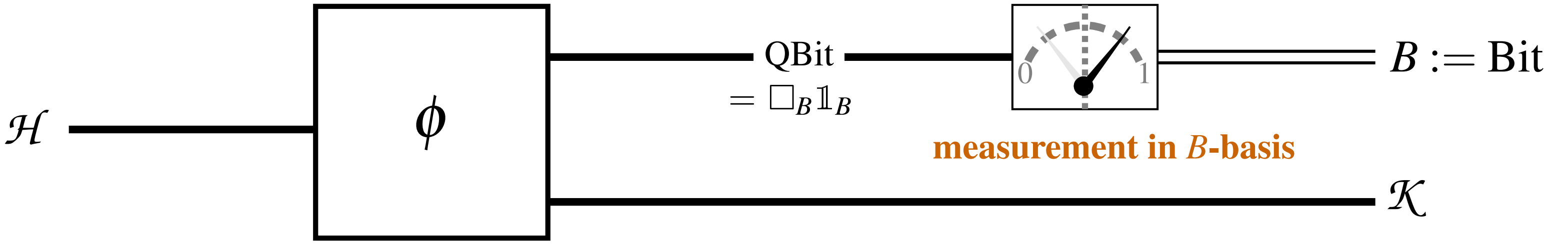
$$\begin{tikzcd}
\text{quantum circuit} \arrow[d, squiggly, "\text{formalization}"] \\
\mathrm{LType}_{\bigcirc_B} \arrow[d, "\text{comparison functor}", "K_{(p_B)^*(p_B)_*}"'] \\
\mathrm{LType}_B
\end{tikzcd}$$

$\bigcirc_B$-modal linear types

$\mathrm{LType}_{\bigcirc_B}$

LType_B

***B*-dependent linear types**



measurement in *B*-basis

$$\begin{tikzcd}
\bigcirc_B \mathcal{H} \arrow[loop above, "\bigcirc_B"] \arrow[r, "\bigcirc_B\left(\mathcal{H} \xrightarrow{\phi} \mathrm{QBit} \otimes \mathcal{K} \simeq \bigcirc_B \mathcal{K}\right)"] & \bigcirc_B \bigcirc_B \mathcal{K} \arrow[loop above, "\bigcirc_B"] \arrow[r, "\mathrm{hndl}^{\bigcirc_B}_{\bigcirc_B \mathcal{K}}", "\bigcirc_B\text{-effect handling}"'] & \bigcirc_B \mathcal{K} \arrow[loop above, "\bigcirc_B"]
\end{tikzcd}$$

subjection to $\bigcirc_B$-effects

quantum gate

$$\begin{tikzcd}
\mathbb{1}_B \otimes \mathcal{H} \arrow[r, "\mathbb{1}_B \otimes \phi"] & \Box_B \mathbb{1}_B \otimes \mathcal{K} \arrow[r, "\epsilon^{\Box_B}_{\mathbb{1}_B} \otimes \mathcal{K}", "\Box_B\text{-counit}"'] & \mathbb{1}_B \otimes \mathcal{K}
\end{tikzcd}$$

$$\begin{tikzcd}
b : B \vdash \quad \mathcal{H} \arrow[r, "\phi"] & \mathrm{QBit} \otimes \mathcal{K} \arrow[r, "\text{quantum state collapse}"] & \mathcal{K}
\end{tikzcd}$$

measurement result

$$\begin{array}{c} |0\rangle \otimes |\psi_0\rangle \\ + |1\rangle \otimes |\psi_1\rangle \end{array} \mapsto |\psi_b\rangle$$

full linearly-typed detail of
quantum measurement logic
is emergent effect in LHoTT

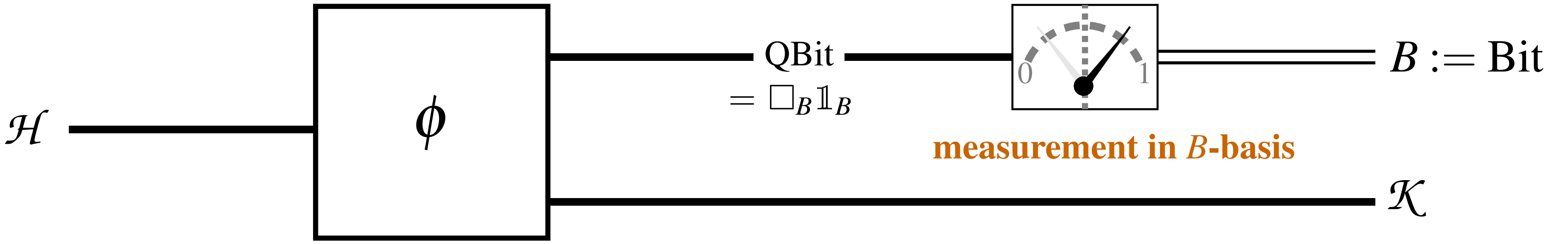
Quantum measurement is Linear indefiniteness-effect handling.

quantum circuit

$$\begin{tikzcd}
\text{quantum circuit} \arrow[d, rightsquigarrow, "\text{formalization}"] \\
\text{LType}_{\bigcirc_B} \arrow[d, "K_{(p_B)^*(p_B)_*}"', "\text{comparison functor}"] \\
\text{LType}_B
\end{tikzcd}$$

$\bigcirc_B$-modal linear types

B-dependent linear types



measurement in B-basis

$$\begin{tikzcd}
\bigcirc_B \mathcal{H} \arrow[loop above, "\bigcirc_B"] \arrow[rr, "{\bigcirc_B\left(\mathcal{H} \xrightarrow{\phi} \text{QBit} \otimes \mathcal{K} \simeq \bigcirc_B \mathcal{K}\right)}"] & & \bigcirc_B \bigcirc_B \mathcal{K} \arrow[loop above, "\bigcirc_B"] \arrow[rr, "\text{hndl}^{\bigcirc_B}_{\bigcirc_B \mathcal{K}}", "\bigcirc_B\text{-effect handling}"'] & & \bigcirc_B \mathcal{K} \arrow[loop above, "\bigcirc_B"]
\end{tikzcd}$$

subjection to $\bigcirc_B$-effects

quantum gate

$$\begin{tikzcd}
\mathbb{1}_B \otimes \mathcal{H} \arrow[rr, "\mathbb{1}_B \otimes \phi"] & & \Box_B \mathbb{1}_B \otimes \mathcal{K} \arrow[rr, "\epsilon^{\Box_B}_{\mathbb{1}_B} \otimes \mathcal{K}", "\Box_B\text{-counit}"'] & & \mathbb{1}_B \otimes \mathcal{K}
\end{tikzcd}$$

$$\begin{tikzcd}
b : B \vdash \mathcal{H} \arrow[rr, "\phi"] & & \text{QBit} \otimes \mathcal{K} \arrow[rr, "\text{quantum state collapse}"] & & \mathcal{K}
\end{tikzcd}$$

measurement result

$$\begin{array}{l} |0\rangle \otimes |\psi_0\rangle \\ + |1\rangle \otimes |\psi_1\rangle \end{array} \mapsto |\psi_b\rangle$$

full linearly-typed detail of
quantum measurement logic
is emergent effect in LHoTT

Aside: **Linear indefiniteness monad recovers Coecke's "classical structures".**

(see nLab:quantum+reader+monad)

$$\bigcirc_B$$

$$\pitchfork$$

$$\mathrm{Monad}\big(\mathrm{LType}\big)$$

Aside: **Linear indefiniteness monad recovers Coecke's "classical structures".**

(see nLab:quantum+reader+monad)

$$\begin{array}{c}
\bigcirc_B \\
\rotatebox[origin=c]{-90}{$=:$} \\
B\text{-Reader} \\
\\
\\
\rotatebox[origin=c]{90}{$\in$} \\
\mathrm{Monad}\big(\mathrm{LType}\big)
\end{array}$$

Aside: **Linear indefiniteness monad recovers Coecke's "classical structures".**

(see nLab:quantum+reader+monad)

$$B : \mathrm{FinType} \;\; \vdash \qquad \begin{tikzcd} \bigcirc_B \arrow[d, phantom, "\overset{!}{=}" rotate=90] \\ B\text{-Reader} \arrow[d, phantom, "\wr\!|"] \\ \mathbb{1}^B\text{-Writer} \arrow[d, phantom, "\Cap"] \\ \mathrm{Monad}\big(\mathrm{LType}\big) \end{tikzcd}$$

$$\mathbb{1}^B\text{-Writer}(D) := \mathbb{1}^B \oplus D$$

$$\mathrm{bind}^{\mathbb{1}^B\mathrm{Writer}}\Big(D_1 \xrightarrow{\mathrm{prog}} \mathbb{1}^B \oplus D_2\Big) :=$$

$$\mathbb{1}^B \oplus D_1 \xrightarrow{\mathbb{1}^B \oplus \mathrm{prog}} \mathbb{1}^B \oplus \mathbb{1}^B \oplus D_2 \xrightarrow{\mu \oplus \mathrm{id}_{D_2}} \mathbb{1}^B \oplus D_2$$

Where $\mathbb{1}^B = \bigoplus\limits_{b:B} \mathbb{C} \cdot P_b \in \mathrm{CMon}\big(\mathrm{LType}\big)$ is algebra of B-projection operators :

unit

$$\mathbb{1} \xrightarrow[\; 1 \mapsto \sum\limits_{b:B} P_b \;]{\eta} \mathbb{1}^B$$

product

$$\mathbb{1}^B \otimes \mathbb{1}^B \xrightarrow[\; P_b \otimes P_{b'} \mapsto \left\{ \begin{array}{l} P_b \text{ if } b=b' \\ 0 \text{ else} \end{array} \right. \;]{\mu} \mathbb{1}^B$$

Aside: **Linear indefiniteness monad recovers Coecke's "classical structures".**

(see nLab:quantum+reader+monad)

$$\begin{tikzcd}
 & & \bigcirc_B \arrow[d, phantom, "\shortparallel"] \\
 & & B\text{-Reader} \arrow[d, phantom, "\wr"] \\
 & & \mathbb{1}^B\text{-Writer} \\
B : \mathrm{FinType} \;\; \vdash & \mathbb{1}^B\text{-DualWriter} \arrow[ur, phantom, "\simeq"] & \\
 & & \mathrm{Monad}(\mathrm{LType}) \arrow[ul, phantom, "\subset"] \\
 & \mathrm{FrobMonad}(\mathrm{LType}) \arrow[ur] &
\end{tikzcd}$$

$$\mathbb{1}^B\text{-Writer}(D) := \mathbb{1}^B \oplus D$$

$$\mathrm{bind}^{\mathbb{1}^B\mathrm{Writer}}\Big(D_1 \xrightarrow{\mathrm{prog}} \mathbb{1}^B \oplus D_2\Big) := \mathbb{1}^B \oplus D_1 \xrightarrow{\mathbb{1}^B \oplus \mathrm{prog}} \mathbb{1}^B \oplus \mathbb{1}^B \oplus D_2 \xrightarrow{\mu \oplus \mathrm{id}_{D_2}} \mathbb{1}^B \oplus D_2$$

Where $\mathbb{1}^B = \bigoplus_{b:B} \mathbb{C} \cdot P_b \in \mathrm{CMon}(\mathrm{LType})$ is Frobenius algebra of B-projection operators :

$$\mathbb{1} \xrightarrow[1 \mapsto \sum\limits_{b:B} P_b]{\substack{\textbf{unit} \\ \eta}} \mathbb{1}^B \xrightarrow[P_b \mapsto P_b \otimes P_b]{\substack{\textbf{co-product} \\ \delta}} \mathbb{1}^B \otimes \mathbb{1}^B \xrightarrow[P_b \otimes P_{b'} \mapsto \left\{\begin{array}{l} P_b \text{ if } b=b' \\ 0 \text{ else} \end{array}\right.]{\substack{\textbf{product} \\ \mu}} \mathbb{1}^B \xrightarrow[P_b \mapsto 1]{\substack{\textbf{co-unit} \\ \varepsilon}} \mathbb{1}$$

Aside: **Linear indefiniteness monad recovers Coecke's "classical structures".**

(see nLab:quantum+reader+monad)

$$B : \mathrm{FinType} \;\; \vdash$$

$$\begin{tikzcd}
 & & \bigcirc_B \arrow[d, phantom, "\shortparallel"] \\
 & & B\text{-Reader} \arrow[d, phantom, "\wr"] \\
\mathbb{1}^B\text{-CoWriter} & & \mathbb{1}^B\text{-Writer} \\
 & \mathbb{1}^B\text{-DualWriter} \arrow[ul, phantom, "\simeq"] \arrow[ur, phantom, "\simeq"] \arrow[d, phantom, "\in"] & \\
\mathrm{CoMonad}\big(\mathrm{LType}\big) & & \mathrm{Monad}\big(\mathrm{LType}\big) \\
 & \mathrm{FrobMonad}\big(\mathrm{LType}\big) \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

$$\mathbb{1}^B\text{-Writer}(D) :\equiv \mathbb{1}^B \oplus D$$

$$\mathrm{bind}^{\mathbb{1}^B\mathrm{Writer}}\Big(D_1 \xrightarrow{\mathrm{prog}} \mathbb{1}^B \oplus D_2\Big) :\equiv \mathbb{1}^B \oplus D_1 \xrightarrow{\mathbb{1}^B \oplus \mathrm{prog}} \mathbb{1}^B \oplus \mathbb{1}^B \oplus D_2 \xrightarrow{\mu \oplus \mathrm{id}_{D_2}} \mathbb{1}^B \oplus D_2$$

Where $\mathbb{1}^B = \bigoplus_{b:B} \mathbb{C} \cdot P_b \;\in\; \mathrm{CMon}(\mathrm{LType})$ is Frobenius algebra of B-projection operators :

$$\mathbb{1} \xrightarrow[1 \mapsto \sum_{b:B} P_b]{\overset{\textbf{unit}}{\eta}} \mathbb{1}^B \xrightarrow[P_b \mapsto P_b \otimes P_b]{\overset{\textbf{co-product}}{\delta}} \mathbb{1}^B \otimes \mathbb{1}^B \xrightarrow[P_b \otimes P_{b'} \mapsto \begin{cases} P_b & \text{if } b=b' \\ 0 & \text{else} \end{cases}]{\overset{\textbf{product}}{\mu}} \mathbb{1}^B \xrightarrow[P_b \mapsto 1]{\overset{\textbf{co-unit}}{\varepsilon}} \mathbb{1}$$

Aside: **Linear indefiniteness monad recovers Coecke's "classical structures".**

(see nLab:quantum+reader+monad)

$$\begin{tikzcd}[column sep=small]
 & & & \bigcirc_B \\
 & B\text{-CoReader} \arrow[d, phantom, "\wr|" description] & & B\text{-Reader} \arrow[u, phantom, "\overset{!}{=}" description] \arrow[d, phantom, "\wr|" description] \\
 & \mathbb{1}^B\text{-CoWriter} & & \mathbb{1}^B\text{-Writer} \\
B : \mathrm{FinType} \;\vdash & & \mathbb{1}^B\text{-DualWriter} \arrow[ul, phantom, "\simeq" sloped] \arrow[ur, phantom, "\simeq" sloped] & \\
 & \mathrm{CoMonad}\big(\mathrm{LType}\big) & \arrow[u, phantom, "\in" sloped] & \mathrm{Monad}\big(\mathrm{LType}\big) \\
 & & \mathrm{FrobMonad}\big(\mathrm{LType}\big) \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

$$\mathbb{1}^B\text{-Writer}(D) :\equiv \mathbb{1}^B \oplus D$$

$$\mathrm{bind}^{\mathbb{1}^B\mathrm{Writer}}\Big(D_1 \xrightarrow{\mathrm{prog}} \mathbb{1}^B \oplus D_2\Big) :\equiv \mathbb{1}^B \oplus D_1 \xrightarrow{\mathbb{1}^B \oplus \mathrm{prog}} \mathbb{1}^B \oplus \mathbb{1}^B \oplus D_2 \xrightarrow{\mu \oplus \mathrm{id}_{D_2}} \mathbb{1}^B \oplus D_2$$

Where $\mathbb{1}^B = \bigoplus_{b:B} \mathbb{C} \cdot P_b \in \mathrm{CMon}(\mathrm{LType})$ is Frobenius algebra of B-projection operators :

$$\mathbb{1} \xrightarrow[1 \,\mapsto\, \sum_{b:B} P_b]{\overset{\textbf{unit}}{\eta}} \mathbb{1}^B \xrightarrow[P_b \,\mapsto\, P_b \otimes P_b]{\overset{\textbf{co-product}}{\delta}} \mathbb{1}^B \otimes \mathbb{1}^B \xrightarrow[P_b \otimes P_{b'} \,\mapsto\, \begin{cases} P_b \text{ if } b = b' \\ 0 \text{ else} \end{cases}]{\overset{\textbf{product}}{\mu}} \mathbb{1}^B \xrightarrow[P_b \,\mapsto\, 1]{\overset{\textbf{co-unit}}{\varepsilon}} \mathbb{1}$$

Aside: **Linear indefiniteness monad recovers Coecke's "classical structures".**

(see nLab:quantum+reader+monad)

$$\begin{tikzcd}
& \star_B \arrow[d, phantom, "\rotatebox{-90}{$=:$}"] & & \bigcirc_B \arrow[d, phantom, "\rotatebox{-90}{$=:$}"] \\
& B\text{-CoReader} \arrow[d, phantom, "\wr"] & & B\text{-Reader} \arrow[d, phantom, "\wr"] \\
& \mathbb{1}^B\text{-CoWriter} \arrow[dr, phantom, "\simeq"] & & \mathbb{1}^B\text{-Writer} \arrow[dl, phantom, "\simeq"] \\
B : \text{FinType} \;\; \vdash & & \mathbb{1}^B\text{-DualWriter} \arrow[dd, phantom, "\rotatebox{90}{$\in$}"] & \\
& \text{CoMonad}(\text{LType}) & & \text{Monad}(\text{LType}) \\
& & \text{FrobMonad}(\text{LType}) \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

$\mathbb{1}^B\text{-Writer}(D) :\equiv \mathbb{1}^B \oplus D$

$\text{bind}^{\mathbb{1}^B\text{Writer}}\Big(D_1 \xrightarrow{\text{prog}} \mathbb{1}^B \oplus D_2\Big) :\equiv$

$\mathbb{1}^B \oplus D_1 \xrightarrow{\mathbb{1}^B \oplus \text{prog}} \mathbb{1}^B \oplus \mathbb{1}^B \oplus D_2 \xrightarrow{\mu \oplus \text{id}_{D_2}} \mathbb{1}^B \oplus D_2$

Where $\mathbb{1}^B = \bigoplus_{b:B} \mathbb{C} \cdot P_b \;\in\; \text{CMon}(\text{LType})$ is Frobenius algebra of B-projection operators :

$$\begin{tikzcd}[column sep=huge]
\mathbb{1} \arrow[r, "\overset{\textbf{unit}}{\eta}", "1 \mapsto \sum\limits_{b:B} P_b"'] & \mathbb{1}^B \arrow[r, "\overset{\textbf{co-product}}{\delta}", "P_b \mapsto P_b \otimes P_b"'] & \mathbb{1}^B \otimes \mathbb{1}^B \arrow[r, "\overset{\textbf{product}}{\mu}", "P_b \otimes P_{b'} \mapsto \begin{cases} P_b \text{ if } b=b' \\ 0 \text{ else} \end{cases}"'] & \mathbb{1}^B \arrow[r, "\overset{\textbf{co-unit}}{\varepsilon}", "P_b \mapsto 1"'] & \mathbb{1}
\end{tikzcd}$$

Aside: **Linear indefiniteness monad recovers Coecke's "classical structures".**

(see nLab:quantum+reader+monad)

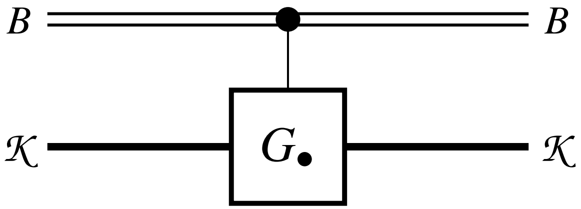
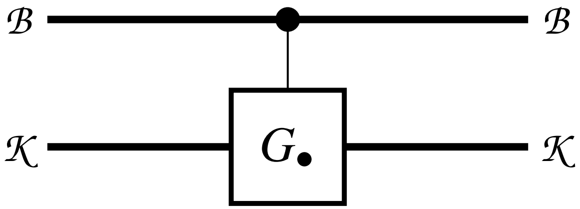
$$\begin{tikzcd}
 & \star_B \arrow[rr, equal] \arrow[dr, equal] & & \bigcirc_B \arrow[dl, equal] & \\
B : \mathrm{FinType} \;\; \vdash & & \mathbb{1}^B\text{-DualWriter} & & \\
 & & \pitchfork & & \\
 & \mathrm{CoMonad}\big(\mathrm{LType}\big) & & \mathrm{Monad}\big(\mathrm{LType}\big) & \\
 & & \mathrm{FrobMonad}\big(\mathrm{LType}\big) \arrow[ul] \arrow[ur] & &
\end{tikzcd}$$

Where $\mathbb{1}^B = \bigoplus_{b:B} \mathbb{C} \cdot P_b \;\in\; \mathrm{CMon}\big(\mathrm{LType}\big)$ is Frobenius algebra of B-projection operators :

$$\begin{tikzcd}[column sep=huge]
\mathbb{1} \arrow[r, "\overset{\textbf{unit}}{\eta}", "1 \mapsto \sum\limits_{b:B} P_b"'] & \mathbb{1}^B \arrow[r, "\overset{\textbf{co-product}}{\delta}", "P_b \mapsto P_b \otimes P_b"'] & \mathbb{1}^B \otimes \mathbb{1}^B \arrow[r, "\overset{\textbf{product}}{\mu}", "P_b \otimes P_{b'} \mapsto \begin{cases} P_b \text{ if } b{=}b' \\ 0 \text{ else} \end{cases}"'] & \mathbb{1}^B \arrow[r, "\overset{\textbf{co-unit}}{\varepsilon}", "P_b \mapsto 1"'] & \mathbb{1}
\end{tikzcd}$$

Exmp: Deferred measurement principle – Proven by monadic effect logic.

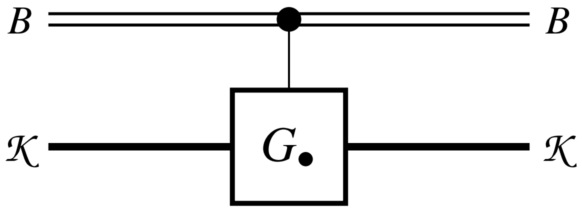
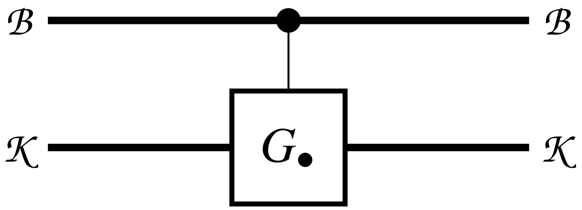
Deferred Measurement Principle

classically controlled gate	**quantumly controlled gate**
 $\mathcal{B}_\bullet \boxtimes \mathcal{K} \xrightarrow{G_\bullet} \mathcal{B}_\bullet \boxtimes \mathcal{K}$ $b : B \;\vdash\; \mathcal{K} \xrightarrow{G_b} \mathcal{K}$	 $\square_B \mathcal{B}_\bullet \boxtimes \mathcal{K} \xrightarrow{\square_B G_\bullet} \square_B \mathcal{B}_\bullet \boxtimes \mathcal{K}$ $b : B \;\vdash\; \bigoplus_{b' B} \mathcal{K} \xrightarrow{\bigoplus_{b' : B} G_{b'}} \bigoplus_{b' B} \mathcal{K}$

Exmp: Deferred measurement principle – Proven by monadic effect logic.

$$\Box_B \mathcal{H}_\bullet \xrightarrow{F} \Box_B \mathcal{H}_\bullet \xrightarrow{\varepsilon^{\Box_B}_{\mathcal{H}_\bullet}} \mathcal{H}_\bullet \xrightarrow{G_\bullet} \mathcal{H}_\bullet \;\mapsto\; \Box_B \mathcal{H}_\bullet \xrightarrow{\mathrm{diag}_B(G_\bullet)\circ F} \Box_B \mathcal{H}_\bullet \;\mapsto\; \Box_B \mathcal{H}_\bullet \xrightarrow{\mathrm{diag}_B(G_\bullet)\circ F} \Box_B \mathcal{H}_\bullet \xrightarrow{\varepsilon^{\Box_B}_{\mathcal{H}_\bullet}} \mathcal{H}_\bullet$$

measurement-controlled quantum gate — quantum-controlled quantum gate... — ...followed by measurement

classically controlled gate	**quantumly controlled gate**
 $\mathcal{B}_\bullet \boxtimes \mathcal{K} \xrightarrow{G_\bullet} \mathcal{B}_\bullet \boxtimes \mathcal{K}$ $b : B \;\vdash\; \mathcal{K} \xrightarrow{G_b} \mathcal{K}$	 $\Box_B \mathcal{B}_\bullet \boxtimes \mathcal{K} \xrightarrow{\Box_B G_\bullet} \Box_B \mathcal{B}_\bullet \boxtimes \mathcal{K}$ $b : B \;\vdash\; \bigoplus_{b' B} \mathcal{K} \xrightarrow{\bigoplus_{b' : B} G_{b'}} \bigoplus_{b' B} \mathcal{K}$

Exmp: Deferred measurement principle – Proven by monadic effect logic.

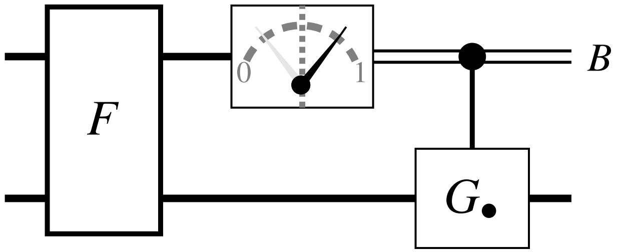
$$\begin{tikzcd} \mathrm{Kl}(\Box_B) \arrow[r, "\sim", "\delta^B \circ \Box_B(-)"'] \arrow[rr, bend left, "\mathrm{id}"] & \mathrm{LType}_{B\Box_B} \arrow[r, "\sim", "\varepsilon^{\Box_B} \circ (-)"'] & \mathrm{Kl}(\Box_B) \end{tikzcd}$$

Kleisli equivalence

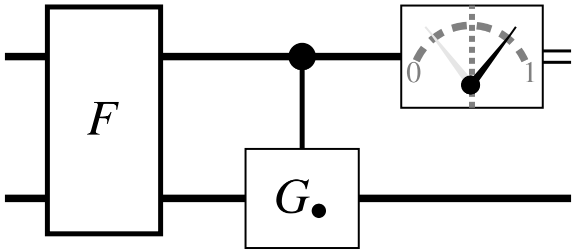
$\Box_B$-Kleisli morphisms | **$\Box_B$-coalgebra homomorphisms** | **$\Box_B$-Kleisli morphisms**

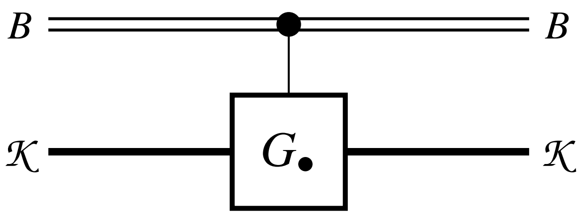
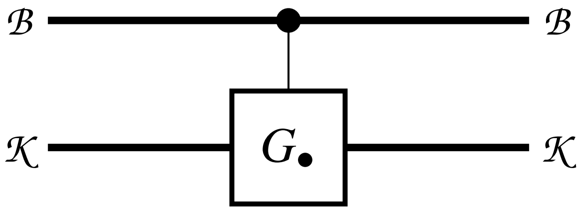
$$\Box_B \mathcal{H}_\bullet \xrightarrow{F} \Box_B \mathcal{H}_\bullet \xrightarrow{\varepsilon^{\Box_B}_{\mathcal{H}_\bullet}} \mathcal{H}_\bullet \xrightarrow{G_\bullet} \mathcal{H}_\bullet \quad \mapsto \quad \Box_B \mathcal{H}_\bullet \xrightarrow{\mathrm{diag}_B(G_\bullet)\circ F} \Box_B \mathcal{H}_\bullet \quad \mapsto \quad \Box_B \mathcal{H}_\bullet \xrightarrow{\mathrm{diag}_B(G_\bullet)\circ F} \Box_B \mathcal{H}_\bullet \xrightarrow{\varepsilon^{\Box_B}_{\mathcal{H}_\bullet}} \mathcal{H}_\bullet$$

measurement-controlled quantum gate | **quantum-controlled quantum gate...** | **...followed by measurement**



Deferred Measurement Principle ⟷



classically controlled gate	**quantumly controlled gate**
	
$\mathcal{B}_\bullet \boxtimes \mathcal{K} \xrightarrow{G_\bullet} \mathcal{B}_\bullet \boxtimes \mathcal{K}$	$\Box_B \mathcal{B}_\bullet \boxtimes \mathcal{K} \xrightarrow{\Box_B G_\bullet} \Box_B \mathcal{B}_\bullet \boxtimes \mathcal{K}$
$b : B \ \vdash \quad \mathcal{K} \xrightarrow{G_b} \mathcal{K}$	$b : B \ \vdash \quad \bigoplus_{b' B} \mathcal{K} \xrightarrow{\bigoplus_{b' : B} G_{b'}} \bigoplus_{b' B} \mathcal{K}$

The Quantum modality.

Also the *exponential modality* traditionally postulated in linear logic is an emergent effect in `LHoTT`,

linear randomization

aka: stabilization/motivization

$$\begin{tikzcd}
\mathrm{Type} \arrow[r, shift left=2, "\Sigma^\infty_+ \,:\, B \;\mapsto\; \star_B \mathbb{1}"] \arrow[r, phantom, "\perp"] & \mathrm{LType} \arrow[l, shift left=2, "\Omega^\infty"] \arrow[loop right, "!"]
\end{tikzcd}$$

exponential modality

classical types

linear types

The Quantum modality.

Also the *exponential modality* traditionally postulated in linear logic is an emergent effect in `LHoTT`, as is the crucial *Quantum Modality*, not considered before:

linear randomization

aka: stabilization/motivization

quantum modality

$$\begin{tikzcd}
\mathrm{Type} \arrow[loop left, "Q"] \arrow[r, shift left=2, "\Sigma^\infty_+ \,:\, B \,\mapsto\, \star_B \mathbb{1}"] \arrow[r, phantom, "\bot"] & \mathrm{LType} \arrow[l, shift left=2, "\Omega^\infty"] \arrow[loop right, "!"]
\end{tikzcd}$$

exponential modality

classical types

linear types

The Quantum modality.

Also the *exponential modality* traditionally postulated in linear logic is an emergent effect in `LHoTT`, as is the crucial *Quantum Modality*, not considered before:

linear randomization

aka: stabilization/motivization

quantum modality

$$\begin{tikzcd} \mathrm{Type} \arrow[loop left, "Q"] \arrow[r, shift left=2, "\Sigma^{\infty}_{+} \,:\, B \;\mapsto\; \star_{B} \mathbb{1}"] \arrow[r, phantom, "\perp"] & \mathrm{LType} \arrow[loop right, "!"] \arrow[l, shift left=2, "\Omega^{\infty}"] \end{tikzcd}$$

exponential modality

classical types

linear types

The Q-monad plays a crucial role in the full formulation of the `QS`-language.

It is the secret actor behind QBit $=$ Q(Bit)...

The Quantum modality.

Also the *exponential modality* traditionally postulated in linear logic
is an emergent effect in `LHoTT`,
as is the crucial *Quantum Modality*, not considered before:

linear randomization

aka: stabilization/motivization

quantum modality

$$\begin{tikzcd}
\mathrm{Type} \arrow[loop left, "Q"] \arrow[r, shift left=2, "\Sigma^{\infty}_{+} \,:\, B \,\mapsto\, \star_{B} \mathbb{1}"] \arrow[r, phantom, "\perp"] & \mathrm{LType} \arrow[l, shift left=2, "\Omega^{\infty}"] \arrow[loop right, "!"]
\end{tikzcd}$$

exponential modality

classical types

linear types

The Q-monad plays a crucial role in the full formulation of the `QS`-language.
It is the secret actor behind QBit $=$ Q(Bit)...

Quantum Circuits

Quantum effects are compatible with tensor product.

Linear Randomness and Indefiniteness are “very strong” effects, in that:

$$\bigcirc_B\big(D \otimes D'\big) \;\simeq\; \big(\bigcirc_B D\big) \otimes D'\,, \qquad \star_B\big(D \otimes D'\big) \;\simeq\; \big(\star_B D\big) \otimes D'$$

There is a whole system of them:

$$\bigcirc_B \bigcirc_{B'} \;\simeq\; \bigcirc_{B \times B'}\,, \qquad \text{NB: } \bigcirc_B \bigcirc'_B \;\simeq\; \bigcirc_B \mathbb{1} \otimes \bigcirc'_B$$

which under dynamic lifting (monadicity comparison functor)
gives the external tensor product of dependent linear types:

free $\bigcirc_B$-effect handlers in linear data types

comparison functor

B-dependent linear data types

$$\begin{tikzcd} & \bigcirc_{B\times B'} \arrow[r, bend left, "\bigcirc_{B\times B'}"] & \bigcirc_{B\times B'}\mathcal{H} \arrow[d, mapsto] \\ \mathrm{LType}_{\bigcirc_B} \arrow[d, "K_{\oplus_B \mathbb{1}_B \otimes}"] & & \\ \mathrm{LType}_B & \big(\Box_B\, 1_B\big) \boxtimes \big(\Box_{B'}\, 1_{B'}\big) \otimes \mathcal{H} & \end{tikzcd}$$

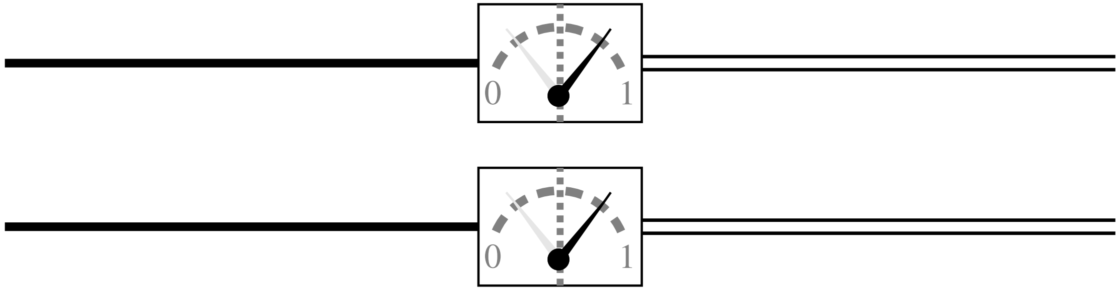
external tensor product

Quantum circuits with classical control & effects

are the *effectful* string diagrams in the linear type system

E.g.

The dependent linear type of a measurement on a pair of qbits:



type of a pair of coherent qbits

pair of measurements

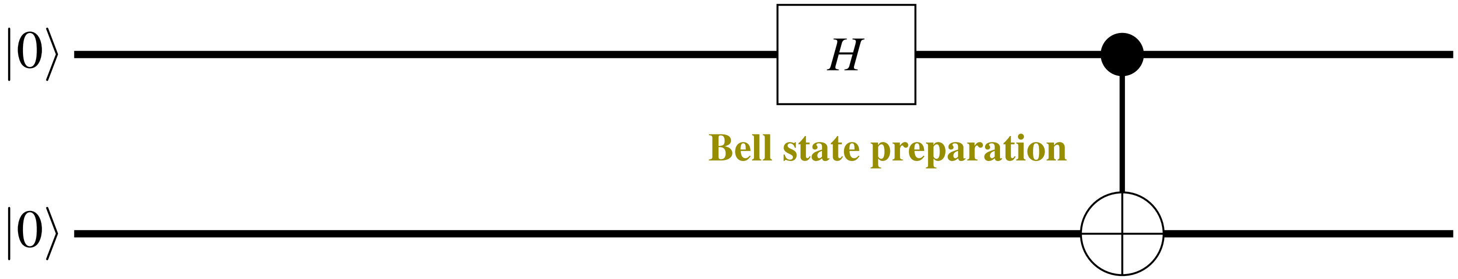
type of collapsed qbits dependent on measured bits b, b'

$$\Box_{\mathrm{Bit}^2}\big(\mathrm{QBit}_\bullet \boxtimes \mathrm{QBit}_\bullet\big) \xrightarrow{\;\varepsilon_{\mathrm{Bit}^2}(\mathrm{QBit}_\bullet \boxtimes \mathrm{QBit}_\bullet)\;} \mathrm{QBit}_\bullet \boxtimes \mathrm{QBit}_\bullet$$

measured bits

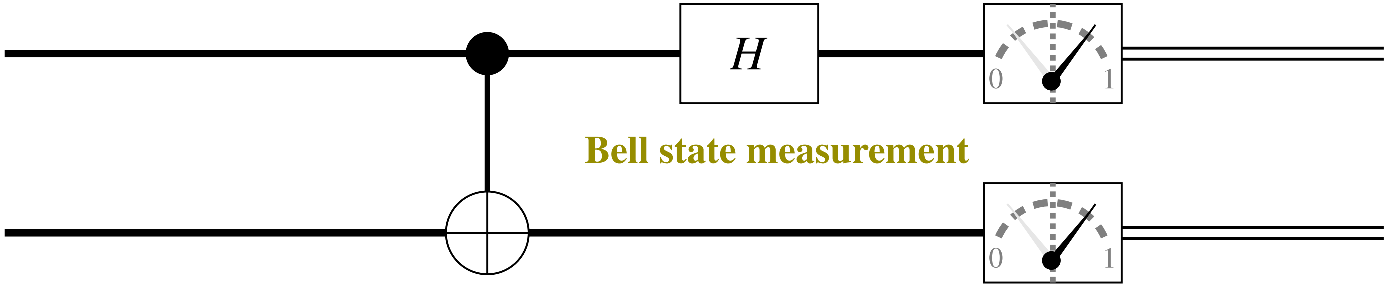
$$(b, b') : \mathrm{Bit}^2 \;\;\vdash\;\; \Box_{\mathrm{Bit}^2}\big(\mathrm{QBit}_\bullet \boxtimes \mathrm{QBit}_\bullet\big)_{(b,b')} \simeq \mathbb{C}^2 \otimes \mathbb{C}^2 \xrightarrow[\text{collapse of the quantum state}]{\;\Sigma_{d,d'}\, q_{dd'}\; |d\rangle \otimes |d'\rangle \;\mapsto\; q_{bb'}\; |b\rangle \otimes |b'\rangle\;} \mathbb{C}\,.$$

Example: Bell states of q-bits are typed as follows (regarded in $\mathrm{LType}_{\mathrm{Bit}\times\mathrm{Bit}}$):



$$\mathrm{QBit}_\bullet \boxtimes \mathrm{QBit}_\bullet \longrightarrow \big(\lozenge_{\mathrm{Bit}}\mathrm{QBit}_\bullet\big) \boxtimes \big(\lozenge_{\mathrm{Bit}}\mathrm{QBit}_\bullet\big) \simeq \square_{\mathrm{Bit}^2}\big(\mathrm{QBit}_\bullet \boxtimes \mathrm{QBit}_\bullet\big) \longrightarrow \square_{\mathrm{Bit}^2}\big(\mathrm{QBit}_\bullet \boxtimes \mathrm{QBit}_\bullet\big)$$

$$b, b' : \mathrm{Bit} \vdash \mathbb{C} \xrightarrow{\;1 \;\mapsto\; |0\rangle\otimes|0\rangle \;\mapsto\; \frac{1}{\sqrt{2}}\big(|0\rangle+|1\rangle\big)\otimes|0\rangle \;\mapsto\; \frac{1}{\sqrt{2}}\big(|0\rangle\otimes|0\rangle+|1\rangle\otimes|1\rangle\big)\;} \mathbb{C}^2 \otimes \mathbb{C}^2$$

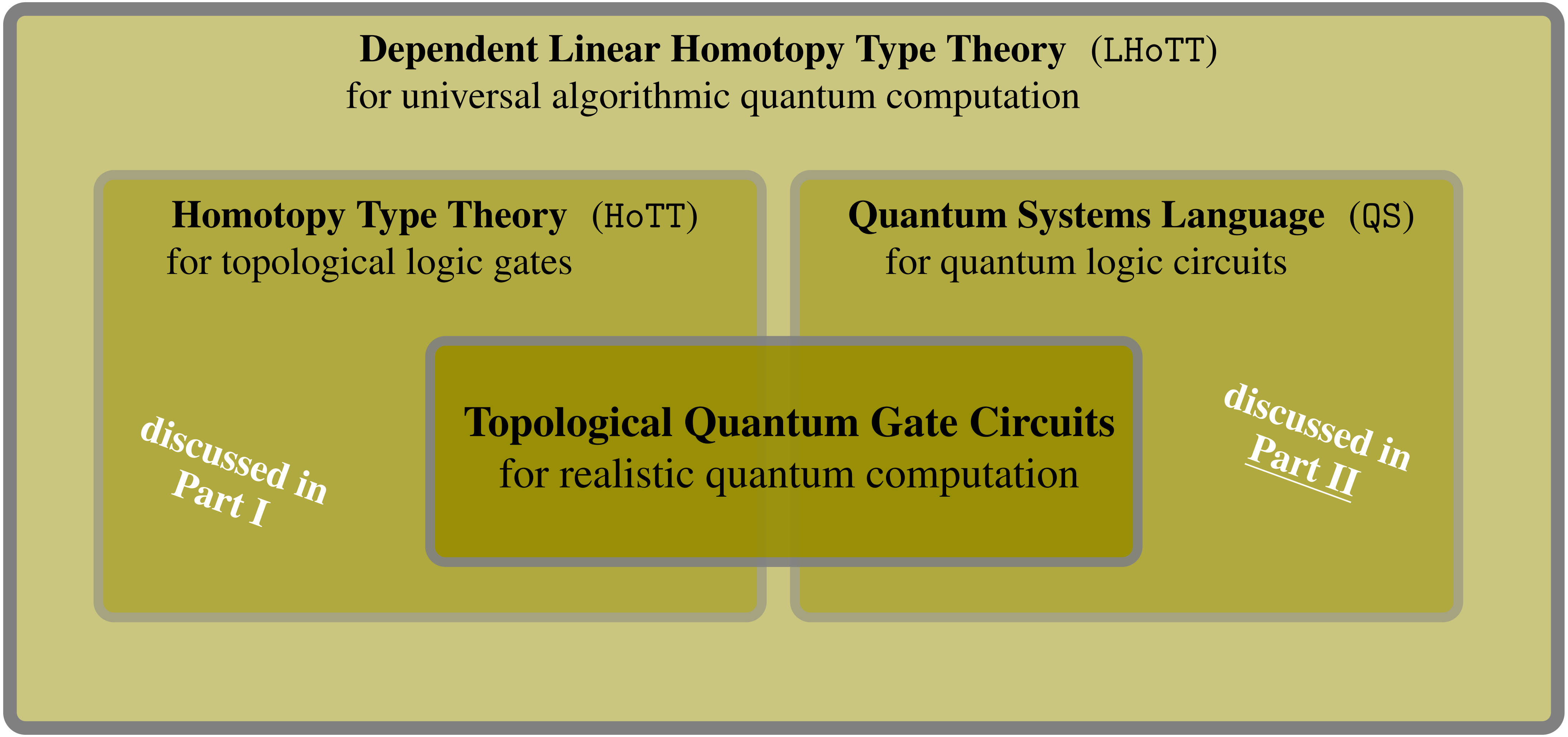


$$\square_{\mathrm{Bit}^2}\big(\mathrm{QBit}_\bullet \boxtimes \mathrm{QBit}_\bullet\big) \longrightarrow \mathrm{QBit}_\bullet \boxtimes \mathrm{QBit}_\bullet$$

$$b_1, b_2 : \mathrm{Bit} \vdash \mathbb{C}^2 \otimes \mathbb{C}^2 \xrightarrow{\;\Sigma_{b'_1 b'_2}\, q_{b'_1 b'_2} \cdot |b'_1\rangle\otimes|b'_2\rangle \;\mapsto\; \big(q_{0,b_2} + (-1)^{b_1} \cdot q_{1,(1-b_2)}\big)\cdot|b_1\rangle\otimes|b_2\rangle\;} \mathbb{C}$$

`QS` – Quantum Systems language @ CQTS

⇝ full-blown Quantum Systems language emerges embedded in `LHoTT`



Dependent Linear Homotopy Type Theory (`LHoTT`)
for universal algorithmic quantum computation

Homotopy Type Theory (`HoTT`)
for topological logic gates

Quantum Systems Language (`QS`)
for quantum logic circuits

Topological Quantum Gate Circuits
for realistic quantum computation