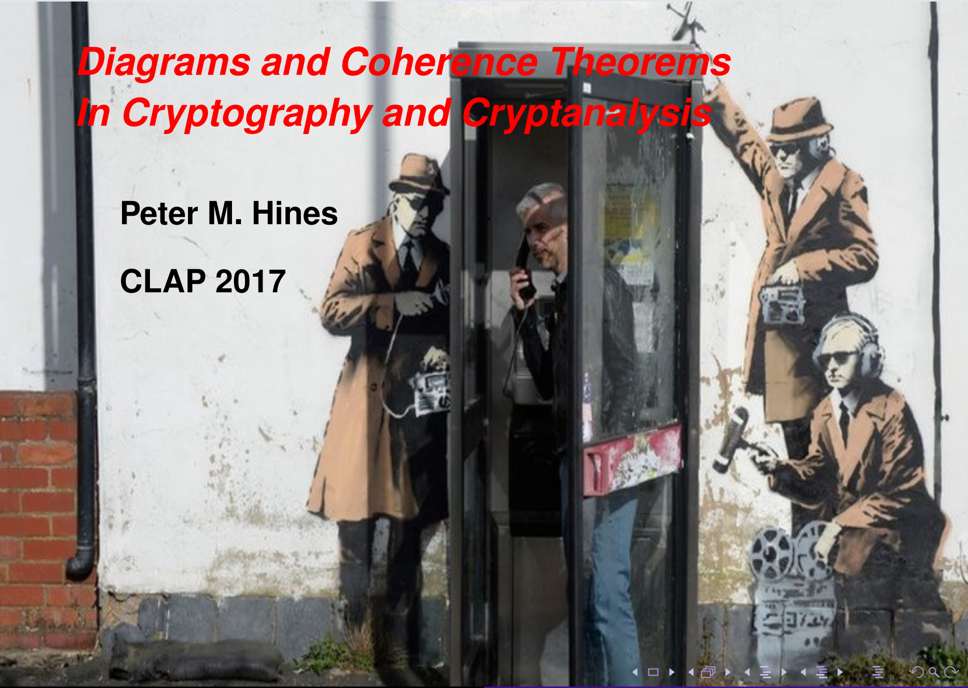
Diagrams and Coherence Theorems In Cryptography and Cryptanalysis



Peter M. Hines

CLAP 2017

The very basics

Cryptography

The art and science of ensuring information can only be understood by *certain people*.

Cryptanalysis

The art and science of ensuring you are *one of those people*.

> *"It is clear that the cryptographers are winning the information war . . .*
>
> *. . . experience tells us that every unbreakable cipher eventually succumbs to cryptanalysis."*

– The Code Book, Simon Singh

An overview ...

This talk is about:

1. Reasoning about cryptographic protocols using categorical diagrams.
2. Some unexpected connections with the foundations of category theory.

'Cryptography for category theorists', not vice versa!

Completely unbreakable encryption(!)

Alice and **Bob** wish to communication privately.

- They meet up to generate a large string of **random** binary digits: $(s_1, s_2, \ldots s_n) \in \mathbb{Z}_2^n$.

  This is their **shared secret** (the **one-time pad**).

- Alice later wishes to send to Bob a message

  $$(m_1, m_2, \ldots, m_k) \in \mathbb{Z}_2^k$$

  She (insecurely) transmits $(m_1 + s_1, m_2 + s_2, \ldots, m_k + s_k)$.

- Everyone can see this message, but a copy of the *shared secret* is needed to decode it:

$$(m_1+s_1+s_1, m_2+s_2+s_2, \ldots, m_k+s_k+s_k) = (m_1, m_2, \ldots, m_k)$$

Completely unbreakable encryption(!)

Alice and **Bob** wish to communication privately.

- They meet up to generate a large string of **random** binary digits: $(s_1, s_2, \ldots s_n) \in \mathbb{Z}_2^n$.

  This is their **shared secret** (the **one-time pad**).

- Alice later wishes to send to Bob a message

  $$(m_1, m_2, \ldots, m_k) \in \mathbb{Z}_2^k$$

  She (insecurely) transmits $(m_1 + s_1, m_2 + s_2, \ldots, m_k + s_k)$.

- Everyone can see this message, but a copy of the *shared secret* is needed to decode it:

  $$(m_1+s_1+s_1, m_2+s_2+s_2, \ldots, m_k+s_k+s_k) = (m_1, m_2, \ldots, m_k)$$

Completely unbreakable encryption(!)

Alice and **Bob** wish to communication privately.

- They meet up to generate a large string of **random** binary digits: $(s_1, s_2, \ldots s_n) \in \mathbb{Z}_2^n$.

  This is their **shared secret** (the **one-time pad**).

- Alice later wishes to send to Bob a message

  $$(m_1, m_2, \ldots, m_k) \in \mathbb{Z}_2^k$$

  She (insecurely) transmits $(m_1 + s_1, m_2 + s_2, \ldots, m_k + s_k)$.

- Everyone can see this message, but a copy of the *shared secret* is needed to decode it:

$$(m_1+s_1+s_1, m_2+s_2+s_2, \ldots, m_k+s_k+s_k) = (m_1, m_2, \ldots, m_k)$$

Completely unbreakable encryption(!)

Alice and **Bob** wish to communication privately.

- They meet up to generate a large string of **random** binary digits: $(s_1, s_2, \ldots s_n) \in \mathbb{Z}_2^n$.

  This is their **shared secret** (the **one-time pad**).

- Alice later wishes to send to Bob a message

$$(m_1, m_2, \ldots, m_k) \in \mathbb{Z}_2^k$$

  She (insecurely) transmits $(m_1 + s_1, m_2 + s_2, \ldots, m_k + s_k)$.

- Everyone can see this message, but a copy of the *shared secret* is needed to decode it:

$$(m_1 + s_1 + s_1, m_2 + s_2 + s_2, \ldots, m_k + s_k + s_k) = (m_1, m_2, \ldots, m_k)$$

Perfect, but impractical (I)

"Anyone who considers algorithmic methods of producing random digits is, of course, living in a state of sin." – **J. von Neumann**

"One time pads are an absolutely *ancient* idea that is easy to implement by means of an ebook that both parties have to independently download" – register.co.uk comments 01/04/2017.

It is important that the shared secret is not reused!

The Venona project (1943-80)

An attack on Soviet encryption by the U.S.

Signals Intelligence Unit or *National Security Agency*

Spectacular sucesses due to *re-use of one-time pads*.

‘We’ll meet again ...’

When their one-time pad has been used up, Alice and Bob have two options:

1. meet up again, to generate more random sequences.
2. rely on a trusted network of couriers.

Both of these options are *inconvenient & insecure*.

Is it possible for Alice and Bob to share a secret without ever having to meet?

Public Key Distribution

Alice and Bob can come to share a secret, even when all their communications are being monitored.

Diffie – Hellman key exchange (1976)

- Relies on the *difficulty* of computing discrete logarithms.
- Very heavily used online.
- Highly vunerable to *quantum computers*.

Security through obscurity?

Previously discovered by Ellis, Cocks, Williamson of GCHQ.

A motivating thought-experiment

Prior to D.-H (or E-C-W), it was believed that such secret-sharing should be possible.

The 'untrusted courier' scenario

Alice wishes to send Bob some physical object.

- Alice padlocks it into a box & sends the locked box to Bob.
- Bob is unable to open it; he secures the box with his own padlock & returns it to Alice.
- Alice is unable to open it; she removes her padlock & sends it back to Bob.
- Bob receives a box that is secured with his padlock only.

Commutativity & the untrusted courier

Algebraic requirements ...

- Locking operations have left inverses.

$$\begin{tikzcd}
 & \square \arrow[dl, "Alice_Locks"'] \arrow[dr, "Bob_Locks"] \arrow[dd, "Id"] & \\
\square \arrow[dr, "Alice_Unlocks"'] & & \square \arrow[dl, "Bob_Unlocks"] \\
 & \square &
\end{tikzcd}$$

- Locking operations commute with each other.

$$\begin{tikzcd}
\square \arrow[r, "Alice_Locks"] \arrow[d, "Bob_Locks"'] & \square \arrow[d, "Bob_Locks"] \\
\square \arrow[r, "Alice_Locks"'] & \square
\end{tikzcd}$$

Epistemic requirements ...

- Only **Alice** can perform:
  - $Alice\_locks : \square \to \square$
  - $Alice\_Unlocks : \square \to \square$
- Only **Bob** can perform:
  - $Bob\_Locks : \square \to \square$
  - $Bob\_Unlocks : \square \to \square$

Aims and Objectives:

1. Express entire protocols as commuting diagrams.
2. Use a single diagram to model
   - Algebra

     Commuting (canonical?) diagrams
   - Knowledge

     Partial order enrichment
   - Information flow

     2-categorical structure
3. Use these to ~~attack~~ study protocols.

Protocols as diagrams

Aims and Objectives:

1. Express entire protocols as commuting diagrams.
2. Use a single diagram to model
   - Algebra
     Commuting (canonical?) diagrams
   - Knowledge
     Partial order enrichment
   - Information flow
     2-categorical structure
3. Use these to ~~attack~~ study protocols.

A family of key exchange protocols

For obvious (quantum) reasons, we seek secret-sharing protocols that are not based on prime fields / factorization / discrete logarithms / etc.

Recent work (January 2017) suggests that *graph isomorphism* is also not a good place to start:

> *"Graph isomorphism in quasi-polynomial time"* – *Lásló Babai, Univ. Chicago*

We will look at some proposed *algebraic* protocols instead.

An algebraic approach to secret sharing

Commuting Action Key Exchange (CAKE)

- A general family of key exchange (secret sharing) protocols.
- Introduced in 2004 by V. Shpilrain & G. Zapata
- Includes many interesting protocols as special cases

  (Ko-Lee key exchange, Braid group protocols, Shpilrain – Ushakov protocol, &c..).

We will look at the semigroup (monoid) version:

Example 3, Section 3 of *Combinatorial Group Theory and Public Key Cryptography* S.-Z. (2004).

CAKE – sharing protocol

Alice and Bob will come to share a secret element of a semigroup $\mathcal{M}$.

1. Alice and Bob both have large **key pools** $A, B \subseteq \mathcal{M}$ that satisfy

$$ab = ba \;\; \forall\, a \in A, \;\; b \in B.$$

2. A fixed public **root element** $\gamma \in \mathcal{M}$ is chosen.
3. Alice chooses her **private key**, $(\alpha_1, \alpha_2) \in A \times A$, and publicly broadcasts $\alpha_1 \gamma \alpha_2 \in \mathcal{M}$
4. Bob chooses his **private key**,$(\beta_1, \beta_2) \in B \times B$, and publicly broadcasts $\beta_1 \gamma \beta_2 \in \mathcal{M}$.
5. Alice computes $\alpha_1 \beta_1 \gamma \beta_2 \alpha_2$ and Bob computes $\beta_1 \alpha_1 \gamma \alpha_2 \beta_2$.

By the point-wise commutativity of $A, B \subseteq \mathcal{M}$, these are equal, giving Alice and Bob's **shared secret** σ as

$$\sigma \;=\; \alpha_1 \beta_1 \gamma \beta_2 \alpha_2 \;=\; \beta_1 \alpha_1 \gamma \alpha_2 \beta_2$$

In a clearer form!

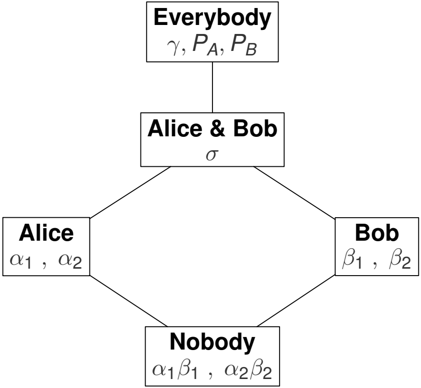
The algebraic data:

Alice	**Public**	**Bob**
	Public root γ	
Selects private $\alpha_1, \alpha_2 \in A$		Selects private $\beta_1, \beta_2 \in B$
Sends $\alpha_1 \gamma \alpha_2$	$\xrightarrow{P_A}$	
	$\xleftarrow{P_B}$	Sends $\beta_1 \gamma \beta_2$
Computes: $\alpha_1 P_B \alpha_2$	*By commutativity, these are equal.*	Computes: $\beta_1 P_A \beta_2$

Knowns and unknowns in semigroup CAKE

The participants: $\{$ Alice, Bob, Eve $\}$.

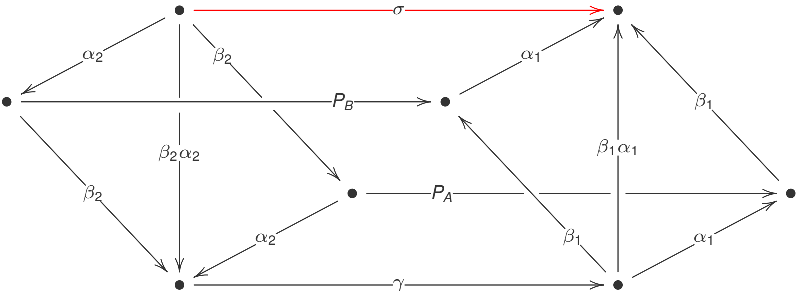
The epistemic data:



CAKE as a commuting diagram over a monoid

The required arrows are:

1. The root γ
2. Alice & Bob's private keys, (α_1, α_2) and (β_1, β_2)
3. Alice & Bob's public announcements, P_A and P_B
4. Their shared secret σ



Combining algebraic & epistemic data

Introducing epistemic data to diagrams

- Form the powerset-lattice of participants.
- Label each edge in the diagram by an element of this lattice:

$$\bullet \xrightarrow{f,X} \bullet$$

$X \in 2^{\{Alice,Bob,Eve\}}$ consists of participants who

  - know the value of f, or (more accurately)
  - are able to perform the operation f.

The **Algebraic-Epistemic diagram** for semigroup-CAKE:

$$
\begin{tikzcd}
 & \bullet \arrow[rrr, red, "{\sigma,\{A,B\}}"] \arrow[dl, "{\alpha_2,\{A\}}"'] \arrow[drr, "{\beta_2,\{B\}}"] \arrow[ddd, "{\beta_2\alpha_2,\perp}"] & & & \bullet & \\
\bullet \arrow[rrr, "{P_B,\top}"] \arrow[ddr, "{\beta_2,\{B\}}"'] & & & \bullet \arrow[ur, "{\alpha_1,\{A\}}"] & & \\
 & & \bullet \arrow[dl, "{\alpha_2,\{A\}}"] \arrow[rrr, "{P_A,\top}"] & & & \bullet \arrow[ul, "{\beta_1,\{B\}}"'] \\
 & \bullet \arrow[rrr, "{\gamma,\top}"] & & & \bullet \arrow[uuu, "{\beta_1\alpha_1,\perp}"] \arrow[uul, "{\beta_1,\{B\}}"] \arrow[ur, "{\alpha_1,\{A\}}"'] &
\end{tikzcd}
$$

Commuting diagrams??

Treating $2^{\{A,B,E\}}$ as a $\wedge$-monoid:

Question: Is this diagram for CAKE a commuting diagram over the product category $\mathcal{M} \times 2^{\{A,B,E\}}$?

Answer: No!

Turning a bug into a feature: *The reasons why / points at which it fails to commute are highly significant.*

1. Information sharing by participants.
2. Different routes to calculating the same value.

Failure of commutativity & public announcements

Diagram 1 commutes, Diagram 2 is a slice of CAKE.

$$\begin{tikzcd} \bullet \arrow[r, "{\beta_1\gamma\beta_2,\{B\}}"] \arrow[d, "{\beta_2,\{B\}}"'] \arrow[dr, phantom, "\text{Diagram } \mathfrak{D}_1"] & \bullet \\ \bullet \arrow[r, "{\gamma,\{A,B,E\}}"'] & \bullet \arrow[u, "{\beta_1,\{B\}}"'] \end{tikzcd} \qquad \begin{tikzcd} \bullet \arrow[r, "{\beta_1\gamma\beta_2,\top}"] \arrow[d, "{\beta_2,\{B\}}"'] \arrow[dr, phantom, "\text{Diagram } \mathfrak{D}_2"] & \bullet \\ \bullet \arrow[r, "{\gamma,\{A,B,E\}}"'] & \bullet \arrow[u, "{\beta_1,\{B\}}"'] \end{tikzcd}$$

1. In **diagram 1**, Bob computes $\beta_2\gamma\beta_1$, and *keeps quiet*.
2. In **diagram 2**, Bob computes $\beta_2\gamma\beta_1$, and *tells the whole world the result*.

Public announcements as 2-categorical data

Announcements are 2-cells:

$$\begin{tikzcd}
\bullet \arrow[rr, "{\beta_1\gamma\beta_2,\top}"] \arrow[d, "{\beta_2,\{B\}}"'] & \arrow[d, phantom, "\Uparrow (=,\leqslant)"] & \bullet \\
\bullet \arrow[rr, "{\gamma,\{A,B,E\}}"'] & & \bullet \arrow[u, "{\beta_1,\{B\}}"']
\end{tikzcd}$$

but not all such 2-cells are announcements!

In a well-designed protocol ...

we have a single simple property they satisfy.

A simple definition ...

A diagram $\mathfrak{D}$ over a **Poset** enriched category satisfies the **edge-path condition (EPC)** when:

- Given **an edge and a path** between the nodes X and Y, we have the following 2-cell:

$$\begin{tikzcd} & \cdots \arrow[dr, "g_n"] \arrow[d, phantom, "\Downarrow \leqslant_{X,Y}"] & \\ X \arrow[ur, "g_1"] \arrow[rr, "f"'] & & Y \end{tikzcd}$$

- Given nodes X, Y with **paths but no edges** between them, we have the following 2-cell:

$$\begin{tikzcd} & & \cdots \arrow[drr, "g_n"] \arrow[d, phantom, "\Downarrow =_{X,Y}"] & & \\ X \arrow[urr, "g_1"] \arrow[r] & h_1 & \cdots & h_k \arrow[r] & Y \end{tikzcd}$$

The Edge-Path condition & protocols

Model protocols using EPC diagrams over a product category $\mathcal{C} \times L$.

- $\mathcal{C}$ models the algebraic structure, and is enriched over the discrete partial order.
- $\mathcal{L}$ models the participants / epistemic data, and has more interesting poset-enrichment.

Consider left- and right- projections

For such a diagram $\mathfrak{D}$,

- The projection $\pi_1(\mathfrak{D})$ is a commuting diagram over $\mathcal{C}$
- The projection $\pi_2(\mathfrak{D})$ simply satisfies the E-P condition.

We can define a C-EPO diagrams over any product category $\mathcal{C} \times \mathcal{L}$, where $\mathcal{L}$ is enriched over **Poset**.

For this talk, we simply need $\mathcal{L}$ to be a lattice
(usually the powerset-lattice of participants).

Even for current protocols, we need $\mathcal{C}$ to be a category, not just a monoid.

Interpreting the edge-path condition

Motivation: Why such conditions on diagrams??

Experimentally – we always find this to be the case.

Conceptually – we will justify this by considering powerset-lattices of participants.

Practically – if this fails, we are missing something!

The edge-path condition: who knows what?

Consider a fragment of the A-E diagram for some protocol:

$$\begin{tikzcd} \bullet \arrow[r, "a_2, R_2"] & \cdots \arrow[r, "a_{n-1}, R_{n-1}"] & \bullet \arrow[d, "a_n, T_n"] \\ H \arrow[u, "a_1, R_1"] \arrow[rr, "b, Q"'] & & K \end{tikzcd}$$

The edge-path condition states that

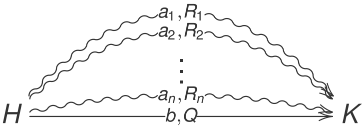
$$b = a_n \ldots a_1 \text{ and } \bigwedge_{j=1}^{n} R_j \leqslant Q$$

In terms of powerset-lattices

Any participant $x \in \bigwedge_{j=1}^{n} R_j$ who knows (is able to perform) each operation $\{a_j\}_{j=1..n}$ certainly knows (is able to perform) the composite $r_n \ldots r_1$.

No participant left behind

Consider a fragment of an A-E diagram for some protocol with a **single edge** and **multiple paths** from node H to node K.



The edge-path condition states that

$$b = a_1 = \ldots = a_n \text{ and } R_j \leqslant Q \; \forall \, j = 1..n$$

In terms of powerset-lattices

The members of $R_1, R_2, \ldots, R_n$ are all able to calculate (perform) b, albeit in different ways. Therefore, the subset of participants who can perform b contains each R_j.

A worked example

Tripartite Diffie-Hellman key exchange

The usual story ...

Three participants $\{Alice, Bob, Carol\}$ will come to share a secret.

Start with a (public) prime p and **root** $g \in \mathbb{Z}_p$.

- *Alice*, *Bob*, and *Carol* have private keys $a, b, c \in \mathbb{Z}_p$.
- They will construct the shared secret $g^{abc} = g^{bca} = g^{cab}$.
- All three of them are required, to construct this.
- The usual evesdropper *Eve* can see all communication.

Tripartite Diffie-Hellman, Round I

Based on the **public root** g, and their **private keys** a, b, c,

1. Alice computes g^a and announces the result to Bob.
2. Bob computes g^b and announces the result to Carol.
3. Carol computes g^c and announces the result to Alice.

Tripartite Diffie-Hellman, Round II

Based on the messages they receive,

1. Alice computes $(g^c)^a = g^{ca}$ and announces the result to Bob.
2. Bob computes $(g^a)^b = g^{ab}$ and announces the result to Carol.
3. Carol computes $\left(g^b\right)^c = g^{bc}$ and announces the result to Alice.

Tripartite Diffie-Hellman, Round III

They are now able to compute the shared secret.

1. Alice computes $\left(g^{bc}\right)^a = g^{abc}$.
2. Bob computes $\left(g^{ca}\right)^b = g^{abc}$
3. Carol computes $\left(g^{ab}\right)^c = g^{abc}$.

The underlying category

The action takes place in a small subcategory of **Set**:

- **Objects:** $\mathbb{Z}_p$ and $\{*\}$
- **Arrows:**
  1. *modular exponentiation* $(\ )^x : \mathbb{Z}_p \to \mathbb{Z}_p$, for all $x = 0 \ldots p-1$
  2. *selecting an element* $[x] : \{\star\} \to \mathbb{Z}_p$, where $[x](\star) = x \in \mathbb{Z}_p$

The core identity

The basic identity is $(((\_)^a)^b)^c = (((\_)^b)^c)^a = (((\_)^c)^a)^b$

$$
\begin{tikzcd}
 & & \mathbb{Z}_p \arrow[dd, "(\_)^c"] & \\
\mathbb{Z}_p \arrow[dd, "(\_)^b"'] & \mathbb{Z}_p \arrow[l, "(\_)^a"'] \arrow[ur, "(\_)^b"] \arrow[drr, "(\_)^c"] & & \\
 & & \mathbb{Z}_p \arrow[dl, "(\_)^a"'] & \mathbb{Z}_p \arrow[dd, "(\_)^a"] \\
\mathbb{Z}_p \arrow[r, "(\_)^c"] & \mathbb{Z}_p & & \\
 & & & \mathbb{Z}_p \arrow[ull, "(\_)^b"]
\end{tikzcd}
$$

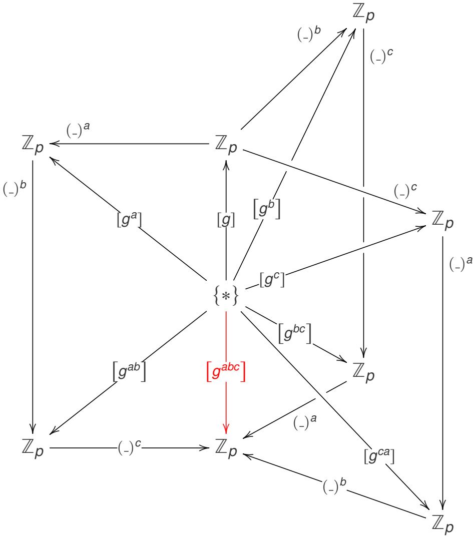
Adding in the root element

We require these equalities *applied to the root* $g \in \mathbb{Z}$.

$$
\begin{tikzcd}
 & & \mathbb{Z}_p \arrow[dddd, "(\_)^c"] & \\
\mathbb{Z}_p \arrow[dddd, "(\_)^b"'] & \mathbb{Z}_p \arrow[l, "(\_)^a"'] \arrow[ur, "(\_)^b"] \arrow[drr, "(\_)^c"] & & \\
 & & & \mathbb{Z}_p \arrow[dddd, "(\_)^a"] \\
 & \{*\} \arrow[uu, "{[g]}"] \arrow[dd, red, "{[g^{abc}]}"] & & \\
 & & \mathbb{Z}_p \arrow[dl, "(\_)^a"] & \\
\mathbb{Z}_p \arrow[r, "(\_)^c"] & \mathbb{Z}_p & & \\
 & & & \mathbb{Z}_p \arrow[uull, "(\_)^b"]
\end{tikzcd}
$$

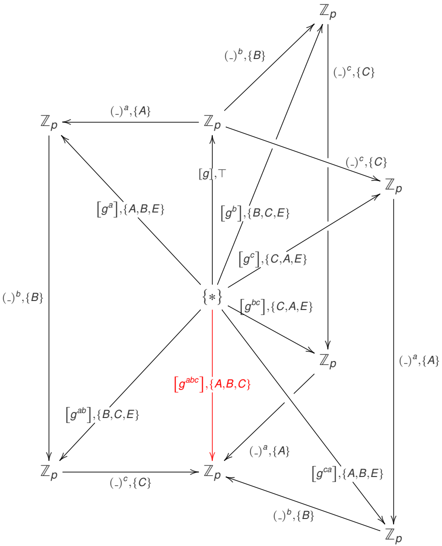
What announcements are made?

The elements $g^a, g^b, g^c, g^{ab}, g^{bc}, g^{ca}$ are all announced:



Who knows what?

Adding in the epistemic data:



Does this help??

Simple diagram-chasing makes it easy to answer some questions:

Question Can we vary the order of computations / announcements?

Answer Yes, quite a bit!

Question Does it matter if any of the participants (apart from Eve) are evesdropping?

Answer No, not at all!

Question What does Eve need to know, to find the shared secret?

Answer *Any of the private keys will do*!

We can also ***compare approaches*** to the same problem.

Another approach ...

How else may *Alice*, *Bob*, and *Carol* communicate privately?

As before, assume:

- Prime p,
- Public Root $g \in \mathbb{Z}_p$
- Private keys $a, b, c \in \mathbb{Z}_p$

Every pair will compute a *distinct* shared secret.

$Alice - -Bob \quad Bob - -Carol \quad Carol - -Alice$

Pairwise three-party Diffie-Hellman

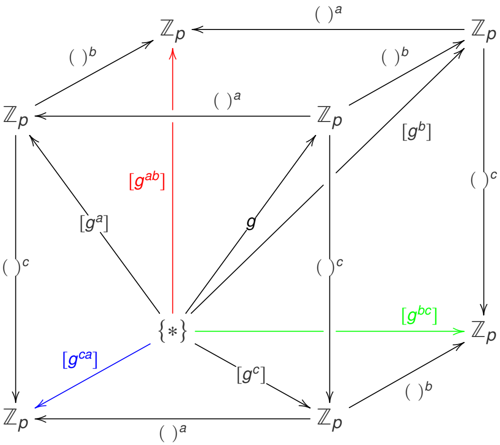
- *Alice*, *Bob*, and *Carol* compute

$$g^a \quad \text{and} \quad g^b \quad \text{and} \quad g^c$$

  respectively. They publicly announce their results.
- They each compute a *pair* of shared secrets:

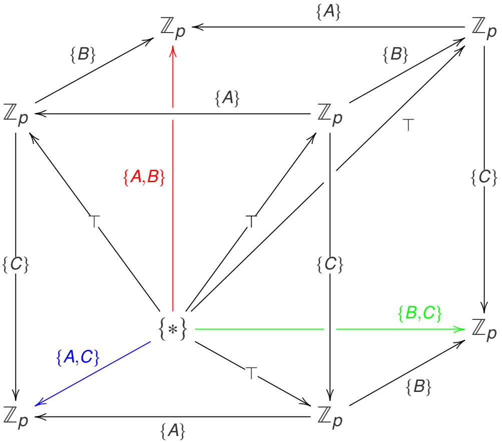
$$\begin{array}{rl} \text{Alice} & \text{computes } g^{ba} \text{ and } g^{ca} \\ \text{Bob} & \text{computes } g^{cb} \text{ and } g^{ab} \\ \text{Carol} & \text{computes } g^{ac} \text{ and } g^{bc} \end{array}$$

The (commuting) algebraic labelling:



A-E diagram for 3-way secret sharing

The (EPC satisfying) lattice labelling:



Again, by simple diagram-chasing:

Question Can any additional information be announced?

Answer No, not without compromising the protocol!

Question What happens if Eve discovers (say) Bob's secret key?

Answer She can discover two out of the three shared secrets.

Question Is this the same as tripartite Diffie-Hellman?

Answer *No, definitely not!*

Can we go further??

Drawing diagrams gives a *visual representation* of algebraic relationships, epistemic knowledge, and information flow.

The underlying algebra has been treated as a '**black box**'.

Is category theory relevant to the underlying algebra?

Back to CAKE

Recall the CAKE protocol

This is a *general recipe* for producing public key protocols.

The key ingredient is the choice of *semigroup*.

In fact, any structure with an associative composition will do.

We could even use canonical coherence isomorphisms!

An interesting first choice ...

CAKE was first proposed in:

Combinatorial group theory and public key cryptography (2004)

General proposals for cryptosystems based on algebraic structures.

A concrete protocol was given in:

Thompson's group and Public Key Cryptography (2004)

The underlying structure was Thompson's group $\mathcal{F}$.

Any particular reasons?

From TFA

- "This group has several properties that make it **particularly fit for cryptographic purposes**."
- "The difficulty of solving equations "resembles the **factorization problem** which is at the heart of the RSA cryptosystem."

Group-based cryptosystems are susceptible

to **length-based cryptanalysis** *(— pioneered by Shamir).*

This works best with groups that are

'close to being free' – Folklore

Thompson's group $\mathcal{F}$ is *'as far from free as possible'*.

Any quotient causes a collapse to an abelian monoid.

This Folklore is incorrect: "Length-based crytanalysis: the case of Thompson's group" – Ruinsky, Shamir, Tsaban (2007)

A practical reason ...

Group-based cryptosystems are susceptible

to **length-based cryptanalysis** *(— pioneered by Shamir).*

This works best with groups that are

'close to being free' – Folklore

Thompson's group $\mathcal{F}$ is *'as far from free as possible'*.

Any quotient causes a collapse to an abelian monoid.

This Folklore is incorrect: "Length-based crytanalysis: the case of Thompson's group" – Ruinsky, Shamir, Tsaban (2007)

This is an ex-protocol.

This protocol is not currently in use!

- **F. Matucci** (2006)

  The Shpilrain-Ushakov Protocol for Thompson's Group F is always breakable

- **Ruinskiy, Shamir, Tsaban** (2007)

  Length-Based Cryptanalysis: the case of Thompson's group

Conjecture: *" no practical public key cryptosystem based on the difficulty of solving an equation in this group can be secure."*

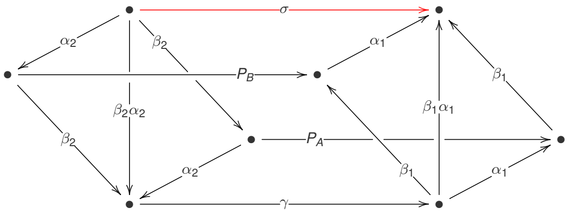
Thompson's group $\mathcal{F}$ and associativity

- **R. McKenzie, R. Thompson** (1971): Close connection between Thompson's group $\mathcal{F}$, and associativity laws
- **K. Brown** (2004) A group homomorphism $\_ \star \_ : \mathcal{F} \times \mathcal{F} \to \mathcal{F}$ that is *associative up to isomorphism*.
- **M. V. Lawson** (2004) The canonical associativity isomorphisms for a class of single-object tensors is precisely $\mathcal{F}$.
- **P. Dehornoy** (2005) 'The only [non-trivial] relations in this presentation of $\mathcal{F}$ correspond to the well-known MacLane-Stasheff pentagon.'
- **M. Brinn** (2005) 'the resemblance of the usual coherence theorems with Thompson's group $\mathcal{F}$'.
- **M. Fiore, T. Leinster** (2010) Thompson's group $\mathcal{F}$ is the symmetry group of an idempotent U in the free strict monoidal category generated by U.

Cryptographic protocols as canonical diagrams

Based on these: Thompson's group $\mathcal{F}$ is a group of associativity isomorphisms, in some setting.

Diagrams for the Shpilrain-Ushakov protocol are **commuting canonical diagrams** in the sense of MacLane's coherence theorem.



The precise setting needs some explanation ...

A bit of terminology ...

A **semi-monoidal category** $(\mathcal{C}, \otimes, \tau_{\_,\_,\_})$ is one that satisfies MacLane's axioms for a monoidal category,

- Functoriality
- Naturality
- Pentagon

except for those relating to the unit object.

The lack of a unit allows us to talk about **semi-monoidal monoids**, or **monoids with tensors**.

When we need a unit object

We rely on the theory of Saavedra units

- *Catégories Tannakiennes* A. Saavedra (1972)
- *Elementary Remarks on Units* J. Kock (2008)
- *Coherence for Weak Units* A. Joyal, J. Kock (2011)

Kock's simplification

A unit object U is a **cancellative pseudo-idempotent**

The functors $U \otimes \_$ and $\_ \otimes U$ are fully faithful, and $U \otimes U \cong U$.

When we need a unit object

We rely on the theory of Saavedra units

- *Catégories Tannakiennes* A. Saavedra (1972)
- *Elementary Remarks on Units* J. Kock (2008)
- *Coherence for Weak Units* A. Joyal, J. Kock (2011)

Kock's simplification

A unit object U is a **cancellative pseudo-idempotent**

The functors $U \otimes \_$ and $\_ \otimes U$ are fully faithful, and $U \otimes U \cong U$.

In the trivial case:

For a monoid $\mathcal{M}$ with a tensor $\_ \star \_$ (e.g. Thompson's group $\mathcal{F}$) the unique object is a unit object precisely when

$$(1 \star \_), (\_ \star 1) : \mathcal{M} \to \mathcal{M}$$

are isomorphisms.

The homology of Thompson's $\mathcal{F}$ – K. Brown (2004)

K. Brown emphasises that the tensor $(\_ \star \_)$ on $\mathcal{F}$ does **not** satisfy this condition.

In the trivial case:

For a monoid $\mathcal{M}$ with a tensor $\_ \star \_$ (e.g. Thompson's group $\mathcal{F}$) the unique object is a unit object precisely when

$$(1 \star \_), (\_ \star 1) : \mathcal{M} \to \mathcal{M}$$

are isomorphisms.

The homology of Thompson's $\mathcal{F}$ – K. Brown (2004)

K. Brown emphasises that the tensor $(\ \star \ )$ on $\mathcal{F}$ does **not** satisfy this condition.

A relevant cohence theorem:

Coherence and Strictification for Self-Similarity
Journal of Homotopy & related structures (PMH 2016)

A semi-monoidal equivalence of monogenic categories	
Self-similarity $S \cong S \otimes S$ up to isomorphism (a.k.a. idempotency)	Strict self-similarity $S = S \star S$ (a.k.a. being a monoid)

A relevant cohence theorem:

Coherence and Strictification for Self-Similarity
Journal of Homotopy & related structures (PMH 2016)

Dropping in the 'generic idempotent' of F.– L. (2010)

The group of associativity isomorphisms for a tensor on a monoid, in the 'free' setting is precisely Thompson's group $\mathcal{F}$.

As proved by M. V. Lawson (2004) in the case where the tensor has projections / injections.

Proposition (from PMH 2016):

A tensor $(\_ \star \_) : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ on a monoid
is strictly associative

$\Longleftrightarrow$

The unique object $\mathfrak{m}$ is the unit object.

Proof $(\Leftarrow)$ *(Standard Theory ...)* By the Eckmann-Hilton argument on the interchange law, the endomorphism monoid of a unit object is abelian, and the tensor coincides (up to isomorphism) with this abelian, associative, composition. $\square$

What does it mean to be ‘free’?

Proposition (from PMH 2016):

A tensor $(\_ \star \_) : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ on a monoid
is strictly associative

$$\Longleftrightarrow$$

The unique object $\mathfrak{m}$ is the unit object.

Proof $(\Leftarrow)$ *(Standard Theory ...)* By the Eckmann-Hilton argument on the interchange law, the endomorphism monoid of a unit object is abelian, and the tensor coincides (up to isomorphism) with this abelian, associative, composition. $\square$

Is it because I is strict?

Proof ($\Rightarrow$) The map

$$\eta = (1 \star \_ \star 1) : \mathcal{M} \hookrightarrow \mathcal{M}$$

is an injective monoid homomorphism, so $\mathcal{M} \cong \eta(\mathcal{M})$.

Define a semi-monoidal tensor on its image, by, for all $\eta(r), \eta(s) \in \eta(\mathcal{M})$

$$\eta(r) \odot \eta(s) = 1 \star (r \star s) \star 1$$

By construction, $(\mathcal{M}, \star) \cong (\eta(\mathcal{M}), \odot)$.

(Hence the unique object of $(\eta(\mathcal{M}), \odot)$ is idempotent).

By definition, for all $\eta(f) \in \eta(\mathcal{M})$,

$$\begin{aligned} 1 \odot \eta(f) &= 1 \star (1 \star f) \star 1 \\ &= (1 \star 1) \star f \star 1 \\ &= 1 \star f \star 1 \\ &= \eta(f) \end{aligned}$$

Thus $1 \odot \_ = Id_{\eta(\mathcal{M})} = \_ \odot 1$, so the unique object of $(\eta(\mathcal{M}), \odot)$ is a unit object!

However, $(\eta(\mathcal{M}), \odot) \cong (\mathcal{M}, \star)$. $\square$

Corollary Let $\mathcal{M}$ be a monoid with a tensor. Then either:

1. The group of associativity iso.s is isomorphic to $\mathcal{F}$
2. *M* is an abelian monoid, and tensor coincides with composition.

By definition, for all $\eta(f) \in \eta(\mathcal{M})$,

$$\begin{aligned} 1 \odot \eta(f) &= 1 \star (1 \star f) \star 1 \\ &= (1 \star 1) \star f \star 1 \\ &= 1 \star f \star 1 \\ &= \eta(f) \end{aligned}$$

Thus $1 \odot \_ \; = \; Id_{\eta(\mathcal{M})} \; = \; \_ \odot 1$, so the unique object of $(\eta(\mathcal{M}), \odot)$ is a unit object!

However, $(\eta(\mathcal{M}), \odot) \;\cong\; (\mathcal{M}, \star)$. □

Corollary Let $\mathcal{M}$ be a monoid with a tensor. Then either:

1. The group of associativity iso.s is isomorphic to $\mathcal{F}$
2. M is an abelian monoid, and tensor coincides with composition.

The key properties are *categorical*

One of the key properties required of $\mathcal{F}$ is highly categorical.

What about the others??

A particularly important one!

It "resembles the **factorization problem** which is at the heart of the RSA cryptosystem." – Shpilrain & Ushakov (2004)

Can there really be a connection between **coherence** *and* **modular arithmetic** *??*

The key properties are *categorical*

One of the key properties required of $\mathcal{F}$ is highly categorical.

What about the others??

A particularly important one!

It "resembles the **factorization problem** which is at the heart of the RSA cryptosystem." – Shpilrain & Ushakov (2004)

Can there really be a connection
between **coherence** *and* **modular arithmetic** *??*

Some relevant work:

Geometry of Interaction (I) — J.-Y. Girard (1988)

A representation of **Linear Logic** in terms of *partial isomorphisms*

(... after getting rid of some non-essential structure).

The representation of conjunction

$$(f \star g)(n) = \begin{cases} 2f\left(\frac{n}{2}\right) & n \ (mod\ 2) = 0 \\ \\ 2g\left(\frac{n-1}{2}\right) + 1 & n \ (mod\ 2) = 1 \end{cases}$$

Girard as a category theorist

This ‘conjunction’ was studied in category-theoretic & inverse semigroup theoretic terms by PMH, M. V. Lawson (1998,1999)

- It is a semi-monoidal tensor on a monoid.
- It is identical (up to scaling) to Brown’s tensor (2004) on a representation of $\mathcal{F}$
- It cannot be strictly associative!
- It is one of a large family of tensors

Associative up to isomorphism

The associativity isomorphism is:

$$\alpha(n) = \begin{cases} 2n & n \ (mod\ 2) = 0, \\ n+1 & n \ (mod\ 4) = 1, \\ \frac{n-1}{2} & n \ (mod\ 4) = 3. \end{cases}$$

In this concrete setting canonical isomorphisms are modular arithmetic functions.

Categorical coherence as modular arithmetic

The components of MacLane's pentagon

$$(id \star \tau)(n) = \begin{cases} n & n \ (mod\ 2) = 0 \\ 2n-1 & n \ (mod\ 4) = 1 \\ n+2 & n \ (mod\ 8) = 3 \\ \frac{n-1}{2} & n \ (mod\ 8) = 7 \end{cases} \qquad (\tau \star id)(n) = \begin{cases} 2n & n \ (mod\ 4) = 0 \\ n+2 & n \ (mod\ 8) = 2 \\ \frac{n+1}{2} & n \ (mod\ 8) = 6 \\ n & n \ (mod\ 2) = 1 \end{cases}$$

$$\tau.\tau(n) = \begin{cases} 4n & n \ (mod\ 2) = 0 \\ n+2 & n \ (mod\ 4) = 1 \\ \frac{n+1}{2} & n \ (mod\ 8) = 3 \\ \frac{n-3}{4} & n \ (mod\ 8) = 7 \end{cases} \qquad \tau^2(n) = (\tau \star id)\tau(id \star \tau)(n) \text{ for all } n \in \mathbb{N}$$

An arithmetic proof of the Pentagon condition seems quite tedious (!)

A general setting

(PMH, MVL 1998-99) Any dissection of $\mathbb{N}$ into two (infinite) disjoint subsets $\mathbb{N} = A \uplus B$ determines a distinct tensor on $End(\mathbb{N})$.

Of particular interest ...

In the case where we consider

$$\{n \ (mod \ p) = k\} \quad \text{and} \quad \{n \ (mod \ p) \neq k\}$$

our associativity isomorphisms are modular arithmetic functions.

Are these (as per Shpilrain - Ushakov) related to those used in RSA?

Another relevant reference:

Modular arithmetic identities from categorical coherence, *PMH (2013)*

Even when looking at the simplest case (Girard's conjunction):

The worst-case scenario – exponential / factorial growth

" categorical diagrams correspond to arithmetic identities over equivalence classes of the form $\{2^k.\mathbb{N} + x\}_{x=0\ldots 2^k-1}$. "

"there are $n!$ simple loops to consider."

"clearly this is unfeasible, even for moderately large diagrams".

A concrete example

Consider a canonical diagram over such functions:

$$
\begin{tikzcd}
\mathbb{N} \arrow[d, "\alpha"'] & & \mathbb{N} \arrow[ll, "\alpha"'] \arrow[rr, "\alpha^{-1}"] \arrow[d, "\alpha^{-1}\star\alpha"] & & \mathbb{N} \arrow[d, "\alpha^{-1}"] \\
\mathbb{N} & & \mathbb{N} \arrow[dl, "\alpha"'] \arrow[dr, "\alpha^{-1}"] & & \mathbb{N} \\
& \mathbb{N} \arrow[ul, "(\alpha(\alpha\star 1))\star 1"] \arrow[rr, "\alpha^{-2}"'] & & \mathbb{N} \arrow[ur, "1\star(\alpha^{-1}(1\star\alpha))"'] &
\end{tikzcd}
$$

How easy is it to decide whether this commutes?

A conjecture

"we suggest that this task is in fact *linear*, instead of *exponential*." – PMH 2013

Which canonical diagrams commute?

Recall the proof of MacLane's coherence theorem for associativity:

In a (non-abelian) monoid $\mathcal{M}$ with a tensor $\_ \star \_$,

The commuting canonical diagrams over $\mathcal{M}$

are precisely those that are the image

of some diagram over MacLane's $\mathcal{W}$,

under the usual substitution functor.

The great leap backwards ...

Let’s make this picture more complicated!

The naming of the variables

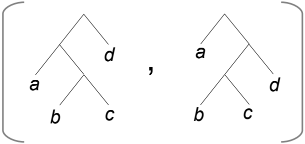
Start with: a countably infinite set Var of **variable symbols**.

We work with *binary trees, with each leaf labelled by a distinct variable symbol*.

Definition

A pair (S, T) of trees is a **linear pair** when *the leaf traversals of S and T are the same.*

A posetal groupoid of linear pairs



$$\textit{Leat Traversal} = (a\,,\; b\,,\; c\,,\; d)$$

Make a *posetal category* $\mathcal{LP}$ of linear pairs by:

$$(T,S)(Q,P) = \begin{cases} (T,P) & S = Q, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

This does not have a tensor

Bound variable names are unimportant

Define an equivalence $\sim_\alpha$ on linear pairs by

$$(Q, P) \ \sim_\alpha (T, S)$$

iff there exists an iso. $\phi : Var \to Var$ *such that*

$$(\phi(Q), \phi(P)) \ = \ (T, S)$$

Identifying equivalent pairs gives a functor:

$$\mathcal{LP} \xrightarrow{\ /\sim_\alpha\ } \mathcal{W}$$

(From linear pairs, to MacLane's category).

... to get something very familiar!

$$\begin{tikzcd}
\mathcal{LP} \arrow[r, "/\sim_\alpha"] \arrow[dr, dotted, "{[\quad]}"'] & \mathcal{W} \arrow[d, "Subst"] \\
& \mathcal{M}
\end{tikzcd}$$

Given a linear pair (T, S), we denote its image by $[T, S] \in \mathcal{M}$.

We will call these **clauses**.

Two crucial questions:

Given linear pairs (T,S) and (V,U) in $\mathcal{LP}$

1. How can we decide when $[T,S] \ = \ [V,U]$?
2. How can we find a linear pair (Q,P) such that

$$[Q,P] \ = \ [T,S]\,[V,U] \ ?$$

A very simple solution

All we need is that:

i/ $\mathcal{M}$ only has one object.

ii/ MacLane's functor $\mathcal{W} \rightarrow \mathcal{M}$ preserves tensors.

Simple consequences:

As $\mathcal{M}$ has a unique object,

$$[T, T] \; = \; 1_{\mathcal{M}} \quad \text{for all trees} \quad T$$

As a corollary:

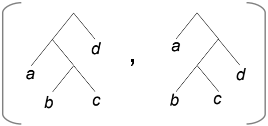
Given a linear pair (T, S), and a function

$$\theta : Var \to VarTree$$

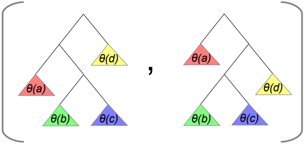
such that $(\theta(T), \theta(S))$ is also a linear pair, then

$$[T, S] \; = \; [\theta(T), \theta(S)]$$

Given a function $\theta : Var \to VarTree$, then the linear pairs:



and



are mapped to the same canonical iso. of $\mathcal{M}$.

Some complexity ...

A linear pair (Q, P) is in **simplest form** when,

for any substitution

$$(Q, P) \;=\; \eta(Q', P')$$

the pairs (Q, P) and (Q', P') have the same rank.

Reduction to simplest form accomplished by $O(n)$ algorithm:

R. Grossi (1992) "On finding common sub-trees".

Counting the linear pairs of rank n needs a surprisingly complex formula

Characterising composition

Given clauses $[V,U]$ and $[T,S]$, how can we find a linear pair (Q,P) satisfying:

$$[Q,P] = [V,U][T,S] \quad ?$$

Assume (w.l.o.g.) that U and T have no variables in common.

Can we find $\theta : \mathit{Var} \to \mathit{VarTree}$ such that $\theta(U) = \theta(T)$??

If so,

$$[V,U][T,S] \;=\; [\theta(V), \underbrace{\theta(U)][\theta(T)}, \theta(S)] \;=\; [\theta(V), \theta(S)]$$

Some (very standard!) theory:

Given binary trees T, U over distinct variable sets,

the set of '**unifiers**' of S, T,

$$\{\theta : Var \rightarrow VarTree \;\; s.t. \;\; \theta(T) = \theta(U)\}$$

is (up to variable renaming) a *poset*, with top element.

The top element is the **most general unifier**, written $mgu_{T,U}$.

Our composition becomes

$$[V, U][T, S] \;=\; [\theta(V), \theta(S)] \text{ where } \theta = mgu_{T,U}$$

This composition was introduced in the **clause algebras** of

Geometry of Interaction (III)

— J.-Y. Girard (1995)

It is seen in a large range of algebraic settings,

including *representations of Thompson's group*:

A correspondence between balanced varieties

— M. V. Lawson (2006)

Unification, generally

Let L be a **term language** freely built from:

- A set of n-ary **predicates** $\{P(\_,\_)\ ,\ Q(\_)\ ,\ R(\_,\_,\_)\ ,\ S()\ ,\ \ldots\}$
- A countably infinite set of **variable symbols** Var

A **substitution** $\sigma : Var \rightarrow L$ assigns terms to variable symbols in L.

A **unification** of a set of terms $\{T_j\}_{j=1}^N$ is a substitution $\mu : Var \rightarrow L$ where

$$\mu(T_i) = \mu(T_j) \ \ \forall i, j = 1 \ldots N$$

Robinson's Unification Algorithm either:

i/ Finds the (unique) most general unifier of $\{T_j\}$.

ii/ Reports that $\{T_j\}$ is not unifiable.

What is the complexity of unification?

- **Robinson (1965)**

  Exponentially complex $O(2^n)$ (in both time & space).

- **Martelli & Montanari (1976), Paterson & Wegman (1978)**

  A linear $O(n)$ algorithm for unification.

- **Ružička & Prívara (1982)**

  Robinson's original algorithm is made 'almost linear'

  i.e. $O(n^{1+\epsilon})$ complexity, where $\epsilon = \frac{1}{Ack(n,n)}$.

How complex is Robinson?

What is the complexity of unification?

- **Robinson (1965)**

  Exponentially complex $O(2^n)$ (in both time & space).

- **Martelli & Montanari (1976), Paterson & Wegman (1978)**

  A linear $O(n)$ algorithm for unification.

- **Ružička & Prívara (1982)**

  Robinson's original algorithm is made 'almost linear'

  i.e. $O(n^{1+\epsilon})$ complexity, where $\epsilon = \frac{1}{Ack(n,n)}$.

How complex is Robinson?

What is the complexity of unification?

- **Robinson (1965)**

  Exponentially complex $O(2^n)$ (in both time & space).

- **Martelli & Montanari (1976), Paterson & Wegman (1978)**

  A linear $O(n)$ algorithm for unification.

- **Ružička & Prívara (1982)**

  Robinson's original algorithm is made 'almost linear'

  i.e. $O(n^{1+\epsilon})$ complexity, where $\epsilon = \frac{1}{Ack(n,n)}$.

How complex is Robinson?

What is the complexity of unification?

- **Robinson (1965)**

  Exponentially complex $O(2^n)$ (in both time & space).

- **Martelli & Montanari (1976)**, **Paterson & Wegman (1978)**

  A linear $O(n)$ algorithm for unification.

- **Ružička & Prívara (1982)**

  Robinson's original algorithm is made 'almost linear'

  i.e. $O(n^{1+\epsilon})$ complexity, where $\epsilon = \frac{1}{Ack(n,n)}$.

Some consequences ...

When working with associativity isomorphisms,

The word problem is linear.

Deciding whether a diagram commutes is easy.

Key tools for solving equations involving unknowns:
Unification, *Resolution* and *Robinson's algorithm*.

Deciding whether a canonical diagram commutes

we do not need to consider $O(n!)$ simple loops.

This is a simple application of *logic programming*

The lazy approach ...

Let PROLOG sort it all out!

Interpret each canonical isomorphism in clause form as a logical proposition, & see whether they are all consistent.

More explicitly ...

Let $\mathfrak{D}$ be a canonical diagram, with nodes $\{n_0, \ldots n_k\}$.

For each edge labelled with canonical isomorphism c,

relabel with some linear pair (C_1, C_0) satisfying $[C_1, C_0] = c$.

$$
\begin{tikzcd}
 & \cdot \arrow[rr, "d"] \arrow[dr, "c"] & & \cdot \arrow[dl, "e"'] \arrow[dr, "f"] & \\
\cdot \arrow[ur, "a"] \arrow[rr, "b"] \arrow[drr, "h"'] & & \cdot \arrow[rr, "g"] & & \cdot \\
 & & \cdot \arrow[urr, "k"'] & &
\end{tikzcd}
$$

(Use distinct variable symbols for each edge!)

More explicitly ...

Let $\mathfrak{D}$ be a canonical diagram, with nodes $\{n_0, \ldots n_k\}$.

For each edge labelled with canonical isomorphism c,

relabel with some linear pair (C_1, C_0) satisfying $[C_1, C_0] = c$.

$$
\begin{tikzcd}
 & \cdot \arrow[rr, "{(D_0,D_1)}"] \arrow[dr, "{(C_0,C_1)}"] & & \cdot \arrow[dl, "{(E_0,E_1)}"'] \arrow[dr, "{(F_0,F_1)}"] & \\
\cdot \arrow[ur, "{(A_0,A_1)}"] \arrow[rr, "{(B_0,B_1)}"] \arrow[drr, "{(H_0,H_1)}"'] & & \cdot \arrow[rr, "{(G_0,G_1)}"] & & \cdot \\
 & & \cdot \arrow[urr, "{(K_0,K_1)}"'] & &
\end{tikzcd}
$$

(Use distinct variable symbols for each edge!)

Computing Unifiers

At the node n_0, we have the set of incident edges:

$$\text{incoming} \left\{ \begin{array}{ll} (T_1, T_1') & (S_1, S_1') \\ (T_2, T_2') & (S_2, S_2') \\ \ldots & \ldots \\ (T_x, T_x') & (S_y, S_y') \end{array} \right\} \text{outgoing}$$

Compute the most general unifier:

$$\theta_0 = mgu\{T_1, \ldots T_x, S_1', \ldots, S_y'\}$$

The iterative step:

Then apply this unifier θ_0 to <u>every</u> edge in the diagram.

We get a new diagram $\mathfrak{D}_1 = \theta_0(\mathfrak{D})$, with the same nodes.

Repeat this process for nodes $n_1, n_2, \ldots$

We get a series of re-labelled diagrams:

$$\mathfrak{D}_{n+1} = \theta_n(\mathfrak{D}_n)$$

If unification *ever* fails, the original diagram does not commute!

Assuming success ...

We have a diagram $\mathfrak{D}_n$ with edges labelled by linear pairs:

- Each linear pair has the same leaf traversal.
- Labelling is 'consistent' at every node.

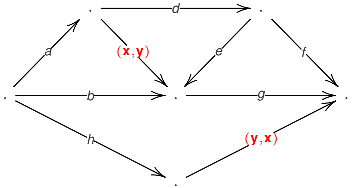
This is the simplest diagram over MacLane's $\mathcal{W}$ satisfying

$$Subst(\mathfrak{D}_n) = \mathfrak{D}$$

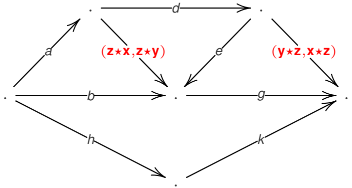
Not just a decision procedure – we get a witness.

Extending techniques ...

We can vary this algorithm, by re-using variable symbols:



“Red edges are mutually inverse”



“Red edges are of the form $1 \star \gamma$ and $\gamma^{-1} \star 1$”

Is this an isolated incident?

Stepping back a bit ...

At one point, cryptographers became fascinated with structures from the foundations of category theory ... was this a one-off?

Some other places to look ...

- Proposed use of Thompson's group $\mathcal{V}$

  – the coherence isomorphisms for a symmetric tensor on a monoid.
  M. Fiore, M. Campos (2013)

- Proposed use of poylcyclic monoids / groups.

  – related to coherence isomorphisms for tensors on monoids with projections / injections.
  PMH MVL (1998,1999)

- Shor's quantum algorithm for factoring.

  – related to Laplaza's theory of coherence for distributivity
  PMH (2013)

- Other proposed algebraic structures (!)

  –T.B.C.