Finding Resolution in unexpected places : Girard's formula in non-logical settings

Peter M. Hines
Y.C.C.S.A. , Univ. York

Marseille – France – 2020

An relevant quote ...

"A PhD thesis is not a life sentence" — Samson Abramsky

. . . explained to mean that, simply because you did your PhD in a particular area, you are not constrained to work in that field forever!

What may, nevertheless, happen :

- tools developed in one field are frequently useful elsewhere.
- particularly significant structures turn up unexpectedly in seemingly unrelated fields.

When we see the same structures in distinct fields,
it is natural to wonder why!

A personal viewpoint

My own thesis:

The algebra of self-similarity & its applications (1997 - 98)

was an algebraic / categorical analysis of J.-Y. Girard's first three Geometry of Interaction papers.

Today's talk :

1. Assumes no particular background, beyond simple category theory,
2. **is definitely not about (linear) logic / λ calculus / GoI,**
3. focuses on where & why similar structures occur in other fields.

A puzzle ...

Why should formulæ introduced to model :

Cut-elimination in Linear Logic (J.-Y. Girard 1994)
Partial isometries on real Hilbert spaces

also have appeared as models of

Transitions of two-way automata (J.-C. Birget 1989)
Relations on finite sets

or indeed

Dynamics of space-bounded Turing machines (PMH 2003)
partial injections, partial functions, relations

never mind

Circuits for quantum computers (PMH 2012)
Unitary maps on complex Hilbert spaces

Some basic structures

The GoI papers were based on bounded linear maps (partial isometries) on separable Hilbert spaces.

Implicitly, working within the category **Hilb**.

As observed by *many people* ...

All the action takes place within the image of M. Barr's $l_2 : \mathbf{pInj} \to \mathbf{Hilb}$ functor,

A faithful contravariant functor

- from the category **pInj** of partial injections on sets,
- to the category **Hilb** of Hilbert spaces & bounded linear maps.

A common claim

It was often claimed that it is easier / simpler to work with partial injections.

The linear-algebraic setting was sometimes referred to as, ‘useless superstructure’.

A contrasting view

Claim : At least for some applications, this “additional structure” is absolutely essential.

About the category **pInj** ..

Objects are all sets, arrows are **partial injections**.

Partial injections :

- A **partial injection** $f : X \to Y$ is a partial function

  that is bijective when restricted to its domain / image.

- Composition is inherited from *partial functions*, and by extension, from the category **Rel** of relations.

pInj is an *Inverse Category* :

Every arrow $a : X \to Y$ has a *unique* generalised inverse $a^\ddagger : Y \to X$ satisfying $aa^\ddagger a = a$ and $a^\ddagger aa^\ddagger = a^\ddagger$.

Partial injections as an inverse category

A couple of key points :

1. The generalised inverse is a categorical dual :

$$(ab)^{\ddagger} = b^{\ddagger}a^{\ddagger} \quad , \quad X^{\ddagger} = X \quad , \quad \left(( )^{\ddagger}\right)^{\ddagger} = Id$$

2. Uniqueness is very powerful. It implies
   - All idempotents commute

$$e^2 = e \text{ and } f^2 = f \Rightarrow ef = fe$$

   - Idempotents 'pass through arrows'. For all $e^2 = e : X \to X$ and $a : X \to Y$

$$ae = fa \quad \text{for some unique } f^2 = f$$

Partial injections as the inverse category

It is the *canonical* inverse category

there is a faithful functor from any inverse category to **plnj**.

- Proved for inverse monoids in 1952-4 : The Wagner – Preston representation theorem for inverse monoids, generalising Cayley's group representation theorem.
- Proved for inverse categories in J. Kastl (1979) – also proved independently by R. Cockett (2002), and C. Heunen (2014)

About Barr's functor ...

M. Barr, Algebraically Compact Functors (1992)

A faithful functor $l_2 : \mathbf{pInj} \to \mathbf{Hilb}$.

In the countable setting, it is convenient to work with a *covariant*, *basis-dependent* version, given by :

Objects $l_2(X)$ is the Hilbert space with orthonormal basis $\{|x\rangle\}_{x \in X}$

Arrows Given $f \in \mathbf{pInj}(X, Y)$,

$$l_2(f)(|x\rangle) = \begin{cases} |f(x)\rangle & x \in dom(f) \\ 0 & \text{otherwise.} \end{cases}$$

Scalar field? Barr is *agnostic* about this!

About Barr's functor ...

M. Barr, Algebraically Compact Functors (1992)

pInj	**Hilb**
Countable Sets	separable Hilbert Spaces
Partial Injections	Partial Isometries
Inclusion Ordering	Halmos - McLaughlin ordering
Bijections	Unitaries
Disjoint union $\_ \uplus \_$	Direct sum $\_ \oplus \_$
Cartesian product $\_ \times \_$	Tensor product $\_ \otimes \_$
Unions of (disjoint) sets $\bigvee_{j \in J}(\ )$	Sums of linear maps $\Sigma_{j \in J}(\ )$
The categorical trace	The Resolution formula

Categorical traces (I)

"Traced Monoidal Categories" — A. Joyal, R. Street, D. Verity (1996)

In a symmetric monoidal category $(\mathcal{C}, \_ \otimes \_)$, a trace is an object-indexed mapping of hom-sets

$$\left\{ Tr^{U}_{X,Y} \ : \ \mathcal{C}(X \otimes U \ Y \otimes U) \to \mathcal{C}(X, Y) \right\}_{X,Y,U \in Ob(\mathcal{C})}$$

that is natural in X, Y, and dinatural in U.

It has the intuition of, "eliminating the behaviour of an arrow at an object".

Categorical traces (II)

As well as (di-)naturality, we have :

Vanishing I $Tr^{I}_{-,-}(f) = f$

Superposing Given $f : X \otimes U \to Y \otimes U$ and $g : A \to B$

$$Tr^{U}_{-,-}(f \otimes g) = Tr^{U}(f) \otimes g$$

Vanishing II Given $f : X \otimes P \otimes Q \to Y \otimes P \otimes Q$,

$$Tr^{P \otimes Q}(f) \;=\; Tr^{Q}\left(Tr^{P}(f)\right) \;=\; Tr^{P}\left(Tr^{Q}(f)\right)$$

– also known as the **confluence** axiom –

A brief categorical logic interlude ...

models of untyped systems.

– not relevant for the rest of the talk!

A well-known application of traces ...

The construction of a **compact closed** category from a *traced monoidal* category.

- The **Int** construction ofJoyal, Street & Verity (1996)
- The **GoI** construction of S. Abramsky (1996)

– Equivalent, but not identical.

Their correspondence given by E. Haghverdi (2000)

Compact closed categories

A **compact closed category** is a symmetric monoidal category $(\mathcal{C}, \_ \otimes \_, I)$ with a dual

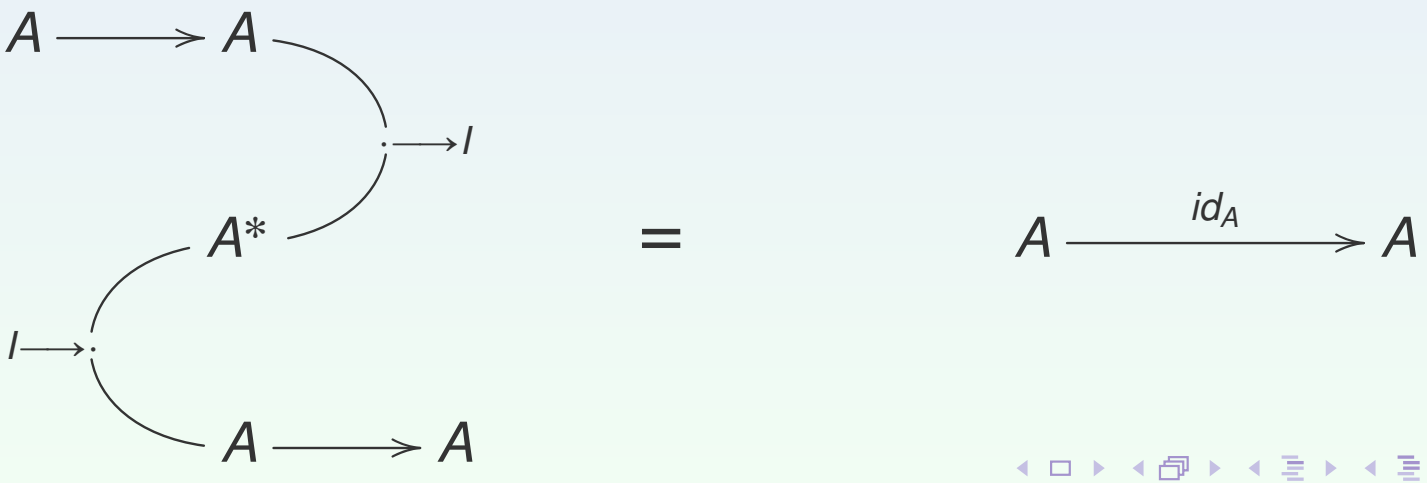
$$(\_)^* : \mathcal{C}^{op} \to \mathcal{C}$$

and distinguished arrows for every object $A \in Ob(\mathcal{C})$

- The unit arrow $\epsilon_A : A \otimes A^* \to I$
- The counit arrow $\eta_A : I \to A^* \otimes A$

that satisfy

$$(\epsilon_A \otimes 1_A)(1_A \otimes \eta_A) = 1_A \quad \text{and dually,} \quad (1_{A*} \otimes \epsilon_A)(\eta_A \otimes 1_{A*}) = 1_{A*}$$



From categories to monoids

The best-known models of untyped systems :

J. Lambek & P. Scott (1986) modeled untyped λ-calculus using C**-monoids** :

Single-object categories that satisfy

> *"All the axioms for a Cartesian closed category, except for the terminal object"*

In GOI(I), Girard comments that his system 'forgets types'.

Q : Can we do similar, for *compact* rather than *Cartesian* closure?

A : Yes, and it is much simpler than constructing C-monoids!

Short steps to untyped closure

Step (I)

In a compact closed category $\mathcal{C}, \_ \otimes \_$, the internal hom is given by

$$[A \to B] \cong A^* \otimes B$$

As a corollary : for any object $A \in Ob(\mathcal{C})$ The full monoidal subcategory generated by $\{A, A^*\}$ is compact closed.

Short steps to untyped closure

Step (II)

In the special case where A is a self-dual object, $A \cong A^*$,

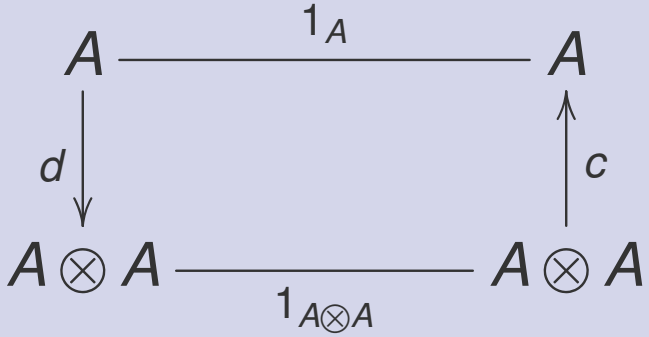
the full monoidal subcategory generated by $\{A\}$ is compact closed.

We then have a monogenic compact closed category.

Short steps to untyped closure

Step (III)

Now assume A is also **self-similar** — there exist Hilbert-hotel style bijections



All objects of the compact closed subcategory generated by $\{A\}$ are isomorphic (apart from the unit object ...) .

In particular, we have $A \cong [A \to A]$.

Some historical context

In (PMH 97-98) and (PMH 2000) this property was used to claim that :

> *The endomorphism monoid of a self-dual, self-similar object in a compact closed category is a 'compact closed monoid'.*

This was based on an axiomatisation of compact closure that

1. didn't mention the unit object
2. reduced to the usual definition in the presence of a unit.

– compact closure in semi-monoidal categories.

A problem & a solution

The problem :

There are different, strictly inequivalent, axiomatisations of compact closure that

1. don't mention the unit object
2. reduce to the usual definition in the presence of a unit.

The solution :

A coherence theorem / strictification procedure that gives an *equivalence of categories* between semi-monoidal *monoids* and more general semi-monoidal *categories*.

"Coherence & Strictification for Self-Similarity" — Journal of Homotopy & Related Structures (PMH 2017)

End of the categorical logic interlude ...

– back to traces & resolution!

A motivating example

Joyal, Street, & Verity gave, as a motivating example,

the category $(\mathbf{Rel}, \_ \uplus \_)$ of relations, with disjoint union :

Objects	All sets
Arrows	Relations between sets
Composition	The usual relational composition
Monoidal tensor	Disjoint union $X \uplus Y = X \times \{0\} \cup Y \times \{1\}$

A matrix calculus for relations

A relation $\rho : P \uplus Q \to R \uplus S$ may be given in matrix form.

$$\rho = \begin{pmatrix} a & b \\ c & d \end{pmatrix} : P \uplus Q \to R \uplus S$$

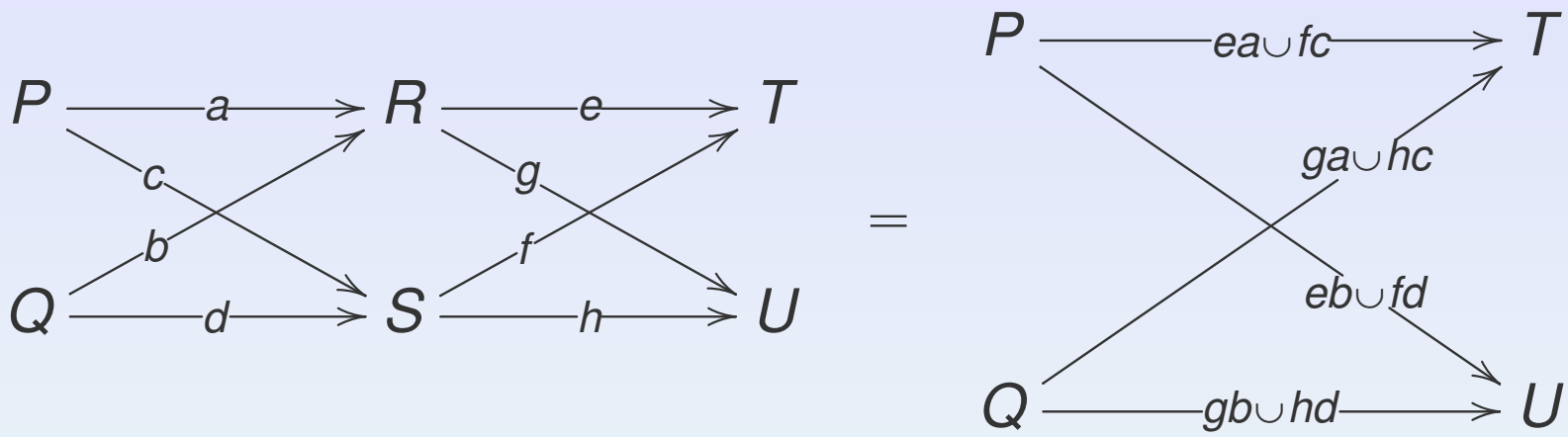
simply by taking projections

$$a : P \to R \qquad b : Q \to R$$

$$c : P \to S \qquad d : Q \to S$$

Composing matrices of relations

Composition is given by the usual ‘summing over paths’:



with *addition* replaced by *union*.

$$\begin{pmatrix} e & f \\ g & h \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} ea \cup fc & eb \cup fd \\ ga \cup hc & gb \cup hd \end{pmatrix}$$

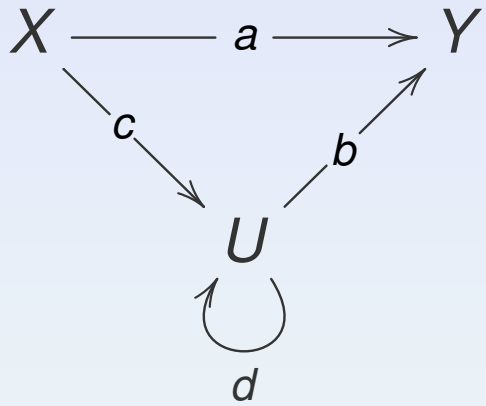
The (particle-style) trace of a relation

Given $\eta = \begin{pmatrix} a & b \\ c & d \end{pmatrix} : X \uplus U \to Y \uplus U$

The trace

$$Tr^U_{X,Y}(\eta) : X \to Y$$

is another 'summing over paths' construction.



As an explicit formula:

$$Tr^U_{X,Y}(\eta) \;=\; a \;\cup \bigcup_{j=0}^{\infty} b d^j c$$

Interesting traced subcategories :

Several interesting monoidal subcategories of $(\mathbf{Rel}, \_ \uplus \_)$ both :

1. admit a matrix calculus
2. are closed under this trace :

- $(\mathbf{pInj}, \_ \uplus \_)$ *partial injections*
- $(\mathbf{pFun}, \_ \uplus \_)$ *partial functions*
- $(\mathbf{Bij_{fin}}, \_ \uplus \_)$ *bijections on finite sets*
- $(\mathbf{sRel}, \_ \uplus \_)$ *stochastic relations*
- $(\mathbf{pInv}, \_ \uplus \_)$ *partial involutions*

The ‘particle-style’ traces

Girard's original setting

The original Resolution formula required :

- A partial symmetry – i.e. $\sigma : \mathbb{N} \to \mathbb{N}$ such that σ^2 is idempotent

$$\sigma^2 \;=\; 1_Y \quad \text{for some} \quad Y \subseteq \mathbb{N}$$

- A partial injection $P : \mathbb{N} \to \mathbb{N}$ that "includes" σ.

Resolution was given by taking the trace

$$Tr^Y(P) \in \mathbf{pInj}(\mathbb{N}\backslash Y, \mathbb{N}\backslash Y)$$

up to the re-embedding $\mathbf{pInj}(\mathbb{N}\backslash Y, \mathbb{N}\backslash Y) \hookrightarrow \mathbf{pInj}(\mathbb{N}, \mathbb{N})$.

A slightly more general setting

We will refer to the composite of a (particle-style) trace

$$Tr^U_{X,X} \; : \; \mathcal{C}(X \uplus U, X \uplus U) \; \mapsto \; \mathcal{C}(X, X)$$

together with a canonical inclusion

$$\mathcal{C}(X, X) \; \hookrightarrow \; \mathcal{C}(X \uplus U, X \uplus U)$$

as the **Resolution**, and write

$$Res^U : \mathcal{C}(X \uplus U, X \uplus U) \to \mathcal{C}(X \uplus U, X \uplus U)$$

What has changed :

We have lost :

- The faithful functor $I_2 : (\mathbf{pInj}, \uplus) \to (\mathbf{Hilb}, \oplus)$
- The rôle of the partial symmetry σ

– although these both re-appear in various settings!

A precursor to the $3n+1$ problem

It is *easy* to express hard / uncomputable problems in this setting :

The Original Collatz Conjecture (OCC)

A (still unsolved) problem from unpublished notes (1932) of L. Collatz

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

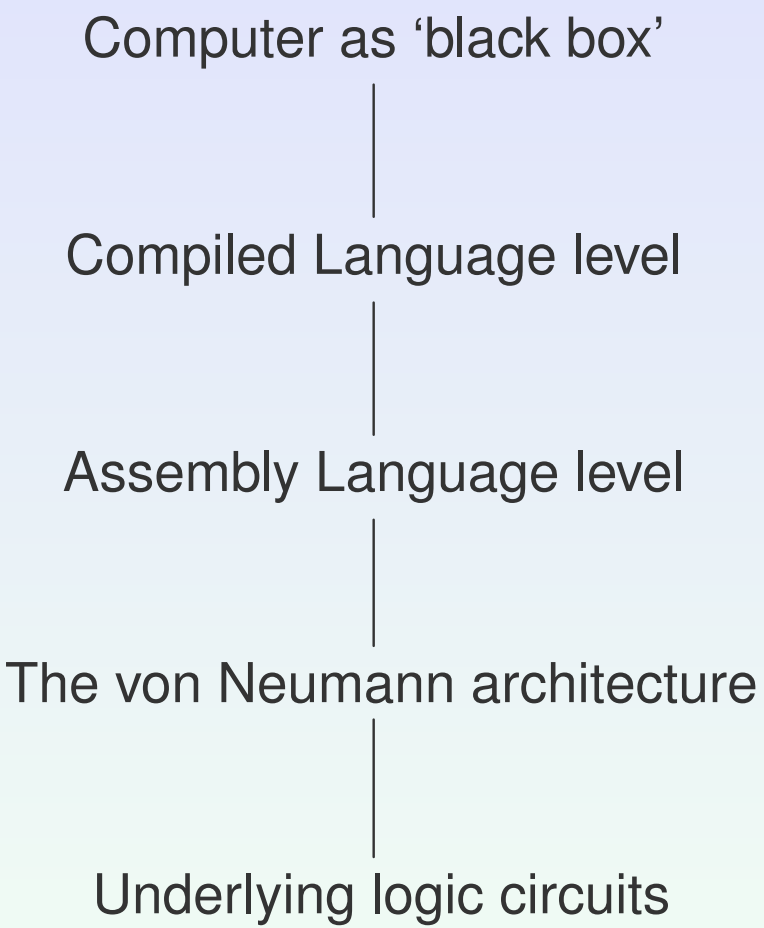
Conjecture : “The orbit of 8 is unbounded”.

$$Res^{\mathbb{N}\setminus\{8\}}(g) = \begin{cases} 0 & \text{O.C.C. is True} \\ I_{\{8\}} & \text{O.C.C. is False} \end{cases}$$

A (very) general setting for Resolution

The starting point

D. Hofstadter's book "Gödel, Escher, Bach" (1979)



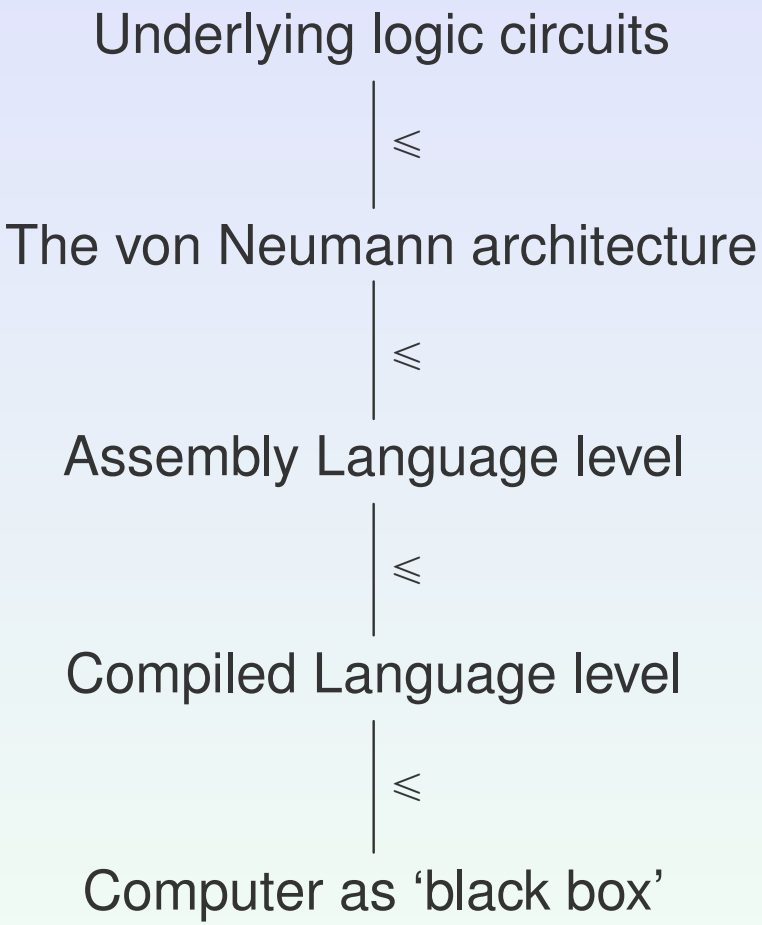
High vs. Low level

The same computation, described at different levels of generality.

Moving between levels was called **chunking** *by D. Hofstadter ...*

The starting point

D. Hofstadter's book "Gödel, Escher, Bach" (1979)



Low vs. High level

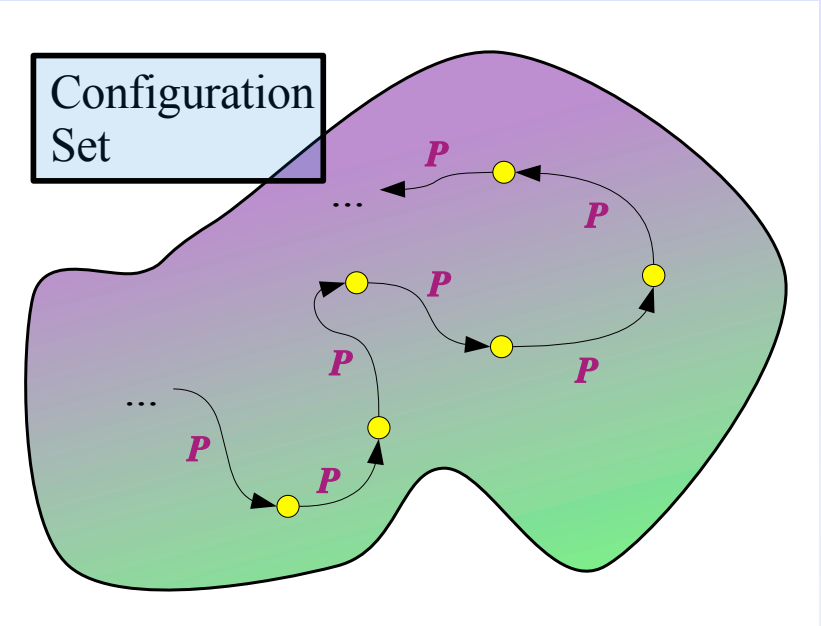
The same computation, described at different levels of generality.

Moving between levels was called **chunking** *by D. Hofstadter ...*

Moving to Unlabeled Transition Systems

An abstract machine $\mathcal{M}$ is :

- A **configuration set** X.
- The **primitive evolution**, a partial function $\mathcal{P} : X \to X$.



Structural Operational Semantics – Gordon Plotkin (1981 / 2004)

- "This idea is hardly new, and examples can be found in any book on automata or formal languages."
- "This definition is too general to produce any useful theory."

Partial views of dynamics

A relation between partial functions :

The '**primitiveness**' relation

For any two partial functions $\eta, \mu : X \to X$

$$\mu \prec \eta \quad \text{iff} \quad \mu \subseteq \eta^+ = \bigcup_{j=1}^{\infty} \eta^j$$

Read this as 'μ is **a representation of** η'.

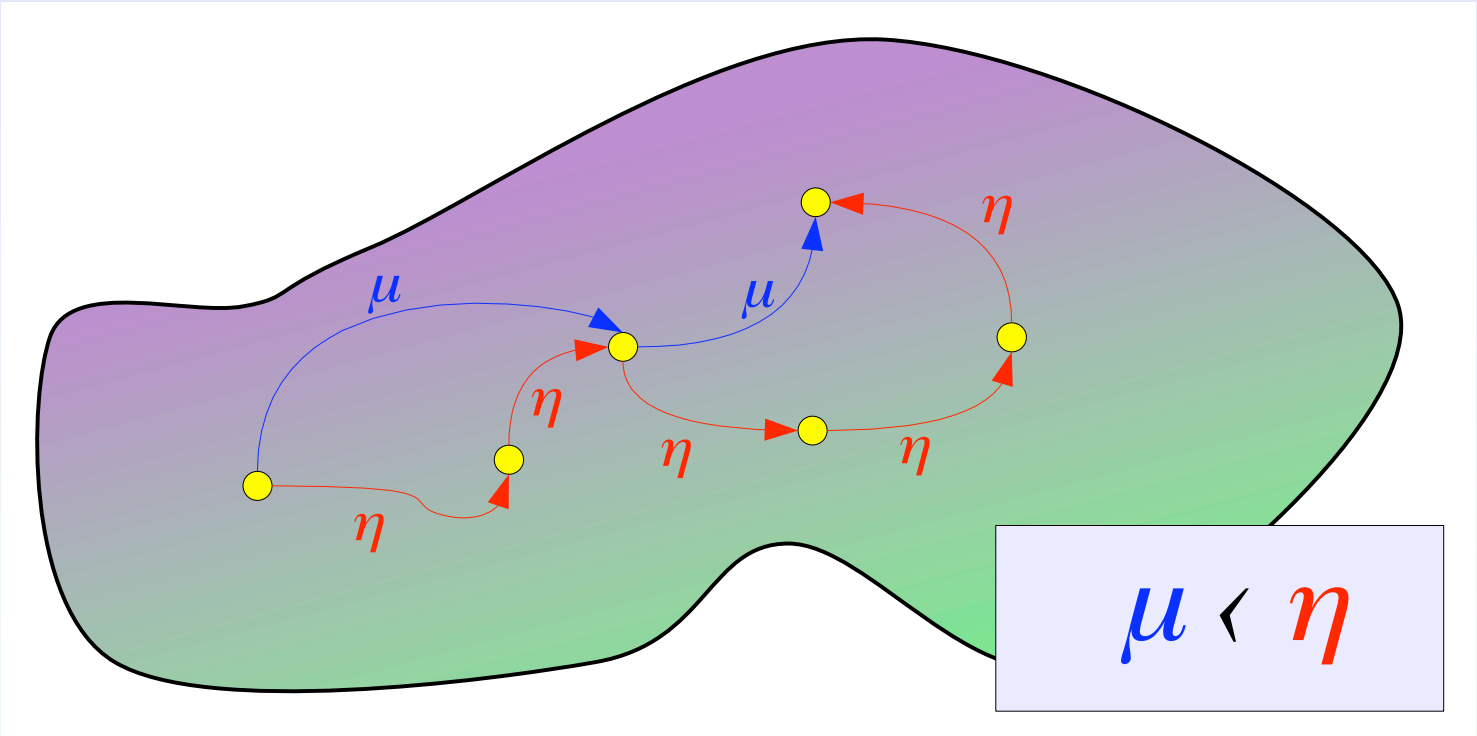
Operationally, this states that:

$$\mu(x) = y \quad \Rightarrow \quad \eta^n(x) = y \quad \text{for some } n > 0 \in \mathbb{N}$$

Interpreting the **primitiveness** relation

Motivation

η is *'finer-grained'*, *'more complete'*, or *lower level* than μ.



A definition

Let $\mathcal{M} = (X, \mathcal{P} : X \to X)$ be an abstract machine.

Its **machine semantics** is the set of all representations of $\mathcal{P}$.

$$[\mathcal{P}] = \{\eta : X \to X \;\; s.t. \; \eta \prec \mathcal{P}\}$$

The interpretation

$[\mathcal{P}]$ is the collection of *all possible* descriptions of the behavior of the machine $\mathcal{M}$, at different levels of generality.

A semigroup with a (pre-)order

For a given abstract machine $(X, \mathcal{P})$, the machine semantics $[\mathcal{P}]$ is a semigroup. The 'primitiveness' relation is a pre-order on this semigroup.

- Closure under composition is straightforward.
- The obstacle to $\prec$ being a *partial order* is cyclic behaviour. In general, $\eta \prec \mu$ and $\mu \prec \eta$ does not imply $\eta = \mu$.
- The Interaction of composition with $\prec$ is non-trivial.

A special case:

An abstract machine $\mathcal{M} = (X, \mathcal{P})$ is **cycle-free** when

$$P^N(x) \neq x \quad \forall\, x \in X\,,\; N > 0$$

This is implied by **nilpotency**:

$$P^K = 0_X \quad \text{for some} \quad K \geqslant 0$$

and is equivalent to nilpotency when X is finite.

For a cycle-free machine:

$[\mathcal{P}], <$ is a lattice,

- The **top** element is $\mathcal{P}$, the *primitive evolution*,
- The **bottom** element is 0_X, the *nowhere-defined function*.

What sort of lattice is this?

Meets of representations

Given a **finite** subset $\{\eta_i\} \subseteq [\mathcal{P}]$, all defined at x.

$$\eta_i(x) = \mathcal{P}^{N_i}(x) \text{ for unique } N_i$$

The meet is defined in terms of a *least common multiple* :

$$\left(\bigwedge \eta_i\right)(x) = \mathcal{P}^{lcm(N_i)}(x)$$

As $\{\eta_i\}$ is **finite**, this always exists.

What sort of lattice is the machine semantics ??

Joins of representations

Given an arbitrary subset $\{\mu_j\} \subseteq [\mathcal{P}]$, all defined at x.

$$\mu_j(x) = \mathcal{P}^{M_j}(x) \quad \text{for unique} \quad M_j$$

The join is defined in terms of a *greatest common divisor* :

$$\left(\bigvee \mu_j\right)(x) = \mathcal{P}^{gcd(M_j)}(x)$$

This **always exists**, for finite or infinite subsets.

The machine semantics is a locale or pointless topology, with an additional semigroup structure.

The general case!

For **arbitrary** abstract machines

We define the **cycle-free semantics**

The subset $[\![\mathcal{P}]\!] \subseteq [\mathcal{P}]$ of all *cycle-free representations*:

$$\eta^N(x) \neq x \quad \forall\, N > 0$$

This need not include $\mathcal{P}$ itself.

As intended:

- $[\![\mathcal{P}]\!], \prec$ is a partial order, with 0_X as bottom element.
- It is not, in general, a lattice or a semigroup.

What sort of poset is $[\![\mathcal{P}]\!], \prec$?

The correct setting is *domain theory*

Assuming a countable configuration set, we have:

- A bottom element.
- Directed-completeness.
- Every element is the supremum of a chain of elements with finite support.
- The down-closure of every element is a *locale*.

it is in fact a Scott Domain

Chains and directed sets:

A **chain** is a totally ordered subset

$$c_0 \leqslant c_1 \leqslant c_2 \leqslant c_3 \leqslant \dots$$

Chain-completeness: every chain has a supremum.

A subset D is **directed**, when For all $a, b \in D$, there exists $c \in D$ such that

$$a \leqslant c \quad \text{and} \quad b \leqslant c$$

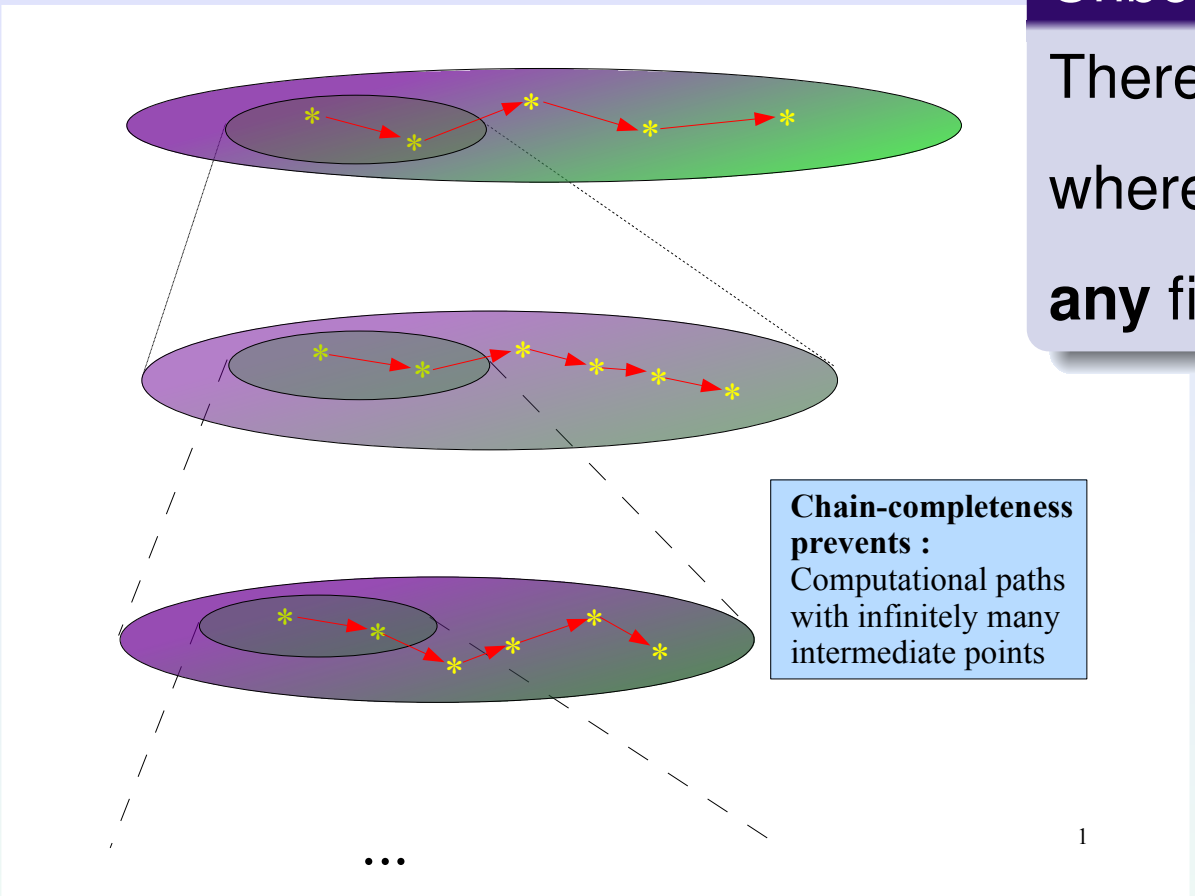
Directed-completeness: all directed sets have suprema.

Iwamura's Lemma (1944)

Chain-completeness is equivalent to directed-completeness.

... at least, provided we accept the Axiom of Choice.

Chain-completeness & Zeno's paradox ?



Unbounded chains implies :

There exists some $y\ \mathcal{P}^+\ x$,

where $\mathcal{P}^N(x) \neq y$ for

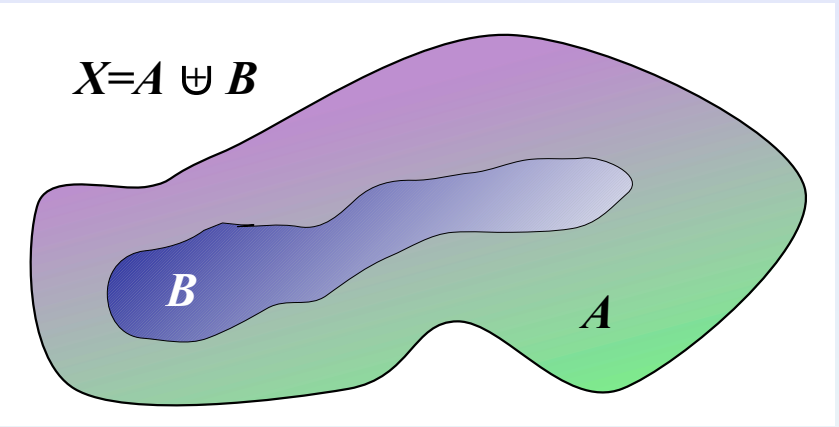
any finite N.

Finding Resolution again ..

We may treat the configuration set as the disjoint union of two subsets $X = A \uplus B$:

Every representation can be written as a **matrix**

$$\eta = \begin{pmatrix} \eta_{11} & \eta_{12} \\ \eta_{21} & \eta_{22} \end{pmatrix}$$



$$\eta_{11} : A \to A \qquad \eta_{12} : B \to A$$

$$\eta_{21} : A \to B \qquad \eta_{22} : B \to B$$

No resolution without representation

Some key points :

For arbitrary $\eta : X \to X$ and $B \subseteq X$,

- $Res^B(\eta) \prec \eta$
- η is cycle-free $\Rightarrow Res^B(\eta)$ is cycle-free.
- $\mu \prec Res^B(\eta)$ for all $\mu \prec \eta$ satisfying $\mu(B) = \varnothing$,

When dealing with cycle-free / nilpotent representations, $Res^B(\ )$ calculates suprema.

A more ‘concrete’ interpretation

When we have a distinguished subset of **start-halt** or **input-output** configurations, we may use Resolution to move

- from a step-by-step, or operational, description of dynamics,
- to a start-halt / input-output, or denotational description of dynamics

in a ‘fine-grained’ manner.

Particularly important :

The ‘confluence’ property

$$Res^{U}\left(Res^{V}(\_)\right) \;=\; Res^{U\cup V}(\_) \;=\; Res^{V}\left(Res^{U}(\_)\right)$$

allows this to happen in a local & asynchronous manner.

A worked example:

Eliminating the description of the behaviour everywhere, except for the start & halt states.

An example, from ‘Machine Semantics’ (TCS 2010)

Deriving the start-halt behaviour for

space-bounded Turing machines

from their step-by-step behaviour

using the Resolution.

The resulting models were **compositional**. Behaviour on a tape of length $r + s$ is derived from behaviour on tapes of length r and s respectively.

Re-introducing the linear structure

Let us use Barr's $l_2 : \mathbf{pInj} \to \mathbf{Hilb}$ to re-introduce the linear structure, with particular reference to the traced subcategory $\mathbf{Bij_{fin}}$ of bijections on finite sets.

(Recall : l_2 maps **bijections** to **unitary maps**)

The question :

Can we map

unitary step-by-step behaviour

to

unitary start-halt behaviour ?

Let us assume the underlying scalar field is the complex plane $\mathbb{C}$.

A question of superposition

Given a unitary map $U = l_2(P) : l_2(X) \to l_2(X)$, this is linear and inner-product preserving.

$$U\left(\sum_{j\in J} \alpha_j \left|x_j\right\rangle\right) = \sum_{j\in J} \alpha_j U\left(\left|x_j\right\rangle\right)$$

Can we derive a unitary that maps "start-states" to corresponding "halt-states",

$$\sum_{j\in J} \beta_j \left|Start_j\right\rangle \mapsto \sum_{j\in J} \beta_j \left|Halt_j\right\rangle$$

in a superposition-preserving, or 'coherent' manner?

Some history from the foundations

of quantum computation

Reversible computation – a precursor

The logical reversibility of computation — Bennett (1973)

- Computation by *reversible Turing machines*.
- Dynamics were reversible *at each step*.
- Temporary ‘storage’ is returned to its original state.

An original aim was to build a unitary version of this.

(Implicitly, applying Barr’s I_2 functor!)

QM Turing machines ??

Quantum Turing machines

- Introduced by D. Deutsch (1985).

  Quantum theory, the Church-Turing principle and the universal quantum computer

- Defined to be *fully quantum* (i.e. coherent),

  "A superposition of start states leads to the corresponding superposition of halt states."

The dominant paradigm in QM computing for several years

Only 12 years later ...

Studied by J. Myers (1997)

Can a Universal Computer be Fully Quantum?

His claim:

QTMs are incapable of ‘fully quantum’ computations

(even when they act classically on some fixed basis!)

The problem with Quantum Turing machines

Consider

- A QTM has a state space, instead of a state set.
- It has unitary dynamics.
- We can also have superpositions of states ...
  including 'halting states' and non-halting states'.

Myer's question:

What happens 'after halting'?

When the QTM is in a superposition:

$$|\mathit{halting\ state}\rangle + |\mathit{non-halting\ state}\rangle$$

What should happen next?

What did happen next(!)

- (1997) Bernstein and Vazirani: QTMs are fully quantum when the *number of steps to halting* is independent of the input.
- (1998) Ozawa: Arbitrary QTMs can be made *fully quantum*.
- (1998) Linden & Popescu: "*Ozawa's halting scheme is not fully quantum, except in the trivial case in which the computer never halts.* "
- (1998) Ozawa publishes a rebuttal of Linden & Popescu
- (1998) This is shown to be incorrect by Kieu & Danos

Every proposed solution relied on an ancillary space

In every case, this became irretrievably entangled with the output.

Why is this a problem??

Elementary quantum information (I)

Consider a state in the space $l_2(\{0,1\}^{\times 5})$ (the **5-qubit** space).

$$\alpha\,|01001\rangle + \beta\,|10010\rangle \qquad |\alpha|^2 + |\beta|^2 = 1$$

This is **entangled** – it cannot be factorised as the tensor product of a 4-qubit state and a 1-qubit state.

The crucial question :

Is there any way to derive the 4-qubit state :

$$\alpha\,|0100\rangle + \beta\,|1001\rangle \qquad |\alpha|^2 + |\beta|^2 = 1$$

Elementary quantum information (II)

What happens when we measure the final qubit of

$$\alpha\,|01001\rangle + \beta\,|10010\rangle \qquad |\alpha|^2 + |\beta|^2 = 1$$

- We observe '1' with probability $|\alpha|^2$.
  - leaving the remaining state as $|0100\rangle$
- We observe '0' with probability $|\beta|^2$.
  - leaving the remaining state as $|1001\rangle$

Elementary quantum information (II)

What happens when we -do not- measure the final qubit of

$$\alpha\,|01001\rangle + \beta\,|10010\rangle \qquad |\alpha|^2 + |\beta|^2 = 1$$

> *"We take the physical system represented by the final qubit, and 'lock it away' somewhere we are sure it will never be measured ..."*

This is equivalent to :

- We observe '1' with probability $|\alpha|^2$.
  - leaving the remaining state as $|0100\rangle$
- We observe '0' with probability $|\beta|^2$.
  - leaving the remaining state as $|1001\rangle$

. . . followed by *forgetting the value we observed*.

Alternatively ... without entanglement

We want:

the 5-qubit state

$$\alpha\,|01001\rangle + \beta\,|10010\rangle$$

We actually have:

The 6-qubit state

$$\alpha\,|010011\rangle + \beta\,|010011\rangle$$

Not a problem: $(\alpha\,|01001\rangle + \beta\,|01001\rangle) \otimes |1\rangle$

The desired register, is *not entangled* with the additional qubit.

We may measure (or simply ignore) the final qubit.

A disturbingly popular non-solution

- Introduce an ancillary **clock space**, with basis

$$\{|t_0\rangle\ ,\ |t_1\rangle\ ,\ |t_2\rangle\ ,\ |t_3\rangle\ ,\ \ldots\}$$

- After reaching a halt state: *'The Turing Machine computation stops, and the ancilla increments at each step'*.

What then happens:

- State $|start_1\rangle|0\rangle$ reaches $|halt_1\rangle|0\rangle$ in T_1 steps
- State $|start_2\rangle|0\rangle$ reaches $|halt_2\rangle|0\rangle$ in $T_2 < T_1$ steps

$$|start_1\rangle|0\rangle + |start_2\rangle|0\rangle \implies |halt_1\rangle|0\rangle + |halt_2\rangle|T_2 - T_1\rangle$$

Two relevant results:

Linden & Popescu Rephrased the problem as a *general problem* about algorithms:

> *We need to restructure algorithms so that the time to halting is independent of the input. It is* unknown *whether this is always possible.*

Bernstein & Vazirani

> *Computations that can be performed by 'properly halting QM Turing machines' are precisely those that can be performed in the circuit paradigm.*

The circuit model became the dominant paradigm in QM computing.

Programming paradigms from the 1920s

Even when working with functions that map basis states to basis states, this is not easy to deal with.

The programmers view ...

We don't get to use any of our favorite tools, such as *recursion*, *fixed point operations*, *the von Neumann architecture*, *loops*, *conditional iteration*, *feedback*, etc.

What would be a useful tool :

A 'compilation' of Resolution into quantum circuits.

An interesting Observation

Taken from :

Categorical Traces from single-photon linear optics

PMH, P. Scott (Clifford Lectures, P.S.A.M. 2012)

(A thought-experiment based on the
Sagnac interferometer / ring laser gyroscope)

The key observation

In a monoidal category with *matrix representations* of arrows:

Consider an arrow:

$$F = \begin{pmatrix} A & B \\ C & D \end{pmatrix} : U \oplus U \to U \oplus U$$

What happens when we compose the following matrices?

$$\begin{pmatrix} B & A & 0 & 0 & 0 & 0 & \cdots \\ D & C & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & I & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & I & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & I & 0 & \cdots \\ 0 & 0 & 0 & 0 & 0 & I & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

The key observation

In a monoidal category with *matrix representations* of arrows:

Consider an arrow:

$$F = \begin{pmatrix} A & B \\ C & D \end{pmatrix} : U \oplus U \to U \oplus U$$

What happens when we compose the following matrices?

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & B & A & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & D & C & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

The key observation

In a monoidal category with *matrix representations* of arrows:

Consider an arrow:

$$F = \begin{pmatrix} A & B \\ C & D \end{pmatrix} : U \oplus U \to U \oplus U$$

What happens when we compose the following matrices?

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & B & A & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & D & C & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

The key observation

In a monoidal category with *matrix representations* of arrows:

Consider an arrow:

$$F = \begin{pmatrix} A & B \\ C & D \end{pmatrix} : U \oplus U \to U \oplus U$$

What happens when we compose the following matrices?

$$\begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & B & A & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & D & C & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

Through tedious calculations !!

After T steps, we build up the *summands* of:

- The Elgot dagger
- The Resolution

$$\begin{pmatrix}
B & A & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots \\
BD & BC & A & 0 & 0 & \dots & 0 & 0 & 0 & \dots \\
BD^2 & BDC & BC & A & 0 & \dots & 0 & 0 & 0 & \dots \\
BD^3 & BD^2C & BDC & BC & A & \dots & 0 & 0 & 0 & \dots \\
BD^4 & BD^3C & BD^2C & BDC & BC & \dots & 0 & 0 & 0 & \dots \\
\vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \\
BD^{T-1} & BD^{T-2}C & BD^{T-3}C & BD^{T-4}C & BD^{T-5}C & \dots & A & 0 & 0 & \dots \\
D^T & D^{T-1}C & D^{T-2}C & D^{T-3}C & D^{T-4}C & \dots & C & 0 & 0 & \dots \\
0 & 0 & 0 & 0 & 0 & & 0 & I & 0 & \dots \\
0 & 0 & 0 & 0 & 0 & & 0 & 0 & I & \\
\vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & & \ddots
\end{pmatrix}$$

Let's call this the **H-S matrix**.

Rather conveniently :

Matrices of this form are **absolutely standard** in QM computation.

They arise via :

- simple logic gates $NOT\,|0\rangle = |1\rangle \quad NOT\,|1\rangle = |0\rangle$
- simple arithmetic functions

$$Succ\,|k\rangle = \left|k + 1\ (mod\ 2^N)\right\rangle$$

- QM conditionals, such as 'controlled-NOT'

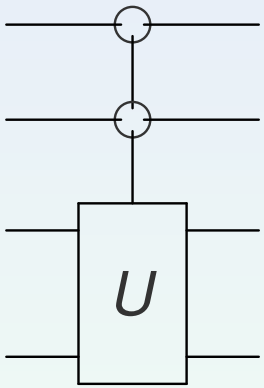
$$CNOT\,|00\rangle = |01\rangle \quad CNOT\,|01\rangle = 10$$

$$CNOT\,|10\rangle = |10\rangle \quad CNOT\,|11\rangle = 11$$

Matrices for multiply-controlled gates

$$U = \begin{pmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{pmatrix} \quad \text{Controlled on } 00$$

The circuit:



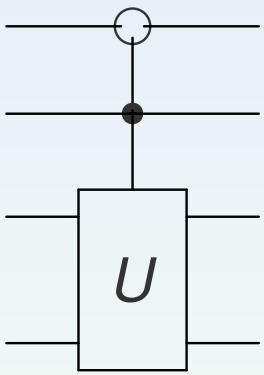
The matrix:

$$\begin{pmatrix} U_{00} & U_{01} & 0 & 0 & 0 & 0 & 0 & 0 \\ U_{10} & U_{11} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Matrices for multiply-controlled gates

$$U = \begin{pmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{pmatrix} \quad \text{Controlled on } 01$$

The circuit:



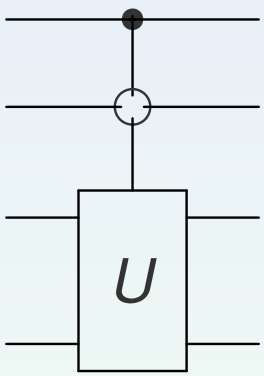
The matrix:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} & 0 & 0 & 0 & 0 \\ 0 & 0 & U_{10} & U_{11} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Matrices for multiply-controlled gates

$$U = \begin{pmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{pmatrix} \quad \text{Controlled on } 10$$

The circuit:



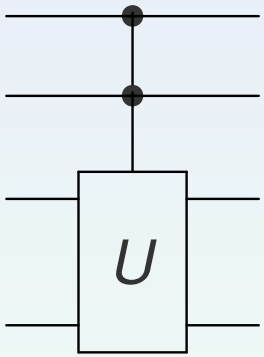
The matrix:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & U_{00} & U_{01} & 0 & 0 \\ 0 & 0 & 0 & 0 & U_{10} & U_{11} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Matrices for multiply-controlled gates

$$U = \begin{pmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{pmatrix} \quad \text{Controlled on } 11$$

The circuit:



The matrix:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & 0 & 0 & 0 & 0 & U_{10} & U_{11} \end{pmatrix}$$

In terms of states / functions / mappings

What is the action of the H-S matrix?

Consider an Abstract Machine $(X, \mathcal{P})$,
that maps a *start configuration* $s_j \in X$ to a *halt configuration* h_j.

$$h_j \;=\; Res(\mathcal{P})(s_j) \;=\; \mathcal{P}^{T_j}(s_j)$$

Now apply the l_2 functor, and work with Hilbert spaces.

The action of the H-S matrix on a *start basis vector* is

$$|n\rangle\,|s_j\rangle \;\mapsto\; |n + T_j\rangle\,|h_j\rangle$$

On a superposition of start states, we get :

$$\sum_{j\in J} |n\rangle\,|s_j\rangle \;\mapsto \sum_{j\in J} |n + T_j\rangle\,|h_j\rangle$$

Just another 'clock space' solution??

Note the re-appearance of

- The infamous clock space — an ancilla with basis $\{|0\rangle, |1\rangle, |2\rangle, \ldots\}$
- Massive entanglement, for superpositions of start states!

$$\sum_{j \in J} |0\rangle |s_j\rangle \mapsto \sum_{j \in J} |T_j\rangle |h_j\rangle$$

Is this – uncontrollable – entanglement?

Starting from a different point!

Let us take this massively entangled state

$$\sum_{j\in J} |0\rangle |s_j\rangle \mapsto \sum_{j\in J} |T_j\rangle |h_j\rangle$$

and re-run the entire procedure ...

This time, we use the abstract machine $(X, \mathcal{P}^{-1})$ (and, if necessary, swap the rôle of start and halt states).

The action becomes :

$$\sum_{j\in J} |T_j\rangle |h_j\rangle \mapsto \sum_{j\in J} |2T_j\rangle |s_j\rangle$$

Up to a simple division by two, we have :

> *"Our original superposition of start states, with each one entangled with the number of steps a computation on it will require."*

The final step

Starting from

$$\sum_{j \in J} |T_j\rangle |s_j\rangle$$

we re-run our original computation, with the clock *decreasing* instead of *increasing*[1].

We arrive at

$$\sum_{j \in J} |0\rangle |h_j\rangle \;=\; |0\rangle \otimes \sum_{j \in J} |h_j\rangle$$

Leaving the superposition of halt states entirely unentangled with any ancilla.

[1] This happens by composing the matrices that make up the H-S matrix, in the opposite order.

Missing a few details :

Full circuits given in :

Quantum Circuits for Abstract Machine Computations
Theoretical Computer Science (PMH 2012)

Also available as `QM_circuits_for_coherent_AMs.pdf`

As a curious point :

The full circuits require — in a fundamental way — precisely the ‘partial symmetries’ we see in the original Resolution formula.

? Quo Vadis ?

– where are we going with this?

– what exactly do we have?

– what can we do with it?

Some important points!

This is **not** a return to Deutsch's QM Turing machines!

1. It is about iterated computations that *act classically*

   — map basis states to basis states

   (i.e. live within the image of Barr's $\updownarrow_2$).

2. We still require an *upper bound* to the number of steps.

3. Even on basis vectors, non-termination, undecidability, universal computation, etc. remain *well out of reach*!

We are 'compiling' classical iterated computations into coherent QM circuits. This is still within the circuit paradigm.

What we do not have!

An Important Point

This is not about

constructing quantum algorithms.

It is about

constructing inputs to quantum algorithms.

An appealing viewpoint

QM algorithms work by :

- Taking as input, some *classical computation*.
- Returning as output, some *global information* about it.

A few classic examples

- The **Deutsch-Jozsa algorithm** can distinguish a *constant* and a *balanced* classical function.
- **Grover's algorithm** finds the *input* to a classical function that leads to a distinguished *output*.
- **Quantum period-finding** computes the period of a (periodic) classical function.
- **Quantum counting** finds the number of inputs to a classical function that give a distinguished output.
- **Shor's algorithm** Applies period-finding to modular exponentiation.

In every case :

The ‘classical function’ appears as a QM circuit that :

- maps basis states to basis states,
- is superposition-preserving.

Shor’s factorisation algorithm simply :

> *“conjugates a coherent circuit for modular exponentiation by a circuit for the QM Fourier transform”.*

What we have ..

We can translate

– classical (reversible) computations defined by iteration

into

– coherent quantum circuits.

This may happen **if and only if** we can set an upper limit to the number of steps before halting.

Important

Even a single branch of a superposition that does not reach a halt state in time acts as a poison pill that destroys coherence for the entire computation.

This may look familiar

A relevant example

Any superposition including $|8\rangle$ is likely to cause decoherence for a computation based on the original Collatz operator

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

An even richer class of examples is found in :

- J. Conway (1972) " Unpredictable Iterations"
- E. Lehtonen (2008) "Two undecidable variants of Collatz's problem"

where we cannot even predict which inputs will cause problems.

Not what we can, but what we cannot do

Our ultimate objective :

To find hard problems that are *not amenable to a quantum attack*, simply because it is not possible to create a suitable coherent circuit for use with (for example) QM period-finding.

Possibly relevant – a purely algebraic study of potential arithmetic functions:

The York mathematics Semigroups Seminars

https://www-users.york.ac.uk/ varg1/Arithmetic.pdf