Inverse Semigroups and their applications

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

York Mathematics – Algebra Seminar

Aims & Objectives

An introduction to inverse semigroup theory

— Elementary definitions & theory

from the point of view of applications.

— Where we find these structures in

- theoretical & practical computer science
- other areas of mathematics

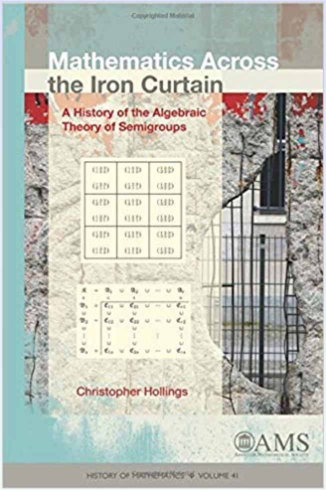
A small corner of a big picture!

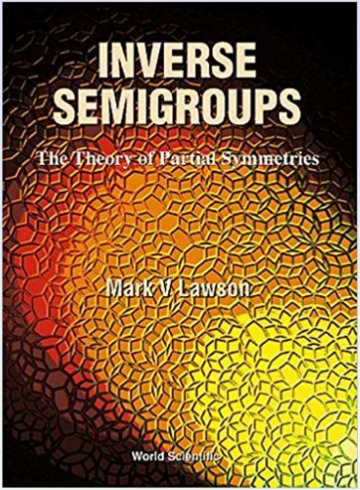
We will look at **inverse semigroups & monoids**:

- A branch of abstract algebra / semigroup theory.
- Introduced simultaneously & independently in 1950's
  - Viktor Wagner (U.S.S.R.)
  - Gordon Preston (U.K.)
- Theory developed separately, along two different tracks
  - USSR
  - U.S. & Europe

  with minimal contact between the two sides.
- Some degree of re-unification in 1990s

A couple of references:





Some -very- basic definitions

A **semigroup** is a set $(S, \_ \cdot \_)$ with an associative binary operation $\_ \cdot \_ : S \times S \to S$. usually written as concatenation:

- Given $a \in S$ and $b \in S$, then $ab \in S$.
- $a(bc) = (ab)c$ for all $a, b, c \in S$.

A **monoid** is a semigroup with an identity $1 \in S$ satisfying

$$1a = a = a1 \;\; \forall \; a \in S$$

A **group** is a monoid where every $a \in S$ has an inverse $a^{-1} \in S$

$$aa^{-1} \; = \; 1 \; = \; a^{-1}a \;\forall\; a \in S$$

Even more basic definitions

Simplest examples include **free semigroups** and **monoids**.

The free semigroup on a set X

X^+ is the set of all non-empty strings of symbols of X.

Composition is just concatenation of strings.

The **free monoid** X^* also allows for the empty string λ.

Free semigroups / monoids have the expected universal property ...

New monoids from old

A **congruence** $\sim$ on a semigroup S is a composition-preserving equivalence relation:

$$a \sim b \quad \text{and} \quad x \sim y \quad \Rightarrow \quad ax \sim by$$

for all $a, b, x, y \in S$.

Equivalence classes form the **quotient semigroup** $S/\sim$.

Every semigroup (monoid) is a quotient of some free semigroup (monoid).

There is no analogy of "normal subgroup" for monoids!

A simple definition

Inverse monoids / semigroups have a ‘relaxed’ notion of inverses:

Inverse semigroups: the definition

Every element $a \in S$ has a unique **generalised inverse** $a^\ddagger \in S$ satisfying

$$aa^\ddagger a = a \quad \text{and} \quad a^\ddagger a a^\ddagger = a^\ddagger$$

Axioms introduced independently by Wagner & Preston, based on (different collections of) concrete examples.

Many examples are – even nowadays – not always recognised as being inverse semigroups.

peter.hines@york.ac.uk

Elementary properties

In an inverse semigroup S, the following are almost immediate:

1. $(a^\ddagger)^\ddagger = a$, for all $a \in S$.
2. $(ab)^\ddagger = b^\ddagger a^\ddagger$, for all $a, b \in S$.
3. $e^\ddagger = e$, for any *idempotent* $e^2 = e$.

   (Special case: $I^\ddagger = I$, when S is a monoid).
4. $aa^\ddagger$ and $a^\ddagger a$ are both idempotent.
5. All idempotents commute:

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

A (well-known) class of examples!

All groups are (trivially) inverse monoids, but not vice versa.

Important

Even in a monoid, the conditions

$$aa^{\ddagger}a = a \quad \text{and} \quad a^{\ddagger}aa^{\ddagger} = a^{\ddagger}$$

do *not* imply that $aa^{\ddagger}$ is the identity.

Instead, $aa^{\ddagger}$ and $a^{\ddagger}a$ are both idempotent (i.e. $e^2 = e$).

The inverse semigroup axioms are strictly more general than the group axioms.

How should we understand these axioms?

A representation theorem or two

Cayley's theorem (1854)

Every group has a representation as bijections on a set.

The Wagner-Preston theorem (1954)

Every inverse semigroup has a representation
as partial injections on a set.

Inverse semigroup theory is what happens when we combine **reversibility** *with* **partiality.**

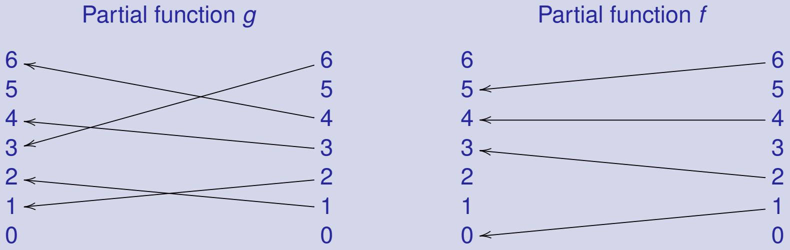
Historically, computer scientists have been more comfortable with partiality than mathematicians.

How do partial functions compose?

Given partial functions $f : X \to Y$ and $g : Y \to Z$, then $gf(x)$ is defined when

- x is in the domain of f.
- $f(x)$ is in the domain of g.

Composing partial reversible functions



peter.hines@york.ac.uk www.peterhines.info

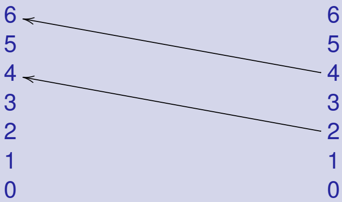
How do partial functions compose?

Given partial functions $f : X \to Y$ and $g : Y \to Z$, then $gf(x)$ is defined when

- x is in the domain of f.
- $f(x)$ is in the domain of g

Composing partial reversible functions

Partial function *gf*



A misleading intuition

It is easy to convince ourselves that,
unless domains / images match exactly[1],

1. The composites get progressively 'less defined'
2. Long composites must tend towards the nowhere-defined partial function 0.

Both of these intuitions are *incorrect*.

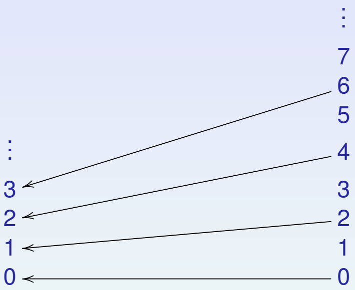
The reason why is best illustrated by example.

[1] In which case, everything reduces to group theory ...

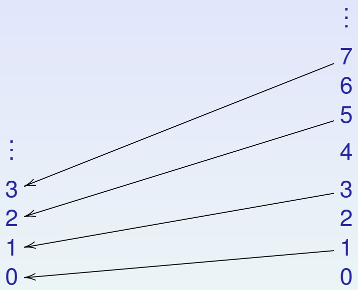
Some interesting partial injections

Partial injections defined on the **odd** and **even** numbers only:

Partial injection p



Partial injection q

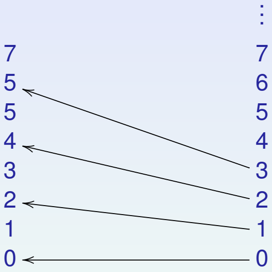


$$p(n) = \begin{cases} \frac{n}{2} & n \text{ even,} \\ \perp & \text{otherwise.} \end{cases} \quad , \quad q(n) = \begin{cases} \frac{n-1}{2} & n \text{ odd,} \\ \perp & \text{otherwise.} \end{cases}$$

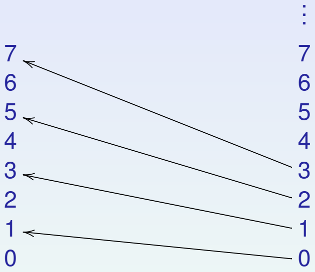
Total inverses of partial injections

Their generalised inverses are globally defined injections:

$p^{\ddagger}(n) = 2n$



$q^{\ddagger}(n) = 2n + 1$



Non-shrinking domains!

Observe:

$$dom(p) \subset dom(p^2) \subset dom(p^3) \subset dom(p^4) \subset dom(p^5) \subset \ldots$$

Nevertheless, these are all countably infinite.

These two functions generate a representation of the (inverse) **polycyclic monoid** P_2 of Nivat & Perot (1972). Specified by simple relations:

$$pp^{\ddagger} = I = qq^{\ddagger} \quad \text{and} \quad pq^{\ddagger} = 0 = qp^{\ddagger}$$

Also known as the logicians' 'dynamical algebra'

A significant inverse monoid:

Given any set X, the **polycyclic monoid** P_X, is the inverse monoid with:

- X as a generating set,
- A zero 0 and an identity I,
- the relations

$$xy^\ddagger = \begin{cases} I & x = y \\ 0 & x \neq y \end{cases}$$

A 'one-sided version of the Kronecker delta'.

A relevant property

Provided $|X| > 1$, there are no non-trivial congruences on P_X.

Any quotient causes a collapse to a single element.

The question of significance ...

What makes P_2 a "significant" monoid ??

A useful criterion

When it is repeatedly re-discovered in different fields:

Logic & λ calculus The dynamical algebra

C^* algebra & mathematical physics The Cuntz algebra

Automata theory Syntactic monoids of certain automata

Language theory The well-formed bracketing language

A range of areas Linguistics, **Ring Theory**, Tilings, Category Theory, Foundations of Mathematics, . . .

First seen as the ‘dynamical algebra’, of

“Local and Asynchronous β-reduction”
– V. Danos & L. Regnier (1992)

Models of untyped λ calculus, and hence computational universality.

Later core to *logical models* — particularly those of J.-Y. Girard.

Let’s look at more ‘elementary’ applications ...

Our starting point ...

Race Conditions

In parallel or multi-threaded computation:

> *"The behaviour of a system varies according to the order in which individual operations from distinct threads are processed'.'*

The distinct behaviours may be:

desirable, *undesirable*, or *unimportant*.

A non-judgmental analysis:

The terms 'desirable' or 'undesirable' are subjective.

We *study* such conditions, without aiming to either cause or eliminate them!

A practical application

"Hacking Starbucks for Unlimited Coffee"

`https://sakurity.com/blog/2015/05/21/starbucks.html`

Egor Homakov (`@homakov`)

Connecting to the same *Starbucks personal account* simultaneously from two *distinct browsers* caused a *race condition* among multiple *asynchronous processes* for:

1. Check balance on card 1.
2. If sufficient funds, add funds to card 2.
3. Decrease funds on card 1.

Disclaimer:This bug has since been fixed!

From hacking to algebra

Consider the free monoid over the set $\{A, C, D\}$

- A – add funds to card 2.
- C – check funds on card 1.
- D – decrease funds on card 1.

Which particular strings of actions (submonoids of the free monoid $\{A, C, D\}^*$) are:

1. Permitted by Starbucks servers?
2. Possible to create, using two distinct connections?
3. Profitable for Egor Homakov??
4. Fair to everyone concerned?

What we wish to find:

Tools to find *intersections* of such monoids,
and *transformations* (homomorphisms?) between them.

Interleaving processes as shuffles

The mathematics of **shuffling cards** and **interleaving processes** is of course identical.

Credit: Johnny Blood Photography

Card shuffles are *very* well-studied in *combinatorics*, *probability*, *representation theory*, *statistics*, &c.

For some applications to C.S., we also need their (inverse) semigroup theory.

Why not something more "traditional"??

Combinatorics answers questions such as:

> *Given K decks of N cards, how many different ways are there of shuffling them into a single stack of $K \times N$ cards?*

The right tools for multi-threaded finite computational tasks such as parallel matrix processing.

We need to consider the infinite setting

unbounded number of decks: Arbitrarily many clients connected to a server.

a never-ending stream of cards: Non-terminating processes (internet servers, again).

How to shuffle two (possibly infinite) decks of cards

Riffle Shuffles

- Cards from Deck A and Deck B are merged into a single stack.
- At each step, a single card is taken from the bottom of either A or B, and placed on top of the stack.

Some important conventions:

- The ordering of cards is preserved.
- Every card from each deck ends up in the stack.

Everything in order ...

Consider two copies of the natural numbers:

$$\mathbb{N} \uplus \mathbb{N} \overset{def.}{=} \mathbb{N} \times \{0, 1\}$$

and give this a *partial order* by

$$(a, i) \leqslant (b, j) \quad \text{iff} \quad a \leqslant b \text{ and } i = j$$

We may only compare members of the same copy of $\mathbb{N}$

- $(4, 0) \leqslant (7, 0)$
- $(3, 1) \leqslant (8, 1)$
- $(4, 0)$ and $(8, 1)$ are incomparable.

Hilbert's well-ordered hotel ...

Shuffles of two infinite decks of cards

$\equiv$

order-preserving injections

from $\mathbb{N} \uplus \mathbb{N}$ to $\mathbb{N}$.

How may we characterise (not count!) these?

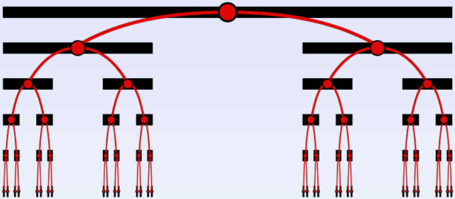
An old result (P.M.H. — M.V. Lawson 1998)

Arbitrary injections from $\mathbb{N} \uplus \mathbb{N}$ to $\mathbb{N}$ are in 1:1 correspondence with (effective) representations of P_2 on $\mathbb{N}$.

What about the monotone (order-preserving) case?

The infinitary setting

Every shuffle of two infinite decks corresponds to a point of Cantor space $\mathfrak{C}$.



Formally, one-sided infinite strings over $\{0, 1\}$,

$$c = 0100101101 \ldots$$

or equivalently, functions from $\mathbb{N}$ to $\{0, 1\}$.

The correspondence (computationally)

Operationally: Cantor points are *descriptions* of shuffles:

Given a Cantor point $c : \mathbb{N} \to \{0, 1\}$,

At the n^{th} step, a card was taken from:

- The first deck, when $c(n) = 0$
- The second deck, when $c(n) = 1$

Caution!

We can also think of Cantor points as *instructions*,

but not all Cantor points arise from valid shuffles.

An illustrative example

The **perfect riffle shuffle**:

Cards are alternately taken from each deck

This is modeled by the function $\phi : \mathbb{N} \times \{0, 1\} \to \mathbb{N}$ given by

$$\phi(n, i) = 2n + i$$

The corresponding Cantor point is $a(n) = n \ (mod\ 2)$.

$$a = 0101010101 \ldots$$

The **alternating Cantor point**

Which Cantor points are *not* shuffles?

Recall our two conditions:

1. The ordering of cards is preserved,
2. Every card is laid at some point.

Condition 1. is accounted for by monotonicity.

Condition 2. is automatically satisfied, simply because $\phi : \mathbb{N} \times \{0, 1\} \to \mathbb{N}$ is a *globally defined* function.

A consequence is that that the corresponding Cantor point is **balanced**:

$$\sum_{i=0}^{\infty} c(i) \; = \; \infty \; = \; \sum_{i=0}^{\infty} (1 - c(i))$$

The correspondence (mathematically)

Given a shuffle of two infinite decks $\phi : \mathbb{N} \times \{0,1\} \to \mathbb{N}$

consider its (global) inverse $\phi^{-1} : \mathbb{N} \to \mathbb{N} \times \{0,1\}$.

For all $n \in \mathbb{N}$, we have a pair $\phi^{-1}(n) \ = \ (x_n, i_n) \ \in \mathbb{N} \times \{0,1\}$.

From shuffles to Cantor points

we define a Cantor point

$$c_\phi \ = \ \pi_2 \phi^{-1} : \mathbb{N} \to \{0,1\} \in \mathfrak{C}$$

by *projecting onto the second component* $c_\phi(n) = i_n \ \in \ \mathfrak{C}$

As ϕ is *monotone*, this Cantor point is enough to characterise ϕ.

What about that other projection?

Does projecting onto the *first* component also characterise ϕ?

Definitely not There are uncountably many distinct shuffles where such projections are identical.

However, taking two *partial* projections will work!

Shuffles via inverse semigroup theory

Given the inverse of a shuffle $\phi^{-1} : \mathbb{N} \to \mathbb{N} \times \{0, 1\}$,

let us split the projection onto the first component into two distinct monotone partial injections

$$p_\phi(n) = \begin{cases} \pi_1\phi^{-1}(n) & \pi_2\phi^{-1}(n) = 0 \\ \text{undefined} & \text{otherwise.} \end{cases}$$

$$q_\phi(n) = \begin{cases} \pi_1\phi^{-1}(n) & \pi_2\phi^{-1}(n) = 1 \\ \text{undefined} & \text{otherwise.} \end{cases}$$

These two partial injections are enough to characterise ϕ.

Perhaps unsurprisingly ...

The generalised inverses of p_ϕ and q_ϕ are monotone injections, and satisfy

$$p_\phi p_\phi^\ddagger = I = q_\phi q_\phi^\ddagger$$

$$q_\phi p_\phi^\ddagger = 0 = p_\phi q_\phi^\ddagger$$

Giving an effective representation of a two-generator polycyclic monoid.

From the Cantor point to the inverse monoid

Given a balanced Cantor point $c : \mathbb{N} \to \{0, 1\}$, we define partial injections by:

Counting the number of 0s up to point n

$$p_c^{\ddagger}(n) = \begin{cases} \left(\sum_{j=0}^{n} 1 - c(j)\right) - 1 & c(n) = 0, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Counting the number of 1s up to point n

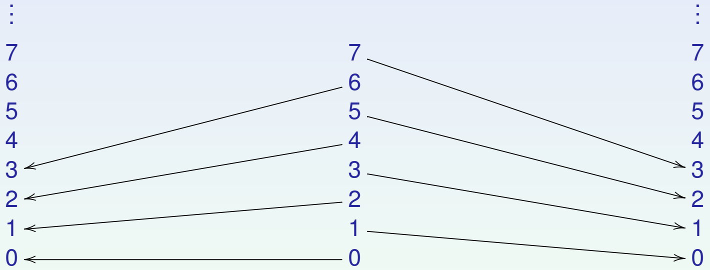
$$q_c^{\ddagger}(n) = \begin{cases} \left(\sum_{j=0}^{n} c(j)\right) - 1 & c(n) = 1, \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Best illustrated by example ...

Consider the alternating Cantor point

$$a = 0101010101010101010\ldots$$

partial injection $p_a^\ddagger$ **partial injection** $q_a^\ddagger$



A correspondence

We have 1:1 mappings between:

Interleavings of two infinite streams of processes

Balanced points of Cantor space

Monotone effective representations of 2-generator polycyclic monoids

Is there any advantage to treating such things algebraically?

Re-ordering processes

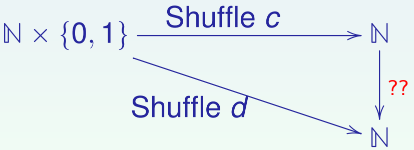
Given a sequence of 'cards'

$$A_0\ A_1\ A_2\ A_3\ A_4\ A_5 \ldots$$

resulting from some (undesirable) shuffle, specified by
$c : \mathbb{N} \to \{0, 1\}$,
how can we *re-order it*

$$A_5\ A_2\ A_3\ A_0\ A_1\ A_4 \ldots$$

so it appears to have come from a (desirable) shuffle
$d : \mathbb{N} \to \{0, 1\}$



peter.hines@york.ac.uk www.peterhines.info

How to re-order shuffles

Some well-known semigroup theory

Given partial injections f, g with:

- disjoint domains
- disjoint images

their set-theoretic union $f \cup g$ is also a partial injection.

Given shuffles $c, d : \mathbb{N} \to \{0, 1\}$, the result of c may be re-arranged into the result of d by

$$p_d^{\ddagger} p_c \;\cup\; p_d^{\ddagger} q_c \;:\; \mathbb{N} \to \mathbb{N}$$

— a globally defined bijection on $\mathbb{N}$.

Some computational caution ...

From a practical viewpoint:

Tasks cannot be re-ordered if they are processed the instant they are received!

For re-arrangement to take place, they must first be held in a buffer / queue.

- How big does this need to be — how long a queue is (computationally) acceptable?
- What transformations on this buffer are needed?
- Are there situations where no *finite* re-arrangement will work?

From balanced Cantor points to Young Tableaux

A traditional approach to (finite) shuffles is via **Young Tableaux**.

We derive infinitary versions from the algebra in a simple manner:

Consider the Cantor point $c = 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ \ldots \in \mathfrak{C}$ and associated partial injections $p^\ddagger, q^\ddagger : \mathbb{N} \to \mathbb{N}$

$n =$	0	1	2	3	4	$\ldots$
$p^\ddagger(n) =$	**1**	**3**	**5**	**8**	**9**	$\ldots$
$q^\ddagger(n) =$	**0**	**2**	**4**	**6**	**7**	$\ldots$

An (∞, ∞) Young tableau.

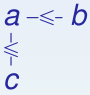
The obvious question:

What about **standard** Young tableaux?

On to standard Young tableaux

In **standard** Young tableaux, the cells are well-ordered both *horizontally* and *vertically*.

a	b
c	



Horizonal ordering corresponds to monotonicity.

What about the vertical ordering?

peter.hines@york.ac.uk
www.peterhines.info

Some computer science motivation

Recall the motivation for studying Shuffles, as *ordering of processes*.

- Operations from thread A push data onto a stack.
- Operations from thread B pop data off a stack.

What conditions would prevents us from trying to *read data from an empty stack*?

... or indeed, transfer funds from an empty account?

A (binary) **ballot sequence** is an element $w \in \{0,1\}^*$ where, for *every* prefix u of w,

$$\#1s \text{ in } u \quad \leqslant \quad \#0s \text{ in } u$$

Denote the set of all finite ballot sequences by $B_{\{0,1\}}$ — this forms a submonoid of $\{0,1\}^*$.

By contradiction: Consider $v, w \in B_{\{0,1\}}$ such that $vw \notin B_{\{0,1\}}$. Then there exists some prefix u of vw satisfying $\#0s$ in $u < \#1s$ in u. As $v \in B_{\{0,1\}}$, u is not a prefix of v, so $u = vl$, for some prefix l of w. However, $\#0s$ in $v \geqslant \#1s$ in v. Therefore, $\#1s$ in $l \geqslant \#0s$ in l, contradicting the assumption that $w \in B_{\{0,1\}}$.

A deceptively simple monoid

Ballot sequences are *well-studied* in combinatorics – but also make for interesting monoids!

Proposition The monoid of binary ballot sequences is not finitely generated.

By contradiction: Assume a finite generating set G for $B_{\{0,1\}} \leqslant \{0,1\}^*$. As G is finite, the longest contiguous string of $1s$ in any member of G is bounded by some finite $K \in \mathbb{N}$. No composite of members of G can account for the ballot sequence $0^{K+1}1^{K+1}$.

From the finite to the infinite:

A Cantor point $c \in \mathfrak{C}$ is **ballot** when every prefix is a member of the Ballot monoid.

$$\sum_{j=0}^{N} c(j) \leqslant \sum_{j=0}^{N} c^{\perp}(j) \ \forall N \in \mathbb{N}$$

Denote the ballot Cantor points by $\mathfrak{B} \subseteq \mathfrak{C}$.

Our claim:

Shuffles described by *ballot* Cantor points
are precisely those whose Young tableaux are standard

A quick outline:

Balanced ballot Cantor points

$$\equiv$$

standard (∞, ∞) Young tableaux

Let $c \in \mathfrak{B}$ be a balanced ballot Cantor point. This determines a monotone representation of P_2 as partial injections on $\mathbb{N}$, and hence an (∞, ∞) Young tableau:

$p^{\ddagger}(0)$	$p^{\ddagger}(1)$	$p^{\ddagger}(2)$	$p^{\ddagger}(3)$	$p^{\ddagger}(4)$	$\dots$
$q^{\ddagger}(0)$	$q^{\ddagger}(1)$	$q^{\ddagger}(2)$	$q^{\ddagger}(3)$	$q^{\ddagger}(4)$	$\dots$

By the interpretation of $p(n)$ and $q(n)$ as 'counting the 0s and 1s in a prefix', $p^{\ddagger}(n) \leqslant q^{\ddagger}(n)$, so this is *standard*.

An almost paradoxical point(!)

A **balanced** ballot point $b \in \mathfrak{B}$ satisfies:

- $\sum_{j=0}^{\infty} b(j) = \sum_{j=0}^{\infty}(1 - b(j))$
  The total number of 0s and 1s is the same.
- $\sum_{j=0}^{N} b(j) \leqslant \sum_{j=0}^{N}(1 - b(j))$.
  Every prefix has at least as many 0s as 1s.

Imposing such conditions on *finite* strings results in an uninteresting theory!

Finite balanced Ballot points are simply powers of (01).

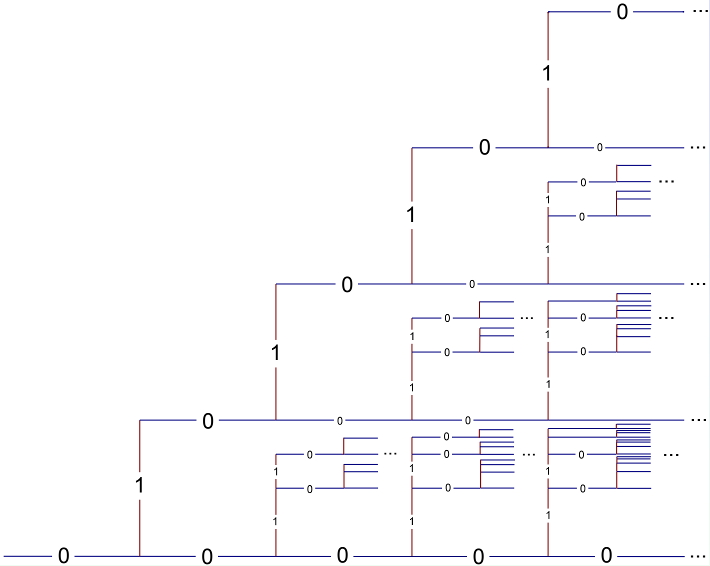
The free monoid on a single generator!

The (balanced) Ballot Cantor points form

a subset of Cantor space;

We can draw a picture.

Ballot Cantor points – the “fork factory”



From semigroup theory to order theory

There are many different ways of ordering:

- Cantor points in general,
- Ballot points in particular.

The pointwise partial order:

Given Cantor points $a, b : \mathbb{N} \rightarrow \{0, 1\}$,

we use the *pointwise* partial ordering:

$$a \leqslant b \text{ iff } a(n) \leqslant b(n) \ \forall n \in \mathbb{N}$$

The Ballot points of Cantor space have a particularly neat form.

The Ballot Scott domain

Key properties:

- There is no top element & they are **not** closed under joins $(c \vee d)(n) \ = \ max\{c(n), d(n)\}$.
- They **are** closed under the meet, $(c \wedge d)(n) \ = \ c(n)d(n)$
- There is a bottom element $\perp(n) = 0$, for all $n \in \mathbb{N}$.
- The supremum of every chain $c_0 \leqslant c_1 \leqslant c_2 \leqslant \ldots$ is also in $\mathfrak{B}$

  – chain-completeness $\Rightarrow$ directed completeness, assuming the axiom of choice (Iwamura's Lemma).
- There is a notion of **finite support** / **compactness**: $c \in \mathfrak{B}$ is "**finitary**" iff $\sum_{j=0}^{\infty} c(j) < \infty$, and every element is the supremum of a chain of such elements.

peter.hines@york.ac.uk www.peterhines.info

Scott Domains ...

- Introduced by Dana Scott (early 1970s) to model pure untyped λ calculus

  — and hence **computational universality**.

- Also used for semantics of **functional programming** languages, due to the existence of solutions of arbitrary **fixed-point equations.**

This particular Scott domain is

a subset of Cantor space

related to standard Young tableaux.

Back to our original motivation ...

Given $C, D \subseteq \mathbb{N}$, the **banker's monoid** $\mathcal{W}_{C,D}$

is a submonoid of the free monoid over $+C \cup -D$.

Interpretation

For any $c \in C$, and $d \in D$,

$+c$ is a *deposit* , $-d$ is a *withdrawal*.

Elements of $\mathcal{W}_{C,D}$ are **no-credit strings** — those for which *the sum of every prefix is non-negative*.

Taking $C = \{2, 4, 6, 8\}$ and $D = \{1, 3, 5, 7\}$,

$(+8)(-5)(+4)(-7)(+4)(-3)$ is a n.-c. string

$(+6)(-5)(+2)(-5)(+8)$ is not a n.-c. string

Reducing the complex to the simple

It is relatively straightforward to prove:

1. The Ballot monoid $\mathcal{B}_{\{0,1\}}$ is isomorphic to $\mathcal{W}_{\{1\},\{1\}}$.
2. For arbitrary $C, D \subseteq \mathbb{N}$, there is an embedding $\mathcal{W}_{C,D} \hookrightarrow \mathcal{B}_{\{0,1\}}$.

We may use the same structures to study

1. Race conditions for stacks
2. Similar for credits / debits of Starbucks cards ...

Hierarchical Shuffles

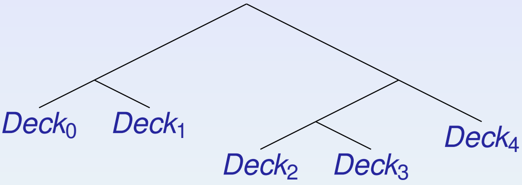
We shuffle two (infinite) decks of cards.

Either or both of these are the result of previous

shuffles of infinite decks of cards.

Characterising iterated shuffles

We take the obvious step of drawing this as a binary tree:



(As a *slight simplification* we assume the same shuffle at each step).

The questions ...

What we would like to do:

1. Write down the appropriate bijection:

$$\mathbb{N} \times \{0, 1, \ldots, k\} \longrightarrow \mathbb{N}$$

2. Give the corresponding Young tableaux.
3. Ensure (when appropriate) these are *standard* tableaux.
4. Transform the result of one tree of shuffles into another.

Very basic T.C.S. / algebra

A **binary code** is a subset $A \subseteq \{0,1\}^*$ such that the submonoid generated by A is *freely generated*.

A simple example:

The set

$$L = \{00, 01, 10, 110, 111\} \subseteq \{0,1\}^*$$

is a binary code.

Operationally: strings of elements of L can be split up, *uniquely*, into elements of L.

$$1111000011100 1111$$

splits, uniquely, as

$$(111)(10)(00)(01)(110)(01)(111)$$

A useful class of examples

A **maximal prefix code** is a subset $A \subseteq \{0,1\}$ where:

1. Members of A are not prefixes of each other.
2. Every word of $\{0,1\}^*$ either:
   - is a prefix of some element of A,
   - has some element of A as a prefix.

Some elementary T.C.S.

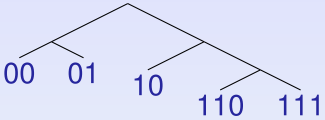
There is a simple and well-known correspondence:

Complete binary trees

$\equiv$

Maximal prefix codes

Maximal prefix codes as binary trees



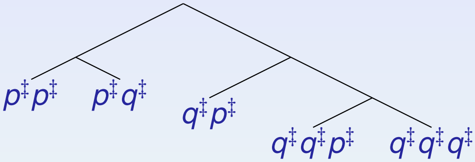
leaf-traversal ≡ lex-ordering

Assuming $0 < 1$, the leaf-traversal, and lexicographic ordering coincide

$$\{00 \leqslant 01 \leqslant 10 \leqslant 110 \leqslant 111\}$$

From codes to polycyclic monoids

Let's label our tree branchings by $\{p^\ddagger, q^\ddagger\}$ instead.



(Maximal) Prefix codes correspond to *embeddings*:
Given a (maximal) prefix code $\mathcal{L}$ over $\{p^\ddagger, q^\ddagger\}^* \hookrightarrow P_2$, then for all $u^\ddagger, v^\ddagger \in L$,

$$uv^\ddagger = \begin{cases} I & u = v \\ 0 & u \neq v \end{cases}$$

(A one-sided version of the Kronecker delta ...)

Giving us an embedding $P_L \hookrightarrow P_2$.

From prefixes to shuffles

Assume a representation of P_2 based on the alternating Cantor point $a = 01010101\ldots$
(Equivalently, the perfect riffle shuffle ...)
We have

$$p^{\ddagger}(n) = 2n \quad \text{and} \quad q^{\ddagger}(n) = 2n + 1$$

The max. prefix code

$$\mathcal{L} = \{p^{\ddagger}p^{\ddagger}\ ,\ p^{\ddagger}q^{\ddagger}\ ,\ q^{\ddagger}p^{\ddagger}\ ,\ q^{\ddagger}q^{\ddagger}p^{\ddagger}\ ,\ q^{\ddagger}q^{\ddagger}q^{\ddagger}\}$$

gives us a $(\infty, \infty, \infty, \infty, \infty)$ Young tableau:

The tableau in question:

$n =$	0	1	2	3	4	...
$p^\ddagger p^\ddagger(n) =$	**0**	**4**	**8**	**12**	**16**	...
$p^\ddagger q^\ddagger(n) =$	**2**	**6**	**10**	**14**	**18**	...
$q^\ddagger p^\ddagger(n) =$	**1**	**5**	**9**	**13**	**17**	...
$q^\ddagger q^\ddagger p^\ddagger(n) =$	**3**	**11**	**19**	**27**	**31**	...
$q^\ddagger q^\ddagger q^\ddagger(n) =$	**7**	**15**	**23**	**31**	**39**	...

This is not a *standard* Young tableau

What we have:

1. Every natural number
2. Ordered rows
3. Unordered columns

The tableau in question:

$n =$	0	1	2	3	4	...
$p^\ddagger p^\ddagger(n) =$	**0**	**4**	**8**	**12**	**16**	...
$p^\ddagger q^\ddagger(n) =$	**2**	**6**	**10**	**14**	**18**	...
$q^\ddagger p^\ddagger(n) =$	**1**	**5**	**9**	**13**	**17**	...
$q^\ddagger q^\ddagger p^\ddagger(n) =$	**3**	**11**	**19**	**27**	**31**	...
$q^\ddagger q^\ddagger q^\ddagger(n) =$	**7**	**15**	**23**	**31**	**39**	...

This is not a *standard* tableau

What we have:

1. Every natural number
2. Ordered rows
3. Unordered columns

For any balanced Ballot Cantor point, $b \in \mathfrak{B}$,

the corresponding shuffle gives a *standard* (∞, ∞) Young tableau.

What were we thinking??

There is no reason to expect that

an arbitrary hierarchical iteration of such shuffles

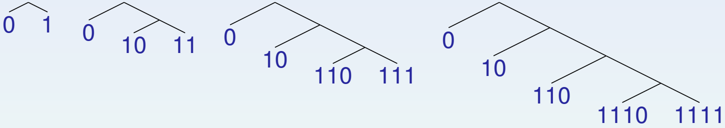
should give a *standard* tableau.

The right way to associate

The claim:

A neccessary & sufficient condition for a hierarchical shuffle to give a *standard* tableaux is that:

the corresponding tree is *right-associated*.

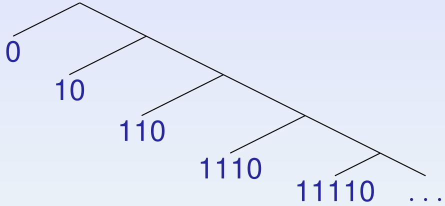


$\{0, 1\}$ $\{0, 10, 11\}$ $\{0, 10, 110, 111\}$ $\{0, 10, 110, 1110, 1111\}$

peter.hines@york.ac.uk www.peterhines.info

From the finite to the infinite

Maximal prefix codes $R \subseteq \{0,1\}$ need not be finite:



$$\{0 \leqslant 10 \leqslant 110 \leqslant 1110 \leqslant 11110 \leqslant \ldots\}$$

The unique right-associated, well-ordered, infinite prefix code.

A worked example:

Let's do this for the representation of P_2 corresponding to the shuffle determined by the *alternating Cantor point*

$$a = 0010101010101\ldots \in \mathfrak{B}$$

Following the same procedure:

Mapping prefix codes to polycyclic monoids

$$0 \mapsto p^{\ddagger} \quad \text{and} \quad 1 \mapsto q^{\ddagger}$$

where

$$p^{\ddagger}(n) = 2n \quad \text{and} \quad q^{\ddagger}(n) = 2n + 1$$

The infinite alternating shuffle

We get our $(\infty, \infty, \infty, \ldots)$ standard Young tableau.

$n =$	0	1	2	3	4	5	$\ldots$
$p^\ddagger(n) =$	0	2	4	6	8	10	$\ldots$
$q^\ddagger p^\ddagger(n) =$	1	5	9	13	17	21	$\ldots$
$(q^2)^\ddagger p^\ddagger(n) =$	3	11	19	27	35	43	$\ldots$
$(q^3)^\ddagger p^\ddagger(n) =$	7	23	39	55	71	87	$\ldots$
$(q^4)^\ddagger p^\ddagger(n) =$	15	47	79	111	143	175	$\ldots$
$(q^5)^\ddagger p^\ddagger(n) =$	31	94	159	223	287	351	$\ldots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

A (Hilbert-hotel style) bijection from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$, that is monotone in both variables:

$$(r, c) \mapsto 2^c(2r+1) - 1$$

peter.hines@york.ac.uk www.peterhines.info

Not forgetting our motivation ...

This is derived from:

Alternating shuffles of decks of cards.

We start by shuffling two Decks A and B.

Deck B arose from shuffling Decks B' and C.

Deck C arose from shuffling Decks C' and D.

Deck D arose from shuffling Decks D' and E.

$\vdots$

Practically – is it easy / possible to perform this shuffle?

Deep fractal structure ??

Play a card from the following decks, in order :

0	1	0	2	0	1	0	3	0	
1	0	2	0	1	0	4	0	1	
0	2	0	1	0	3	0	1	0	
2	0	1	0	5	0	1	0	2	
0	1	0	3	0	1	0	2	0	. . .

Question : How may we characterise this sequence?

Deep fractal structure ??

Which deck do we play from, at each step?

	$Deck_0$	$Deck_1$	$Deck_2$	$Deck_3$	$Deck_4$
Step 1	•				
Step 2		•			
Step 3	•				
Step 4			•		
Step 5	•				
Step 6		•			
Step 7	•				
Step 8				•	
Step 9	•				
Step 10		•			
Step 11	•				
Step 12			•		
Step 13	•				
Step 14		•			
Step 15	•				
Step 16					•

This looks kind of familiar!

	2^4	2^3	2^2	2^1	2^0
Step 1					1
Step 2				1	0
Step 3				1	1
Step 4			1	0	0
Step 5			1	0	1
Step 6			1	1	0
Step 7			1	1	1
Step 8		1	0	0	0
Step 9		1	0	0	1
Step 10		1	0	1	0
Step 11		1	0	1	1
Step 12		1	1	0	0
Step 13		1	1	0	1
Step 14		1	1	1	0
Step 15		1	1	1	1
Step 16	1	0	0	0	0

Performing the perfect infinite riffle

A very simple rule

1. Count in binary ...
2. Which bit has changed from 0 to 1?
3. Play a card from that deck!

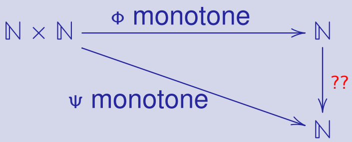
We can do the same with any $b \in \mathfrak{B}$

Each Balanced Ballot point determines a distinct:

- $(\infty, \infty, \infty, \ldots)$ standard Young tableau.
- shuffle of infinitely many decks of cards, satisfying:
  "Number of cards played from $Deck_i$ is always $\geqslant$ Number of cards played from $Deck_{i+1}$"
- bijection $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ monotone in both variables.

Using inverse semigroup theory

It is straightforward to describe the mappings between these:



Exercise : What group do we get??

From the infinite to the finite

Every balanced Ballot Cantor point determines an $(\infty, \infty, \infty, \ldots)$ standard Young tableau. These are equivalent to:

Infinite inclusion-ordered chains of finite standard Young tableaux.

For the alternating Cantor point:

n=0

0

n=1

0
1

n=2

0	2
1	

n=3

0	2
1	
3	

n=4

0	2	4
1		
3		

n=5

0	2	4
1	5	
3		

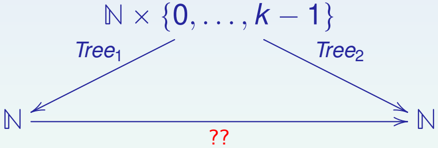
. . . just a complicated way of counting in binary!

Transforming (finite) hierarchical shuffles

Let us fix some balanced Cantor point $c \in \mathfrak{C}$

(We do not assume it is a Ballot point!)

Given trees $Tree_1$, $Tree_2$, both with k leaves,
how can we transform the result of one hierarchical shuffle into the other?



From prefix codes to groups (I)

$Tree_1$ and $Tree_2$ both determine k-element maximal prefix codes over $\{p^\ddagger, q^\ddagger\}$. Call these

$$R = \{r_0^\ddagger, \ldots, r_{k-1}^\ddagger\} \text{ and } S = \{s_0^\ddagger, \ldots, s_{k-1}^\ddagger\}$$

The required bijection is simply:

$$s_0^\ddagger r_0 \ \cup \ldots \ s_j^\ddagger r_j \ldots \cup \ \ldots \ \cup s_{k-1}^\ddagger r_{k-1}$$

The intuition

Element r_j maps the j^{th} row of a Young tableau

to the whole of $\mathbb{N}$

Element $s_j^\ddagger$ maps the whole of $\mathbb{N}$

to the j^{th} row of another Young tableau.

From prefix codes to groups (II)

For an *arbitrary* effective representation of P_2,

The set of *all* such bijections (including varying $k \in \mathbb{N}$) is closed under compositions and inverses.

Question: Which group is this?

From prefix codes to groups (III)

This has already been shown, as a study in abstract algebra / semigroup-theory, to be **Thompson's group** $\mathcal{F}$ in

"The Polycyclic Monoids & The Thompson Groups"
M. Lawson, **Comm. In Alg.** *(35)   (2007)*

A corollary

Each *balanced Cantor point* uniquely determines a representation of $\mathcal{F}$ as bijections on $\mathbb{N}$.

Does 'anything special' happen when we choose a Ballot point?

About that group ...

A particularly ‘significant’ group:

Thompson’s group $\mathcal{F}$

- one of the best-known groups in mathematics
- defined in 1965, as a potential counter-example to a conjecture of von Neumann
- a rich source of conjectures & counterexamples
- has linear-time word problem
- closely connected to both complexity and category theory
- proposed (2004) as a platform for non-commutative cryptography

Time for a definition!

The group $\mathcal{F}$ was originally defined via *representations*.

Abstractly, it may be defined as the group with:

- A countably infinite set of generators $\{x_0, x_1, x_2, \ldots\}$
- Relations given by

$$x_k^{-1} x_n x_k \;=\; x_{n+1} \quad \text{for all} \;\; k < n$$

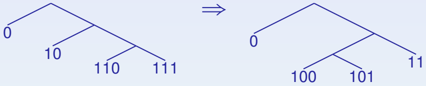
(Other presentations are possible, but
this is the most intuitive / natural)

Some explicit calculations ...

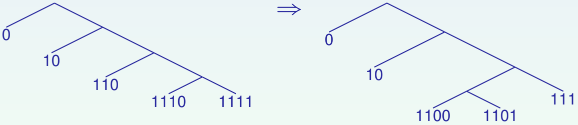
To which tree re-arrangements do these correspond?

X_0 performs:

X_1 performs:



X_2 performs:



X_3 performs: ...

Let us consider shuffles defined by a balanced **ballot** point.

For convenience, we again consider the alternating point:

$$a = 010101010101\ldots \in \mathfrak{B}$$

Important:
this is for illustration;
other balanced ballot points will do!

What is immediately noticeable?

Some obvious points ...

Consider (the representation of) each generator as 'mapping the results of one (hierarchical) shuffle into another'.

Generator x_j re-arranges the result of S_j into that of T_j'.

- S_{j+1} is obtained by using the result of S_j as the 2^{nd} deck in an alternating shuffle (and similarly for $T_j + 1$).
- Each S_j is the **unique** hierarchical shuffle that gives a **standard** $(\infty, \infty, \ldots, \infty)$ Young tableau.
- Each S_j is re-arranged into T_j by a single *rotation* or *associator*

on its final three leaves.

Generators converge to the identity

Generator x_j is the *identity* on the first $j-1$ rows of our $(\infty, \infty, \ldots)$ standard Young tableau:

$n =$	0	1	2	3	4	5	...
$p^\ddagger(n) =$	0	2	4	6	8	10	...
$q^\ddagger p^\ddagger(n) =$	1	5	9	13	17	21	...
$(q^2)^\ddagger p^\ddagger(n) =$	3	11	19	27	35	43	...
$(q^3)^\ddagger p^\ddagger(n) =$	7	23	39	55	71	87	...
$(q^4)^\ddagger p^\ddagger(n) =$	15	47	79	111	143	175	...
$(q^5)^\ddagger p^\ddagger(n) =$	31	94	159	223	287	351	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

$$\lim_{n\rightarrow\infty}(x_n) \ = \ I_{\mathbb{N}}$$

Formally, this is a point-wise limit:

$$\forall a \in \mathbb{N}\ ,\ \exists T \in \mathbb{N} \ \text{ such that } \ n \geqslant T \ \Rightarrow \ x_n(a) = a$$

The NP-intermediate class

The complexity class **NPI** is the class of problems that are:

- in **NP**,
- not in **P**,
- not **NP**-complete.

Ladner's Theorem (1975)

NPI is non-empty $\iff$ **P** $\neq$ **NP**

Some (possibly) **NPI** problems

We cannot say, for certain, that *any* problem is in **NPI**.

Some 'promising candidates'

- Prime factorisation.
- Deciding graph isomorphism.
- Finding the 'rotation distance' between two trees.
- Computing discrete logarithms, and related problems.

Ladner produced 'highly artificial' problems that are guaranteed to be in **NPI**, provided **P** $\neq$ **NP**.

No 'natural' problems with the same property are known.

The tree rotation distance problem

A **rotation** of binary trees is a *local tree transformation* of the form:

where **A**, **B** and **C** may be leaves or subtrees.

Sleator, Tarjan & Thurston (1988)

Any n-node tree can be transformed into any other n-node tree using a maximum of $2n - 6$ rotations.

Čulík and Wood's problem (1982)
Given two trees, what is the *shortest* sequence of rotations that will transform one into the other?

The notion of rotation

Let us fix some (possibly Ballot) balanced Cantor point

- A rotation, applied to a tree S, gives another tree T.
- Together, this pair of trees determines an element of $\mathcal{F}$.
- The generators $\{X_0, X_1, \ldots\}$ are of this form.

Can we re-write Čulík and Wood's problem
as a question about words in $\mathcal{F}$?

Significant previous work ...

"On the rotation distance between binary trees"
– P. Dehornoy (2009)

" ...introducing a partial action of $\mathcal{F}$ on trees and expressing the rotation distance between two trees as the length of an element of $\mathcal{F}$.

This approach easily leads to a lower bound. However, **due to the lack of control on the geometry of** $\mathcal{F}$, it seems difficult to obtain higher lower bounds .... "

Partiality / lack of control??

Given a rotation:

we get the *same* element of $\mathcal{F}$, for *arbitrary* **A**, **B** and **C**.

Thinking semigroup-theoretically

Generators of $\mathcal{F}$ decompose into ‘more primitive’ operations :

- Mapping rows between Young tableaux.
- Splitting a single row into two.
- Merging two rows into one.

We have a more ‘fine-grained’ control,
using inverse semigroups instead.

peter.hines@york.ac.uk www.peterhines.info

What we are missing ...

We can take this further, but at some point, we are forced to interpret as **Category theory:**

Coherence for associativity An entire field based on the study of associators (rotations).

Higher categorical coherence Operads & related structures.

Symmetries of Polyhedra Associahedra, permutahedra, &c. via group and inverse semigroup theory.

More at N.Y.C. category theory seminar, 10th of Feb., 2021