Turing Categories

Dedicated to Pieter Hofstra

Robin Cockett
University of Calgary

This talk is based on :

A. Introduction to Turing categories (with Pieter)

B. Timed sets, functional complexity, & computability
(with Boils, Gallagher, Hrubes)

C. Total maps of Turing categories
(with Pieter and Pavel Hrubes)

D. Estonia notes (on my web.page)

Time line

2004	Met Pieter in Ottawa (shared office!)
2005	Pieter joined me as postdoc
2007	Pieter returned to Ottawa as Faculty

basic theory of Turing cats →

Continued to work together

⋮

Ingredients:

Turing categories $\equiv$ abstract computability

- notion of computable map
- partially defined functions
- relation to partial combinatory algebras (PCA)
- reconstructing recursion theory
- Turing objects and maps

Plan of talk:

- Restriction categories — abstract theory of partial maps
- Timed sets and complexity — a construction from CS
- Turing categories and examples — Δ-time maps + more
- The road ahead...

Restriction categories:

but had a history

Developed with Steve Lack

Definition:

A restriction category is a category $\mathbb{X}$ together with a restriction combinator:

$$\frac{A \xrightarrow{f} B}{A \xrightarrow[\bar{f}]{} A}$$

Satisfying:

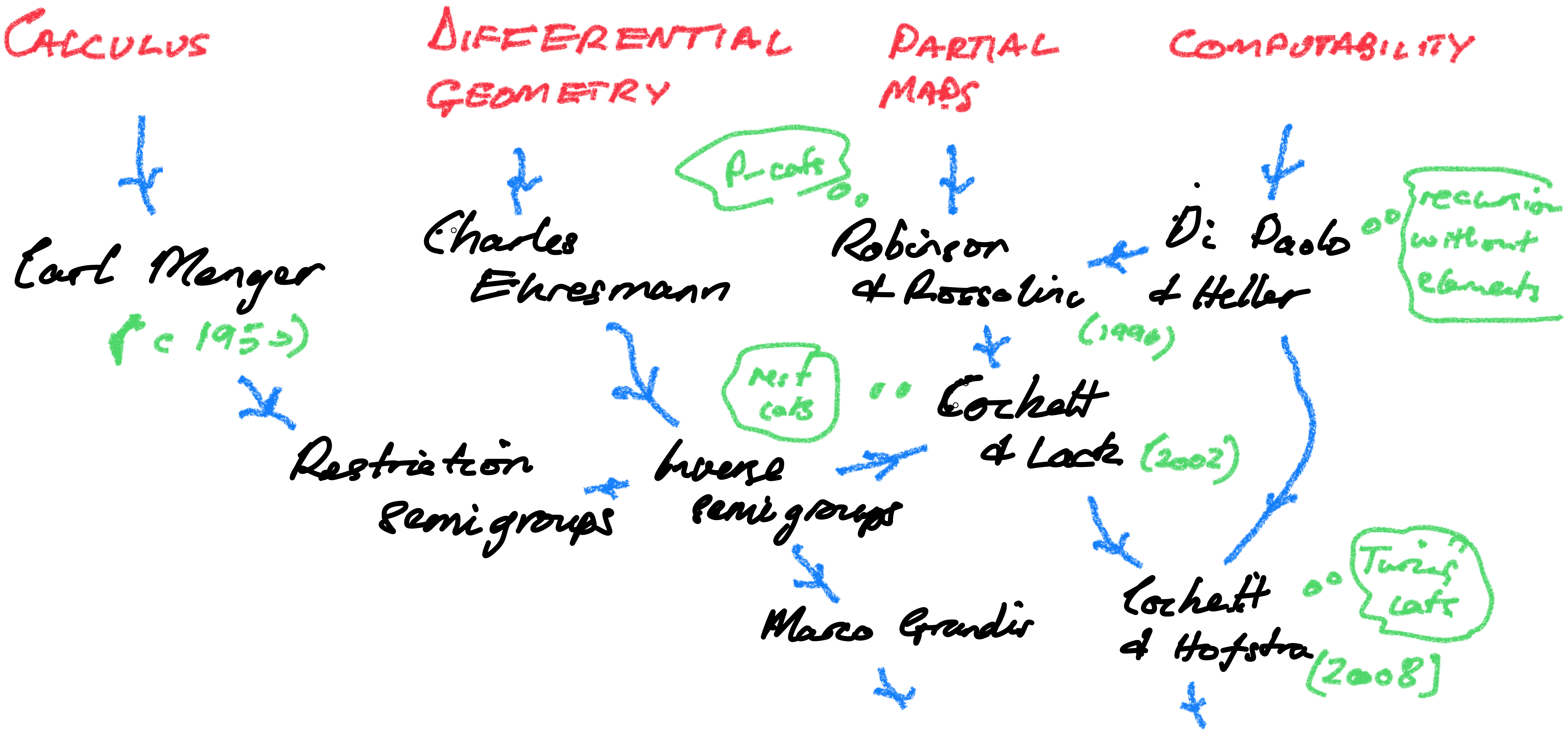
[R.1] $\bar{f} f = f$

[R.2] $\bar{f}\,\bar{g} = \bar{g}\,\bar{f}$

[R.3] $\bar{f}\,\bar{g} = \overline{\bar{f} g}$

[R.4] $f\,\bar{h} = \overline{fh}\, f$

Addressing Partiality ...



CALCULUS

DIFFERENTIAL GEOMETRY

PARTIAL MAPS

COMPUTABILITY

Karl Menger

(c 1953)

Charles Ehresmann

P-cats

Robinson & Rossolini

Di Paola & Heller

recursion without elements

(1986)

Restriction semigroups

Inverse semigroups

Rest cats

Cockett & Lack (2002)

Marco Grandis

Cockett & Hofstra

Turing cats

(2008)

Some important manipulations:

In any restriction category $\mathbb{X}$:

- f monic $\Rightarrow \bar{f} = 1$
- $\overline{fg} = \overline{f\bar{g}}$
- $\bar{\bar{f}} = \bar{f}$
- $\bar{f}\bar{f} = \bar{f}$
- $f \leq g \iff \bar{f}g = f$

$$\overline{f\bar{g}} = \overline{\overline{fg}f} = \overline{fg}\,\bar{f} = \bar{f}\,\overline{fg} = \overline{\bar{f}fg} = \overline{fg}$$

restriction idempotent

restriction partial order

Sets and partial maps, Par, is a restriction category

$$\bar{f}(x) = \begin{cases} x & \downarrow f(x) \\ \uparrow & \uparrow \end{cases}$$

Defn f in a restriction category is *total* iff $\bar{f} = 1_A$.

Total maps form a subcategory, $\mathrm{Total}(\mathbb{X}) \subseteq \mathbb{X}$

$$\bar{f} = 1_A \quad \bar{g} = 1_B \Rightarrow \overline{fg} = \overline{f\bar{g}} = \overline{f1_B} = \bar{f} = 1_A$$

Thm. Fundamental thm of restriction cats!

Every restriction category embeds fully and faithfully in a partial map category.

theory of partiality captured by restriction cats.

Proof (sketch)

the idempotent completion

Form $\mathrm{Split}_r(\mathcal{X})$ then consider $\mathrm{Total}(\mathrm{Split}_r(\mathcal{X}))$ splittings of idempotents form a stable system of monics $\mathcal{M}$. $\mathrm{Split}_r(\mathcal{X}) \simeq \mathrm{Par}(\mathcal{M}, \mathrm{Total}(\mathrm{Split}_r(\mathcal{X})))$

Key difficulty is to show sections of restriction idempotents pullback

$$\begin{tikzcd}
& A' \arrow[rr, "s_{\bar{fe}} f r_e"] \arrow[d, hook, "s_{\bar{fe}}"] \arrow[dr, phantom, very near start, "\lrcorner"] & & B' \arrow[d, hook, "s_e"] & \\
& A \arrow[rr, "f"'] & & B \arrow[dr, "e=\bar{e}"'] & \\
A \arrow[ur, "\bar{fe}"'] \arrow[uur, dashed, bend left, "r_{\bar{fe}}"] & & & & B \arrow[uul, dashed, bend right, "r_e"']
\end{tikzcd}$$

Cartesian restriction categories

Product:

$$\begin{tikzcd}
 & \rotatebox{-90}{$\leq$} & A \\
X \arrow[urr, bend left, "f"] \arrow[drr, bend right, "g"'] \arrow[r, dashed, "\langle f,g\rangle"'] & A\times B \arrow[ur, "\pi_0"'] \arrow[dr, "\pi_1"] & \\
 & \rotatebox{90}{$\leq$} & B
\end{tikzcd}$$

total

unique $\langle f,g\rangle$ with

$$\overline{\langle f,g\rangle} = \bar{f}\,\bar{g}$$

Final:

$$X \xrightarrow{!} 1$$

total

$$X \xrightarrow{h} 1$$

$$\Downarrow$$

$$h = \bar{h}\,!$$

Examples (Cartesian restriction categories)

- Sets and partial maps, $\underline{\mathrm{Par}}$
- $\underline{\mathrm{Top}}$, partial maps whose domains are open sets
- $\underline{\mathrm{CRing}}^{op}$ with rational maps

$$\begin{tikzcd} & R[\Sigma^{-1}] & \\ R \arrow[ur, "\text{inverts } \Sigma"'] & & \Sigma \arrow[ul, "\text{arbitrary}"'] \end{tikzcd}$$

main subject of algebraic geometry!

Timed sets and complexity

building an example central to computer science

TSet

Objects : sets

Maps :
$$\begin{tikzcd} X \arrow[r, "f"] \arrow[dr, "|f|"'] & Y \\ & \mathbb{N} \end{tikzcd}$$

|f|(x) says how long it takes to compute f at x

partial maps with

$$\bar{f} = \overline{|f|}$$

identities :
$$\begin{tikzcd} X \arrow[r, "1_X"] \arrow[dr, "0"'] & X \\ & \mathbb{N} \end{tikzcd}$$

composition :
$$\begin{tikzcd} X \arrow[rr, "f"] \arrow[d, "|f|"'] & & Y \arrow[rr, "g"] \arrow[d, "|g|"] & & Z \\ \mathbb{N} \arrow[dr, dash] & & \mathbb{N} \arrow[dl, dash] & & \\ & \oplus \arrow[d, dash] & & & \\ & \mathbb{N} & & & \end{tikzcd}$$

Computer scientists do not care about the exact timing they consider complexity orders:

$$\mathcal{C} \subseteq \mathbb{N} \to \mathbb{N}$$

- $\mathcal{C}$ is a class of monotone maps $(x \le y \Rightarrow f(x) \le f(y))$
- $\mathcal{C}$ is downward closed $Q \le P$ $P \in \mathcal{C} \Rightarrow Q \in \mathcal{C}$
- $P, Q \in \mathcal{C} \Rightarrow PQ \in \mathcal{C}$, $id \in \mathcal{C}$, closed to composition
- additive: $P, Q \in \mathcal{C}$ then $P+Q \in \mathcal{C}$

Examples:

$$\mathcal{L} = \langle \lambda x . n \cdot x \; / \; n \in \mathbb{N} \rangle$$

Linear time

$$\mathcal{P} = \langle \lambda x . a . x^{i} \; / \; a, i \in \mathbb{N} \rangle$$

Polynomial time

$$\mathcal{E} = \langle \lambda x . a_1^{a_2^{\cdot^{\cdot^{\cdot^{a_n^{x}}}}}} \; / \; a_i \in \mathbb{N} \rangle$$

Exponential time

Say $f \preceq_{\mathcal{C}} g \iff f \geq g$

$\exists P \in \mathcal{C}\ \forall x\ |f|(x) \leq P(|g|(x))$

f more defined

f has better $\mathcal{C}$-complexity than g

f has better (or no worse) timing upto $\mathcal{C}$

Say $f \approx_{\mathcal{C}} g$ iff $f \preceq_{\mathcal{C}} g$ and $g \preceq_{\mathcal{C}} f$

$f = g$ and $|f|(x) \leq P(|g|(x))$

$|g|(x) \leq Q(|f|(x))$ for $P, Q \in \mathcal{C}$

Proposition: $=_e$ is a congruence on TSet which gives the category $TSet_e$

So in $\underline{TSet}_{\mathcal{P}}$ two maps f and g are identified if $f = g$ and $\exists P, Q \in \mathcal{P}$ with $P(|f|(x)) \geq |g|(x)$ and $Q(|g|(x)) \geq |f|(x)$

i.e they have the same polytime complexity!
(or e-time)

Proposition

$\underline{TSet}_e$ for any complexity order e is a Cartesian restriction category (in fact a distributive rest cat).

$$\frac{f : X \longrightarrow Y}{\bar{f} = (\bar{f}, \|f\|)}$$

Some timing

Now consider $Split_r(TSet_e)$ .....

$\mathrm{Spot}_r(\mathrm{TSet}_e)$:

idempotents give size of elements

Objects: $(X, |-|_X)$ oo to equiv

Maps: $(X, |-|_X) \xrightarrow[(f, \|f\|)]{} (Y, |-|_Y)$

Magically objects obtain a size

Total maps:

$$(X, |-|_X) =_e (1, \|f\|)$$

$$\Updownarrow$$

$(f, \|f\|)$ is e-timed

cost of map must accomodate size of input and size of output as in splittings

$$(X, |-|_X)(f, \|f\|)(Y, |-|_Y) =_e (f, \|f\|)$$

total maps are P-timed (when $e = P$.)

S☺ : $\mathrm{Split}_r(\mathrm{TSet}_{\mathbb{P}})$

is a (distributive) Cartesian restriction category whose total maps are precisely poly-timed maps .....

How do we isolate PTime ?

This is where Turing categories come in !

Turing categories

Defn A Turing category is a cartesian restriction category with a Turing structure $(T, \bullet_{XY})$

- T is a Turing object
- $\bullet_{X,Y}$ is an application map such that for every $f : A \times X \longrightarrow Y$

$$\begin{tikzcd}
T \times X \arrow[r, "\bullet_{XY}"] & Y \\
A \times X \arrow[u, "c_f \times 1"] \arrow[ur, "f"'] &
\end{tikzcd}$$

there is a total map c_f, the code of f with $c_f \times 1 \bullet_{XY} = f$.

- In a Turing category every object is a retract of the Turing object:

$$\begin{tikzcd}
T \times 1 \arrow[r, "\bullet_{1A}"] & A \\
A \times 1 \arrow[u, "C_{\pi_0} \times 1"] \arrow[ur, "\sim" description] \arrow[ur, "\pi_0"'] &
\end{tikzcd}$$

- In fact, it suffices to have every object a retraction of the Turing object, T, AND one application $T \times T \xrightarrow{\bullet} T$
- One can "stack" applications:

$$T \times T \times T \times T \xrightarrow{(\bullet \times 1 \times 1)(\bullet \times 1)\bullet} T$$

Prop. Every Turing object is a partial combinatory algebra (PCA).

PCA $(X, \cdot, s, k)$

$$X \times X \xrightarrow{\cdot} X$$ application

$$\left.\begin{array}{l} 1 \xrightarrow{k} X \\ 1 \xrightarrow{s} X \end{array}\right\}$$ combinators (total)

such that $(k \cdot x) \cdot y = x$

$$((s \cdot x) \cdot y) \cdot z = (x \cdot z) \cdot (y \cdot z)$$

$s \cdot x \cdot y$ total, $k \cdot x$ total

Proposition:

relative PCA

and a set of points P of X

Given a PCA, $\mathcal{X} = (X, \bullet, s, k)$, in a Cartesian restriction category the $\mathcal{X}$-computable maps form a Turing category.

When is a map $f : X \times \ldots \times X \longrightarrow X$ $\mathcal{X}$-computable?

Answer: when there is a "code", c_f, for f

$$\begin{tikzcd}
X \times X \times \ldots \times X \arrow[r, "(\bullet \times \ldots \times 1) \ldots \bullet"] & X \\
X \times \ldots \times X \arrow[u, "c_f \times 1"] \arrow[ur, "f"'] &
\end{tikzcd}$$

this can use the points.

So: to obtain a Turing category it suffices to identify a DCA (and take <u>its</u> computable maps)

Question: can we identify a DCA in $\text{Split}(\text{TSet}_{\mathcal{P}})$?

Answer: YES oo

classical complexity theory shows steps of Turing machines in PTime

⇒

There is a Turing category of P-time maps

Why is this interesting?

⇓ It unifies computability and complexity!

Both are theory of Turing cats

Amazingly there are linear time Turing categories, ...

can get very low complexity ...

Pieter helped determine exactly which categories can be the total maps of a Turing category ...

Things left undone ....

- The initial Turing category
  (rewriting theory - handling partiality)
- Reflexive Turing objects
- Computability theory of special Turing categories
- Computability in special categories
  e.g. SMC, differential cats,
  multi-cats, poly-cats, ...

lots of different notions of computability

~~Church–Turing thesis~~

END!

Thank you for listening!