Differential Turing Categories

Jonathan Gallagher
with Robin Cockett *

June 13, 2011

* University of Calgary. Partially supported by NSERC.

Goal

This talk will introduce differential Turing categories and their basic structural properties.

Background

Resource Calculus

1992 Milner: Λ translated to Π; asked “What is the semantics induced by this translation?”[10]

1993 Boudol: Λ refined: terms in the ”argument position” are bags of terms [3] [5].

1999 Boudol *et al*: The “lambda calculus with multiplicities” further refined [4].

2009 Pagani and Tranquilli: The resource lambda calculus; resource control and nondeterminism [11].

Differential Calculus

2002 Ehrhard: Models of linear logic where all functions can be differentiated [8].

2003 Ehrhard and Regnier: Formalized the above notion syntactically; differential Λ [9].

2006 Blute *et al*: The (monoidal) categorical setting for differential structure [2].

2009 Blute *et al*: The Cartesian categorical setting for differential structure [1].

2010 Bucciarelli *et al*: The connection between resource Λ and differential Λ made exact. Models of differential Λ are models of resource Λ. Main source of models: “linear” reflexive objects in a Cartesian closed differential category [6].

Turing Categories

- Differential structure has been proposed as a Curry-Howard-Lambek style correspondence for distributed computation. However, the role differential structure plays in computability theory has not been worked out.
- Turing categories give a way to view computability theory in terms of categorical structure [7]. They also give a categorical framework we can use to combine differential structure with computability theoretic structure.

Differential Turing Categories

Structures involved

- Restriction structure (for partiality)
- Differential structure
- Turing structure

Restriction structure

A **restriction category** $\mathbb{X}$ has a combinator

$$\frac{f : A \longrightarrow B}{\overline{f} : A \longrightarrow A,}$$

that satisfies the following four axioms

R.1 $\overline{f}\, f = f$ R.3 $\overline{f}\, \overline{g} = \overline{\overline{f}\, g}$
R.2 $\overline{f}\, \overline{g} = \overline{g}\, \overline{f}$ R.4 $f\overline{h} = \overline{fh}\, f$

Product diagrams in a restriction category should not commute on the nose; e.g.,

$$\langle f, g\rangle \pi_0 = \overline{g}\, f.$$

Differential Structure

A **differential restriction category** is a Cartesian restriction category $\mathbb{X}$, with each $\mathbb{X}(A, B)$ a commutative monoid that is preserved by products, and with a differential combinator

$$\frac{f : A \longrightarrow B}{D[f] : A \times A \longrightarrow B}$$

which satisfies nine axioms.
A key definition is:

Definition 1. *A* **linear map** *in a differential restriction category is f such that $D[f] \smile \pi_0 f$* [1]*.*

Proposition 1. *The linear maps of a differential restriction category form a subcategory.*

---

[1] $a \smile b$ means $\overline{a}\, b = \overline{b}\, a$

Turing Categories

Turing Categories generalize computable functions on $\mathbb{N}$.

- Programs assigned natural numbers
- There is a program ϕ which "compiles and runs" natural numbers
- A function f is computable when $f(x_1, \ldots, x_n) = \phi(n, x_1, \ldots, x_n)$

Definition 2. *A* **Turing category** *is a Cartesian restriction category, with an object T where for every B, C there is a map $\phi : T \times B \longrightarrow C$ such that for any $f : A \times B \longrightarrow C$ there is a total map $\hat{f}$ such that the following diagram commutes.*

$$\begin{tikzcd} T \times B \arrow[r, "\phi"] & C \\ A \times B \arrow[u, "\hat{f} \times 1"] \arrow[ur, "f"'] & \end{tikzcd}$$

The maps ϕ are called **Turing morphisms**, and the maps $\hat{f}$ are called **codes**.

Proposition 2. *Let $\mathbb{X}$ be a Cartesian restriction category. Then $\mathbb{X}$ is a Turing category iff there is an object T such that all other objects A are retracts of T and $\phi : T \times T \longrightarrow T$ is universal.*

The proof of the above theorem involves splitting certain idempotents, and also using these splittings to construct Turing morphisms.

Another View on Turing Categories

Let $\mathbb{X}$ be a Cartesian restriction category. An **applicative system** is a pair $(A, \bullet : A \times A \longrightarrow A)$. **Combinatory completeness** means that every $(A, \bullet)$ word can be represented by a total point.

Theorem 1 (Curry-Schonfinkel)**.** *An applicative system* $(A, \bullet)$ *is combinatory complete (aka a PCA) iff the computable maps with respect to* A *form a Turing category.*

Further, we have the following:

$$\begin{tikzcd} \mathbb{Y} \arrow[d, "F"'] & (A, \bullet, \mathcal{V}) \arrow[d, mapsto] \\ \mathbb{X} & (F(A), \bullet, \mathcal{V}) \end{tikzcd}$$

with F faithful, $V = \{F(a) : 1 \longrightarrow FA | a \in \text{Total}(1, A)\}$, so that Split(Comp(FA,$\bullet$,$\mathcal{V}$) is equivalent to $\mathbb{Y}$.

Combining differential and Turing structure

Interested in differential structure and computability; i.e. the PCA's...
How can these two structures be combined?

$$(A, \bullet, s, k, 0, +, D[\bullet])$$

3 views:

Do nothing	$\rightsquigarrow$ Differential restriction category with a Turing subcategory
Assume that $D[\bullet]$ has a code	$\rightsquigarrow$ Differential restriction category with Turing object
Let $D[\_]$ have a code	$\rightsquigarrow$ Structure goes outside the differential category

The Middle Road

Rationale: in the differential lambda models, we have a code for $D[\bullet]$.
Sanity Check: the linear, additively closed maps can be split in a differential restriction category, but not all idempotents can be split
Also, differential lambda models have split, linear idempotents.

Definition 3. *Let $\mathbb{X}$ be a Turing category and a differential restriction category. $\mathbb{X}$ is a* **uniform differential Turing category** *when ϕ is linear in its first argument and there is a code for π_0 that is linear in its first argument.*

Uniform Differential Turing Categories

Proposition 3.

- *$\mathbb{X}$ is a differential Turing category iff π_0 has a code that is linear in its first variable, $\phi : T \times T \to T$ is linear in its first argument and universal, and every object is a linear retract of T.*
- *Uniform differential PCAs and uniform differential Turing categories correspond.*
- *Linear reflexive objects in Cartesian closed differential categories generate (total) uniform differential Turing categories.*

Conclusion

References

[1] R.F Blute, J.R.B. Cockett, and R.A.G. Seely. Cartesian differential categories. *Theory and Application of Categories*, 22:622–672, 2009.

[2] Richard Blute, Robin Cockett, and Robert Seely. Differential categories. *Mathematical Structures in Computer Science*, 16:1049–1083, December 2006.

[3] G. Boudol. The lambda-calculus with multiplicities. In *CONCUR ’93 Proceedings of the 4th International Conference on Concurrency Theory*, 1993.

[4] G. Boudol, P. Curien, and C. Lavatelli. A semantics for lambda calculi with resources. *Mathematical Structures in Computer Science*, 9:437–482, 1999.

[5] G. Boudol and Cosimo Laneve. Lambda-calculus, multiplicities, and the pi-calculus. Technical report, Institut National de Recherche en Informatique et en Automatique, 1995.

[6] Antonio Bucciarelli, Thomas Ehrhard, and Giulio Manzonetto. Categorical models for simply typed resource calculi. *Electronic Notes in Theoretical Computer Science*, 265:213–230, September 2010.

[7] J.R.B Cockett and P. Hofstra. Introduction to turing categories. *Annals of Pure and Applied Logic.*, 156.:183–209., 2008.

[8] Thomas Ehrhard. On kothe sequence spaces and linear logic. In *Mathematical Structures in Computer Science*.

[9] Thomas Ehrhard and Laurent Regnier. The differential lambda calculus. *Theoretical Computer Science*, 309:1–41, 2003.

[10] R. Milner. Functions as processes. *Mathematical Structures in Computer Science*, 2:119–141, 1992.

[11] Sixth ASIAN Symposium on Programming Languages and Systems. *Parallel Reduction in Resource Lambda-Calculus*, 2009.