Some Categorical Structures and Their Rôle in Differential Calculus

William Steingartner, Štefan Korečko, Jaroslav Porubän
Faculty of Electrical Engineering and Informatics
Technical University of Košice
Letná 9, 042 00 Košice, Slovakia
{william.steingartner, stefan.korecko, jaroslav.poruban}@tuke.sk

Abstract. *The development of mathematics stands as one of the most important achievements of humanity, and the development of the calculus, both the differential calculus and integral calculus is one of the most important achievements in mathematics. Differential calculus is about finding the slope of a tangent to the graph of a function, or equivalently, differential calculus is about finding the rate of change of one quantity with respect to another quantity. On the other hand, integration is an important concept in mathematics and, together with its inverse, differentiation, is one of the two main operations in calculus. Integrals and derivatives became the basic tools of calculus, with numerous applications in science and engineering. Category theory is a mathematical approach to the study of alge-braic structure that has become an important tool in theoretical computing science, particularly for semantics-based research. The notion of a limit in category theory generalizes various types of universal constructions that occur in diverse areas of mathematics. In our paper we show how to represent some parts of infinitesimal calculus in categorical structures.*

Keywords. CECIIS, conference paper, template

1 Introduction

The science and technology are nowadays the unthinkable parts of global world. Their expansion simplify the work in many branches and of course the daily life [7]. In day to day life we are often interested in the extent to which a change in one quantity affects a change in another related quantity. This is called a rate of change. Differential calculus, a part of mathematics, is about describing in a precise fashion the ways in which related quantities change. Differential and integral calculus are dual fields and they together form a base for infinitesimal calculus [4, 5, 6, 9, 12, 15]. Infinitesimal calculus is a part of mathematics concerned with finding slope of curves, areas under curves, minima and maxima, and other geometric and analytic problems.

On the other hand, category theory is an area of study in mathematics that examines in abstract way the properties of particular mathematical concepts by formalizing them as collections of objects and arrows (called morphisms, although this term also has a specific, non category-theoretical meaning), where these col-lections satisfy some basic conditions [2, 3]. Category theory is a branch of mathematics that has been developed over the last fifty years, and it has concerned with the study of algebraic structures [11]. Category theory, a branch of abstract algebra, has found many applications in mathematics, logic, and computer science. Like such fields as elementary logic and set theory, category theory provides a basic conceptual apparatus and a collection of formal methods useful for addressing certain kinds of commonly occurring formal and informal problems, particularly those involving structural and functional considerations.

Nowadays, many significant areas of mathematics and informatics can be formalized as categories, and the use of category theory allows many intricate and subtle mathematical results in these fields to be stated, and proved, in a much simpler way than without the use of categories. Functions are mostly represented by morphisms from a domain into a codomain of a function, and we can consider them as the structures enclosable into category [13].

In our paper we show the *rôle* of categorical structures in infinitesimal calculus - we construct a diagram of functions and we show how to find a categorical limit of that diagram. In the second part of the article we show how to express derivatives in the another way in categories.

2 Motivation

Nowadays, there are always some software products that are less reliable than their more traditional engineering counterparts. Part of the reason is that, being digital, they are readily reused in unexpected situations, a problem seldom experienced by "physical" products. Even if software works correctly in one context it may fail arbitrarily when reused in another. There exists approach where differential equations are used to specify behavior and the calculus used to anal-

yse and simulate it. As a result, engineering products can be accounted for even before they are commissioned. The aim is then accountably correct information systems. Formal specifications form the first important technique; by capturing precisely the functional behaviour of a system they promote its accurate reuse in differing contexts. Formal descriptions at varying levels of abstraction are the equivalent of approximation in the calculus, and they enable a realistically complex system to be understood or developed incrementally top-down. Then criteria for conformance of descriptions at differing levels of abstraction underpin the correctness of that enterprise. Technical contributions produce laws that can be automated and readily used by the system designer to validate designs, and semantic models to establish the soundness of those laws.

The programme's methodology incorporates the inseparability of theory and realistic application: theory is needed to address pressing problems; application is required to validate the theory. It endorses an indivisible relationship between research, development and knowledge-transfer/pedagogy. Projects where this methodology is particularly important include the following:

- Web Services;
- Mobile Phone as a Platform for Development Applications;
- Distributed Systems;
- Model-Driven Development of Component-Based Software;
- Emergence;
- Probabilistic Behaviour;
- Software Engineering in Health Care;

Numbers of software applications, such as those in healthcare, governance and financial industry, are becoming increasingly komplex nowadays, due to the rapid increase in the power of hardware systems and advances in communication network technologies. The failure of such software would be expensive and inconvenient - even chaotic - but it could also, in an increasing number of cases, impinge on public safety. There are many cases where the importance of the research on software engineering foundations and software development methodologies, as well as their applications in sharded, interoperable and trustworthy information systems have been recognized. Some methods, for instance such of model-driven development which studies how models of large software systems are divided into smaller models across competing design concerns, and how models are refined through different levels of abstraction, are actual nowadays. Techniques and tool support for model construction, decomposition/composition, validation and transformations have been developed. The techniques and their tool support are developed based on a unified theory of program semantics that allows them to be applied consistently in different phases of a model-driven development process. The semantic method needed forthat approach can be chosen from the most usual used like action semantics, denotational or operational semantics, also categorical semantics which is a new approach in some ways. The rigor and techniques of abstraction are effective for mastering the complexity of the process, and critical to assuring the trustworthiness of the system developed.

Depends on whether we want to really understand the application area, or just write code based on specifications given to programmer by another engineer. Many communication systems and control systems, for example, are based on engineering topics which rely heavily on integral and differential calculus.

Since computers work with fixed-precision numbers (integers or floating-point), the math-book-style calculus of continuous functions has to be translated into forms that are computable using discrete functions. For example, in digital signal processing you often see a discrete convolution (used in filtering, etc.) which is more or less the same operation as a convolution, only replacing the integrals with summations. On the other hand, typical computer work like database design, operating systems, web-based applications, point-of-sale terminals, etc. - these type of algebraic/logical systems would rarely use integral or differential calculus. Only in the case when the knowledge of probability and statics is required and they are useful in several areas of computer science.

Finally, there are other types of "calculus" than just integral and differential, for example the lambda calculus of Alonzo Church which is fundamental to topics in computability.

Several software fields use the calculus directly: machine learning, software for science and engineering, physics engines in computer games. Fields that use numerical methods generally prefer or require some understanding of calculus and linear algebra. These include:

- computer graphics, including computer games;
- anything that uses statistics;
- engineering and science;
- quantitative finance.

We can say that there are a fairly substantial minority of high-quality technical programming positions. But often it is necessary to know about limits and derivatives to understand what Big-O notation means, and why for example an exponential algorithm will take forever, while a quadratic algorithm might be feasible.

Based on all these knowledge it is convenient and also necessary to formulate differential calculus by means of categorical structures because the category

theory offers a unique toolkit to formulate many aspects of the some areas and it allows us to develop further at an appropriate and useful level of abstraction and generality [14].

In the next sections we define the basic notions from category theory needed for our approach and we express the modeling of some parts of the differential calculus in categories. We enclose functions and their derivatives into special category where objects are functions, the so called arrow category. For defining the relations between functions and their derivatives we define a codomain functor.

3 Basic Notions

In mathematics, and especially in category theory, a commutative diagram is a diagram of objects and morphisms such that all directed paths in the diagram with the same start and endpoints lead to the same result by composition. Commutative diagrams play the crucial *rôle* in category theory that equations play in algebra [3]. A general form of commutative diagram is depicted in Fig. 1.

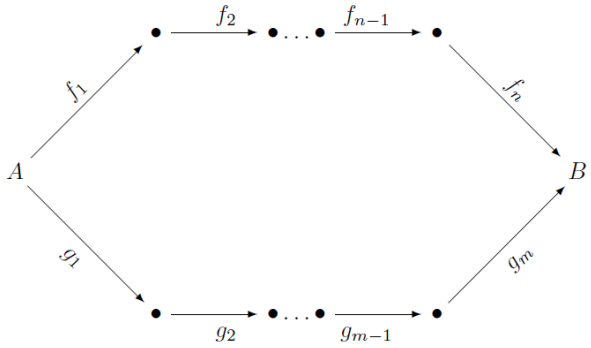

Figure 1: Commutative diagram

Diagram in Fig. 1 expresses the following equality:

$$f_n \circ f_{n-1} \circ \ldots \circ f_2 \circ f_1 = g_m \circ g_{m-1} \circ \ldots \circ g_2 \circ g_1$$

for $m, n \in \mathbb{N}$.

A category $\mathscr{C}$ is mathematical structure consisting of objects, e.g. $A, B, \ldots$ and morphisms of the form

$$f : A \to B$$

between them. Every object has the identity morphism $id_A : A \to A$ and morphisms are composable. Because the objects of category can be arbitrary structures, categories are useful in computer science [3, 13], where we often use more complex structures not expressible by sets. These data are required to satisfy the following laws:

- Associativity:

$$h \circ (g \circ f) = (h \circ g) \circ f$$

for all morphisms $f : A \to B$, $g : B \to C$ and $h : C \to D$.

- Unit:

$$f \circ id_A = id_B \circ f = f$$

for all $f : A \to B$ such that the diagram at Fig. 2 commutes.

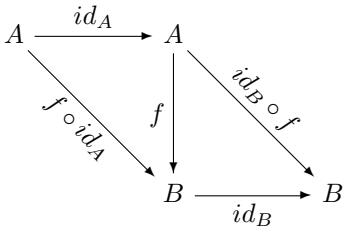

Figure 2: Diagram of unit in category

In our approach we construct special type of category which is called *arrow category* or *comma category*. We denote it $\mathscr{C}^{\to}$. It is defined over a base category $\mathscr{C}$ and it holds, that its objects are the morphisms of $\mathscr{C}$, and its morphisms are commuting squares (diagrams) in $\mathscr{C}$ [1].

Homomorphisms between categories are called functors, e.g. a functor

$$F : \mathscr{C} \to \mathscr{D}$$

from a category $\mathscr{C}$ into a category $\mathscr{D}$ (or between categories) considered as a structure-preserving mapping between categories as follows:

1. $F(f : A \to B) = F(f) : F(A) \to F(B)$;
2. $F(id_A) = id_{F(A)}$;
3. $F(g \circ f) = F(g) \circ F(f)$.

That is, F preserves domains and codomains, identity arrows, and compostion.

Codomain functor is the special functor defined for arrow category. Codomain functor is always defined for the arrow category and the appropriate base category [8]. It is defined as follows:

$$F : \mathscr{C}^{\to} \to \mathscr{C}$$

which assigns to each object from arrow category its codomain and to each morphism from arrow category the morphism between appropriate codomains.

4 Categorical Structures in Differential Calculus

The practical applications of differential calculus are very widely ranging. Suffice to say that differential calculus is an indispensable tool in every branch of science and engineering. In elementary mathematics there are two main applications of differential calculus. One is to help in sketching curves, and the other is in optimisation problems [15].

We could also say that differential calculus is a procedure for finding the exact derivative directly from the formula of a function, without having to use graphical methods. In practise we use a few rules that tell us how to find the derivative of almost any function that we are likely to encounter.

The derivative of a function at a chosen input value describes the best linear approximation of the function near that input value. Informally, the derivative is the ratio of the infinitesimal change of the output over the infinitesimal change of the input producing that output. For a real-valued function of a single real variable, the derivative at a point equals the slope of the tangent line to the graph of the function at that point. The process of finding a derivative is called differentiation. The reverse process is called antidifferentiation. The fundamental theorem of calculus states that antidifferentiation is the same as integration. Differentiation and integration constitute two fundamental operations in single-variable calculus.

4.1 Derivatives in category

For defining the derivatives in category we consider the arrow category $\mathscr{D}er^{\rightarrow}$ over the category of sets. The arrow category, also called a comma category is defined for each category [8]. Any arrow category $\mathscr{C}^{\rightarrow}$ over the base category $\mathscr{C}$ has as objects all morphisms from the base category. Then morphisms of arrow category are morphisms defined as tuples between domains and codomains of objects.

Because the derivative f' to function f is also function, we can construct the base category as category of sets $\mathscr{D}er$, where objects are sets (the domains and codomains of functions) and morphisms are functions. The arrow category $\mathscr{D}er^{\rightarrow}$ over a base category of sets exist according to the definition, so we can define any function as object in arrow category. Morphisms of $\mathscr{D}er^{\rightarrow}$ are the operations of differentiation which assign to any function f its derivative f' according to the definition.

The category $\mathscr{D}er^{\rightarrow}$ is defined as follows:

1. objects are functions $f, g, h, \ldots$ and their derivatives $f, f', \ldots$ The functions are arrows in the base category $\mathscr{D}er$;

2. morphisms are tuples (D, C) of mappings between domains and codomains of objects:

$$(D, C) : f \rightarrow f',$$

is a differentiation operation which assigns to function f its derivative f', where D is a mapping between domains and C between codomains. The tuple (D, C) we denote as morphism der.

3. for any object f there is defined an identity morphism which sends any object to itself:

$$id_f : f \rightarrow f.$$

4. composition of morphisms is defined as follows: for two morphisms $der_1 : f \rightarrow f'$ and $der_2 : f' \rightarrow f''$, their composition $der_2 \circ der_1 : f \rightarrow f''$ is a new morphism that assigns to function f its the second derivative, because the first derivative of f' is defined as follows,

$$(f') = f''.$$

5. composition of morphisms is associative operation defined as follows:

$$(der_3 \circ der_2) \circ der_1 = der_3 \circ (der_2 \circ der_1).$$

The codomain functor $\mathcal{C}od : \mathscr{D}er^{\rightarrow} \rightarrow \mathscr{D}er$ always exists. It assigns to any object f in $\mathscr{D}er^{\rightarrow}$ its codomain $cod(f)$ in $\mathscr{D}er$, and to any morphism $der : f \rightarrow f'$ in $\mathscr{D}er^{\rightarrow}$ the arrow between codomains of der in $\mathscr{D}er$ (3).

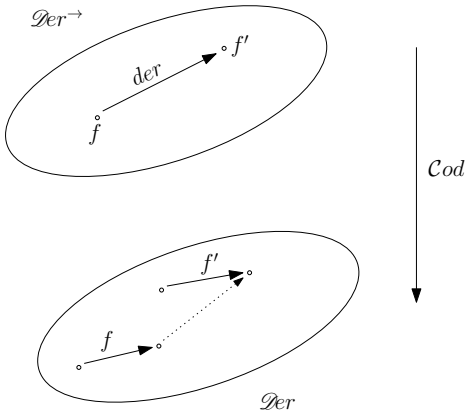

Figure 3: The codomain functor between categories

4.2 Example of expressing the derivatives in category

Let's take the function of natural logarithm as an example. This function has the following specification:

$$f : (0; \infty) \rightarrow \mathbb{R}.$$

The logarithmic function is a morphism with the domain of positive real numbers and the codomain is the

set of all real numbers. When finding the derivative of function, we use rules for derivation. For logarithmic function its derivative is as follows:

$$f'(x) = (\ln x)' = \frac{1}{x},$$

which is well-known basic reciprocal linear rational function, simply called reciprocal function. This function is defined on the set of non-zero reals, and it sends every real number to its reciprocal value, i.e. its multiplication inverse. The reciprocal function $f'(x) = \frac{1}{x}$ has the specification

$$f' : \mathbb{R}\setminus\{0\} \to \mathbb{R}\setminus\{0\},$$

where both sets - the domain and the codomain are identical, namely the set of nonzero real numbers. The second derivative of $f(x) = \ln x$ is the first derivative of the reciprocal function $f'(x) = \frac{1}{x}$ and it is the function

$$f''(x) = -\frac{1}{x^2},$$

which has the specification

$$f'' : \mathbb{R}\setminus\{0\} \to (-\infty; 0).$$

All functions listed above are depicted in the Fig. 4.

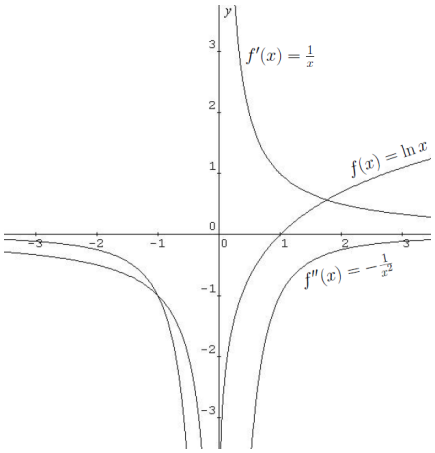

Figure 4: Natural logarithm function and its the first and the second derivative

It holds that the second derivative of function $\ln x$ is a function $f''(x) = -\frac{1}{x^2}$ which is the first derivative of the reciprocal function $f'(x) = \frac{1}{x}$,

$$f''(x) = (f'(x))'.$$

In the category $\mathscr{Der}$, the functions f, f' and f'' are the following morphisms in category:

$$f : (0; \infty) \to \mathbb{R}$$
$$f' : \mathbb{R}\setminus\{0\} \to \mathbb{R}\setminus\{0\}$$
$$f'' : \mathbb{R}\setminus\{0\} \to (-\infty; 0)$$

with their domains and codomains

$$dom(f) = (0; \infty) \quad cod(f) = \mathbb{R}$$
$$dom(f') = \mathbb{R}\setminus\{0\} \quad cod(f) = \mathbb{R}\setminus\{0\}$$
$$dom(f'') = \mathbb{R}\setminus\{0\} \quad cod(f'') = (-\infty; 0)$$

and the sets $(0; \infty)$, $\mathbb{R}$, $\mathbb{R}\setminus\{0\}$ and $(-\infty; 0)$ are objects in this category. For any object A the identity morphism is defined as follows:

$$id_A : A \to A,$$

and morphisms are composable according to the obvious rules for composition of functions.

In category $\mathscr{Der}^{\to}$ the functions f, f' and f'' are objects in this category. The morphisms between objects are differentations, denoted der:

$$der : f \to f',$$

and they are defined as tuples of two arrows for mapping of domains and codomains of objects, $der = (D, C)$, where

$$D : dom(f) \to dom(f'), C : cod(f) \to cod(f').$$

The identity is defined for each object,

$$id_f : f \to f,$$

is the zero derivation which is an identity mapping: it sends any function to itself. The composition of functions represents the gradual increasing of order of differentiation. Here, for the functions $f'(x) = \frac{1}{x}$ and $f''(x) = -\frac{1}{x^2}$, the differentiations are:

- $der_1 : f \to f'$ for the first derivative of f;
- $der_2 : f' \to f''$ for the first derivative of f'.

By composition of der_1 and der_2 we obtain a new morphism $der_2 \circ der_1$ (which we can also denote der_{12}):

$$der_2 \circ der_1 : f \to f'',$$

and it is the second derivative of f:

$$f''(x) = (\ln x)'' = -\frac{1}{x^2}.$$

For the composition it holds the following commutative diagram (Fig. 5).

From the diagram at Fig. 5 the following equality holds:

$$C_2 \circ C_1 \circ f = f'' \circ D_2 \circ D_1.$$

The relation between categories $\mathscr{Der}$ and $\mathscr{Der}^{\to}$ is expressed by the codomain functor

$$\mathcal{C}od : \mathscr{Der}^{\to} \to \mathscr{Der}.$$

This functor assigns to every object in $\mathscr{Der}^{\to}$ its codomain - a set in $\mathscr{Der}$:

$$\mathcal{C}od(f) = B$$

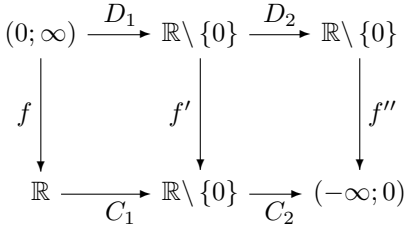

Figure 5: Diagram of the second derivative

for any morphism $f : A \to B$, and any morphism in $\mathscr{D}er^{\to}$ it sends to a morphism between the appropriate codomains:

$$\mathcal{C}od(der) = C$$

for any $der : f \to f'$, where morphism C is

$$C : cod(f) \to cod(f').$$

5 Conclusion

In this paper we have showed our approach for derivatives. The aim of our paper was an illustration of usability of categories in various fields of mathematics. Our next goal is to investigate how to implement mathematical expressions of some parts of infinitesimal calculus in practical approach [10].

6 Acknowledgments

This work has been supported by KEGA grant project No. 050TUKE-4/2012: "Application of Virtual Reality Technologies in Teaching Formal Methods".

References

[1] Adámek, J., Herrlich, H. and Strecker, G. E. Abstract and Concrete Categories. John Wiley & Sons, 1990.

[2] Awodey, S. *Category Theory*, Carnegie Mellon University (2005)

[3] Barr, M., and Wells, C. *Category Theory for Computing Science*, Prentice Hall International, ISBN 0-13-120486-6 (1990)

[4] Bourbaki, N. *Integration I*, Springer Verlag (2004)

[5] Burton, D. M. *The History of Mathematics: An Introduction* (6th ed.), McGraw-Hill (2005)

[6] Forster, O. Analysis 1. *Differential- und Integralrechnung einer Vernderlichen.* 7. Au. Vieweg-Verlag, ISBN 3-528-67224-2 (2004)

[7] Hrušková, R. *Social Skills in the Context of Science and Technology*, In: Proceedings of scien-tific works - Veda a technika v procese globalizácie a humanizácia vzdelávania na technických univerzitách, Slovak University of technology in Bratislava, 2009 (in Slovak)

[8] Jacobs, B. *Categorical Logic and Type Theory*. No. 141 in Studies in Logic and the Foundations of Mathematics. North Holland, Amsterdam, 1999

[9] Keisler, H.J. *Elementary Calculus: An Infinitesimal Approach* (On-line Edition). Copyright 2000/revised 2012, University of Wisconsin, http://www.math.wisc.edu/k̃eisler/calc.html

[10] Luković I., Ristić S., Popović A., Mogin P. *An approach to the platform independent specification of a business application*, In: Proceedings of the 23rd Central European Conference on Information and Intelligent Systems - CECIIS'2012, 19th-21st Sept 2012, University of Zagreb, Varaždin, Croatia, pp.449-456 (2012)

[11] Novitzká, V., Slodičák, V. *Categorical structures and their application in informatics*, Equilibria, Košice, in Slovak (2010)

[12] Ristić S., Luković I., Aleksić S., Banović J., Al-Dahoud A. An approach to the specification of user interface templates for business applications. In Proceedings of the Fifth Balkan Conference in Informatics (BCI '12). ACM, New York, NY, USA, 124-129 (2012).

[13] Slodičák, V., Macko, P. *New approaches in functional programming using algebras and coalgebras*, In European Joint Conferrences on Theory and Practise of Software-ETAPS 2011, Universität des Saarlandes, Saarbrücken, Germany, pp. 13-23, ISBN 978-963-284-188-5 (2011)

[14] Tarlecki, A. *Categories, Institutions, Abstract Model Theory, and Software Specification* Workshop on Applied and Computational Category Theory - ACCAT/ETAPS 2013, Rome, Italy (2013)

[15] Thomas, C. *Introduction to Differential Calculus*. University of Sydney (1997)