Discrete Applied Mathematics 239 (2018) 45–61

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Enumerating lambda terms by weighted length of their De Bruijn representation☆

Olivier Bodini [a], Bernhard Gittenberger [b,*], Zbigniew Gołębiewski [c]

[a] Laboratoire d'Informatique de Paris-Nord, Université de Paris-Nord, 99, avenue Jean-Baptiste Clément, 93430 Villetaneuse, France
[b] Institute for Discrete Mathematics and Geometry, Technische Universität Wien, Wiedner Hauptstraße 8-10/104, A-1040 Wien, Austria
[c] Department of Computer Science, Wrocław University of Science and Technology, ul. Wybrzeze Wyspianskiego 27, 50-370, Wrocław, Poland

ARTICLE INFO

Article history:
Received 5 July 2017
Received in revised form 21 December 2017
Accepted 25 December 2017
Available online 1 February 2018

Keywords:
Lambda term
Asymptotic enumeration
Generating function
Infinitely nested radical
Boltzmann sampling

ABSTRACT

John Tromp introduced the so-called 'binary lambda calculus' as a way to encode lambda terms in terms of 0–1-strings using the de Bruijn representation along with a weighting scheme. Later, Grygiel and Lescanne conjectured that the number of binary lambda terms with m free indices and of size n (encoded as binary words of length n and according to Tromp's weights) is $o\left(n^{-3/2}\tau^{-n}\right)$ for $\tau \approx 1.963448\ldots$. We generalize the proposed notion of size and show that for several classes of lambda terms, including binary lambda terms with m free indices, the number of terms of size n is $\Theta\left(n^{-3/2}\rho^{-n}\right)$ with some class dependent constant ρ, which in particular disproves the above mentioned conjecture.

The methodology used is setting up the generating functions for the classes of lambda terms. These are infinitely nested radicals which are investigated then by a singularity analysis.

We show further how some properties of random lambda terms can be analyzed and present a way to sample lambda terms uniformly at random in a very efficient way. This allows to generate terms of size more than one million within a reasonable time, which is significantly better than the samplers presented in the literature so far.

1. Introduction

The objects of our interest are lambda terms, which are the basic objects of lambda calculus. For a thorough introduction to lambda terms and lambda calculus we refer to [1]. This paper will not deal with lambda calculus and no understanding of lambda calculus is needed to follow our proofs. We will instead be interested in the enumeration of lambda terms, the study of some properties of random lambda terms and the efficient generation of terms of a given size uniformly at random.

A lambda term is a formal expression which is described by the grammar $M ::= x \mid \lambda x.M \mid (M\,M)$ where x is a variable, the operation $(M\,M)$ is called application, and using the quantifier λ is called abstraction. In a term of the form $\lambda x.M$ each occurrence of x in M is called a bound variable. We say that a variable x is free in a term M if it is not in the scope of any abstraction. A term with no free variables is called closed, otherwise open. Two terms are considered equivalent if they are identical up to renaming of the variables, *i.e.*, more formally speaking, they can be transformed into each other by α-conversion. We shall always mean 'equivalence class w.r.t. α-conversion' whenever we write 'lambda term'.

☆ A preliminary version of this work appeared in the proceedings of STACS 2016.

* Corresponding author.
E-mail addresses: olivier.bodini@lipn.univ-paris13.fr (O. Bodini), gittenberger@dmg.tuwien.ac.at (B. Gittenberger), zbigniew.golebiewski@pwr.edu.pl (Z. Gołębiewski).

https://doi.org/10.1016/j.dam.2017.12.042
0166-218X/

In this paper we are interested in counting lambda terms whose size corresponds to their De Bruijn representation (*i.e.*, what was called 'nameless expressions' in [10]).

Definition 1. A De Bruijn representation is a word described by the following specification:

$$M ::= n \mid \lambda M \mid M\,M$$

where n is a positive integer, called a De Bruijn index. Each occurrence of a De Bruijn index is called a variable and each λ an abstraction. A variable n of a De Bruijn representation w is bound if the prefix of w which has this variable as its last symbol contains at least n times the symbol λ, otherwise it is free. The abstraction which binds a variable n is the nth λ before the variable when parsing the De Bruijn representation from that variable n backwards to the first symbol.

For the purpose of the analysis we will use the notation consistent with the one used in [2]. This means that the variable n will be represented as a sequence of n symbols, namely as a string of $n-1$ so-called 'successors' S and a so-called 'zero' 0 at the end. Obviously, there is a one to one correspondence between equivalence classes of closed lambda terms (as described in the first paragraph) and De Bruijn representations. For instance, the De Bruijn representation of the lambda term $\lambda x.\lambda y.xy$ (which is e.g. equivalent to $\lambda a.\lambda b.ab$ or $\lambda y.\lambda x.yx$) is $\lambda\lambda 21$; using the notation with successors this becomes $\lambda\lambda((S0)0)$.

Since we are interested in counting lambda terms of given size we have to specify what we mean by 'size': We use a general notion of size which covers several previously studied models from the literature. The building blocks of lambda terms, zeros, successors, abstractions and applications, contribute a, b, c and d, respectively, to the total size of a lambda term. Formally, if M and N are lambda terms, then

$$|0| = a, \qquad |Sn| = |n| + b, \qquad |\lambda M| = |M| + c, \qquad |MN| = |M| + |N| + d.$$

Thus for the example given above we have $|\lambda\lambda((S0)0)| = 2a+b+2c+d$. Assigning sizes to the symbols like above covers several previously introduced notions of size:

- so called 'natural counting' (introduced in [2]) where $a = b = c = d = 1$,
- so called 'less natural counting' (introduced in [2]) where $a = 0$, $b = c = 1$, $d = 2$.
- binary lambda calculus (introduced in [35]) where $b = 1$, $a = c = d = 2$,

Assumption 1. Throughout the paper we will impose the following assumptions on the constants a, b, c, d:
1. a, b, c, d are nonnegative integers, **2.** $a + d \geq 1$, **3.** $b, c \geq 1$, **4.** $\gcd(b, c, a + d) = 1$.

If the zeros and the applications both had size 0 (*i.e.* $a + d = 0$), then we would have infinitely many terms of the given size, because one could insert arbitrarily many applications and zeros into a term without increasing its size. If the successors or the abstractions had size 0 (*i.e.* b or c equals to 0), then we would again have infinitely many terms of given size, because one could insert arbitrarily long strings of successors or abstractions into a term without increasing its size. The last assumption is more technical in its nature. It ensures that the generating function associated with the sequence of the number of lambda terms will have exactly one singularity on the circle of convergence, which is on the positive real line. The case of several singularities is not only technically more complicated, but it is for instance not even *a priori* clear which singularities are important and which are negligible. So we cannot expect that it differs from the single singularity case only by a multiplicative constant.

We mention that in [27] lambda terms with size function corresponding to $a = b = 0$ and $c = d = 1$ were considered, but another restriction was imposed on the terms.

Historical remarks

Of course, the enumeration of combinatorial structures or the study of random structures is of interest in its own right. However, there is rising interest in enumeration problems related to structures coming from logic. One of the first works in such a direction were the investigation of random Boolean formulas [30,34,38] and the counting of finite models [39]. The topic was resumed later, studying those random Boolean formulas under different aspects [8,11,20], analogous formulas for other logical models [18,19,21], studying the number of tautologies [29] or the number of proofs in propositional logic [13], comparing logical systems [22], comparing size notions of Boolean trees [14], or applying results in that domain to satisfiability [23]. In [32] combinatorial enumeration of graphs was applied to study satisfiability problem and some generalizations.

To our knowledge first enumerative investigation of lambda terms was performed in [12]. Later, particular classes of lambda terms like linear and affine terms have been enumerated [6,25]. The random generation of terms was for instance treated in [6,27]. Parts of these results have been extended to more general classes of lambda terms [5]. Lambda terms related to combinatory logic were studied in [3,35] where [3] also investigates the question on how many normalizing lambda term there are among all terms. As opposed to the above mentioned results, where lambda terms were interpreted as graphs and their size is then the number of vertices of their graph representation, [35] introduced a size concept related to the De Bruijn representation of lambda terms. The resulting enumeration problem is then different and also simpler when approaching it with generating functions. The reason is there are much fewer terms of a given size than in the graph interpretation, making