How to count linear and affine closed lambda terms?

Pierre Lescanne[*]
University of Lyon, École normale supérieure de Lyon [†]

In this paper we propose a method for counting and generating (including random generation) linear and affine closed λ-terms based on a data structure which we call *SwissCheese* because of its holes. Actually we count those λ-terms up to α-conversion. Therefore it is adequate to use de Bruijn indices [1], because a term with de Bruijn indices represents an α-equivalence class. For instance, generated by a Haskell program, here are 16 linear terms of natural size 8:

$(\lambda 0\,(\lambda 0\,\lambda 0))$ $(\lambda 0\,\lambda(\lambda 0\,0))$ $(\lambda 0\,\lambda(0\,\lambda 0))$ $((\lambda 0\,\lambda 0)\,\lambda 0)$ $(\lambda(\lambda 0\,0)\,\lambda 0)$ $(\lambda(0\,\lambda 0)\,\lambda 0)$ $\lambda(\lambda 0\,(\lambda 0\,0))$ $\lambda(\lambda 0\,(0\,\lambda 0))$

$\lambda((\lambda 0\,\lambda 0)\,0)$ $\lambda(\lambda(\lambda 0\,0)\,0)$ $\lambda(\lambda(0\,\lambda 0)\,0)$ $\lambda(0\,(\lambda 0\,\lambda 0))$ $\lambda(0\,\lambda(\lambda 0\,0))$ $\lambda(0\,\lambda(0\,\lambda 0))$ $\lambda((\lambda 0\,0)\,\lambda 0)$ $\lambda((0\,\lambda 0)\,\lambda 0)$

The Haskell programs of this development are on GitHub: `https://github.com/PierreLescanne/CountingGeneratingAfffineLinearClosedLambdaterms` and a full version of th paper is in [3].

1 Notations

In this paper we use specific notations.

Given a predicate p, the Iverson notation written $[p(x)]$ is the function taking natural values which is 1 if $p(x)$ is true and which is 0 if $p(x)$ is false.

Let $\mathbf{m} \in \mathbb{N}^p$ be the p-tuple $(m_0, ..., m_{p-1})$. Thus $\mathbf{m} \in \mathbb{N}^\omega$ is the sequence $(m_0, m_1, ...)$. Notice in the case of infinite tuples, we are only interested in infinite tuples equal to 0 after some index.

- p is the *length* of $\mathbf{m}$, which we write also $\mathsf{length}\ \mathbf{m}$.
- The p-tuple $(0, ..., 0)$ is written 0^p. 0^ω is the infinite tuple made of 0's.
- The *increment* of a p-tuple at i is:

$$\mathbf{m}^{\uparrow i} = \mathbf{n} \in \mathbb{N}^p \text{ where } n_j = m_j \text{ if } j \neq i \text{ and } n_i = m_i + 1$$

- Putting an element x as *head* of a tuple is written

$$x : \mathbf{m} = x : (m_0, ...) = (x, m_0, ...)$$

  tail removes the head of a tuple:

$$\mathsf{tail}(x : \mathbf{m}) = \mathbf{m}.$$

- $\oplus$ is the componentwise addition on tuples.

[*]pierre.lescanne@ens-lyon.fr
[†]LIP (UMR 5668 CNRS ENS Lyon UCBL INRIA)

To appear in EPTCS.

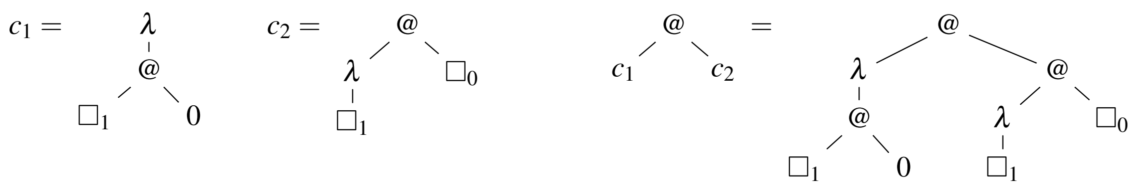

Figure 1: Building a SwissCheese by application

2 SwissCheese

The basic concept is that of **m-SwissCheese** or **Swisscheese of characteristic m** or simply **SwissCheese** if there is no ambiguity on $\mathbf{m}$. A $\mathbf{m}$-SwissCheese or a SwissCheese of characteristic $\mathbf{m}$, where $\mathbf{m}$ is of length p, is a λ-term with holes at p levels, which are all counted, using $\mathbf{m}$. Holes at level i are written $\square_i$. An $\mathbf{m}$-SwissCheese contains holes $\square_0$,... $\square_{p-1}$. A hole $\square_i$ is meant to be a location for a variable at level i, that is under i λ's. Beside holes a SwissCheese contains variables, represented by de Bruijn indices. But all these variables are bound and each binder binds at most one variable, like in closed and affine λ-terms. A Swisscheese can be seen as an affine or a linear closed λ-terms to which holes have been added. According to the way bound variables are inserted when creating abstractions (see below), we consider linear or affine SwissCheeses. The holes have size 0. An $\mathbf{m}$-SwissCheese or a SwissCheese of characteristic $\mathbf{m}$ has m_0 holes at level 0, m_1 holes at level 1, ... m_{p-1} holes at level $p-1$. Let $l_{n,\mathbf{m}}$ (resp. $a_{n,\mathbf{m}}$) count the linear (resp. the affine) $\mathbf{m}$-SwissCheese of size n. $l_{n,\mathbf{m}} = l_{n,\mathbf{m}'}$ and $a_{n,\mathbf{m}} = a_{n,\mathbf{m}'}$ if $\mathbf{m}$ is finite, length $\mathbf{m} \geq n$, $m_i = m'_i$ for $i \leq$ length $\mathbf{m}$, and $m'_i = 0$ for $i >$ length $\mathbf{m}$. $l_{n,0^n}$ (resp. $a_{n,0^n}$) counts the linear closed (resp. the closed affine) λ-terms of size n, since it counts SwissCheeses with no hole.

Growing a SwissCheese

Given two SwissCheeses, we can build a SwissCheese *by application* like in Figure 1. In Figure 1, c_1 is a $(0,1,0,0,0)$-SwissCheese, c_2 is a $(1,1,0,0,0)$-SwissCheese and $c_1@c_2$ is a $(1,2,0,0,0)$-SwissCheese. Said otherwise, c_1 has characteristic $(0,1,0,0,0)$, c_2 has characteristic $(1,1,0,0,0)$ and $c_1@c_2$ has characteristic $(1,2,0,0,0)$. According to what we said, $c_1@c_2$ has characteristic $(1,2)$ as well as characteristic $(1,2,0,0,\ldots)$ (a tuple starting with 1, followed by 2, followed by infinitely many 0's). We could also say that c_1 has characteristic $(0,1)$ and c_2 has characteristic $(1,1)$ making @ a binary operation on SwissCheeses of length 2 whereas previously we have made @ a binary operation on SwissCheeses of length 5. In other words, when counting SwissCheeses of characteristic $\mathbf{m}$, the trailing 0's are irrelevant. In actual computations, we make the lengths of characteristics consistent by adding trailing 0's to too short ones.

Given a SwissCheese, there are two ways to grow a SwissCheese to make another SwissCheese *by abstraction*.

1. We put a λ on the top of a $\mathbf{m}$-SwissCheese c. This increases the levels of the holes: a hole $\square_i$ becomes a hole $\square_{i+1}$. λc is a $(0:\mathbf{m})$-SwissCheese. See Figure 2. This way, no index is bound by the top λ, therefore this does not preserve linearity (it preserves affinity however). Therefore this construction is only for building affine SwissCheeses, not for building linear SwissCheeses. In Figure 2, we colour the added λ in blue and we call it *abstraction with <u>no</u> binding*.

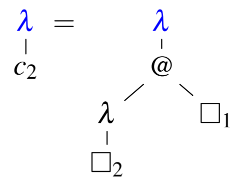

Figure 2: Abstracting a SwissCheese with no binding

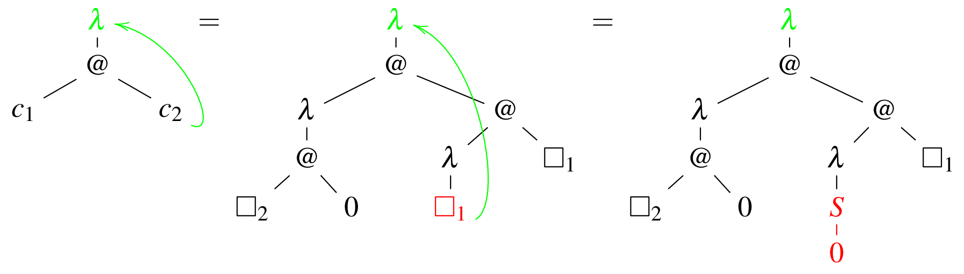

Figure 3: Abstracting a SwissCheese with binding

2. In the second method for growing a SwissCheese by abstraction, we select first a hole $\square_i$, we top the SwissCheese by a λ, we increment the levels of the other holes and we replace the chosen hole by S^i0, i.e., by the de Bruijn index $\underline{i}$. In Figure 3 we colour the added λ in green and we call it *abstraction with binding*.

Measuring SwissCheese

We considers several ways of measuring the size of a SwissCheese derived from what is done on λ-terms. In all these sizes, applications @ and abstractions λ have size 1 and holes have size 0. The differences are in the way variables are measured.

- Variables have size 0, we call this **variable size** 0.
- Variables have size 1, we call this **variable size** 1 .
- Variables (or de Bruijn indices) S^i0 have size $i+1$, we call this **natural size**.

3 Counting linear closed terms

We start with counting linear terms since they are slightly simpler. We will give recursive formulas first for the numbers $l^{\nu}_{n,\mathbf{m}}$ of linear SwissCheeses of natural size n with holes set by $\mathbf{m}$, then for the numbers $l^{0}_{n,\mathbf{m}}$ of linear SwissCheeses of size n, for variable size 0, with holes set by $\mathbf{m}$, eventually for the numbers $l^{1}_{n,\mathbf{m}}$ of linear SwissCheeses of size n, for variable size 1, with holes set by $\mathbf{m}$. When we do not want to specify a chosen size, we write only $l_{n,\mathbf{m}}$ without superscript. This is for specific cases when the value does not depend of the measure of the size.

Natural size

First let us count linear SwissCheeses with natural size. This is given by the coefficient l^v which has two arguments: the size n of the SwissCheese and a tuple $\mathbf{m}$ which specifies the number of holes of each level, i.e, which specifies the characteristics of the SwissCheese. In other words we are interested in the quantity $l^v_{n,\mathbf{m}}$. We assume that the length of $\mathbf{m}$ is p, greater than n.

Size is 0 Whatever size is considered, there is only one SwissCheese of size 0 namely $\square_0$. This means that the number of SwissCheeses of size 0 is 1 if and only if $\mathbf{m} = (1,0,0,...)$:

$$l^v_{0,\mathbf{m}} = l^0_{0,\mathbf{m}} = l^1_{0,\mathbf{m}} = [m_0 = 1 \wedge \bigwedge_{j=1}^{p-1} m_j = 0]$$

Size is $\mathbf{n+1}$ and application If a SwissCheese of size $n+1$ has holes set by $\mathbf{m}$ and is an application, then it is obtained from a SwissCheese of size k with holes set by $\mathbf{q}$ and a SwissCheese of size $n-k$ with holes set by $\mathbf{r}$, with $\mathbf{m} = \mathbf{q} \oplus \mathbf{r}$:

$$\sum_{\mathbf{q}\oplus\mathbf{r}=\mathbf{m}} \sum_{k=0}^{n} l_{k,\mathbf{q}}\, l_{n-k,\mathbf{r}}.$$

Size is $\mathbf{n+1}$ and abstraction with binding Consider a level i, that is a level of hole $\square_i$. If one wants to get a SwissCheese of size $n+1$ by abstraction with binding, first this SwissCheese must be a $0{:}\mathbf{m}$-SwissCheese (there is no hole at level 0 is this SwissCheese), second the SwissCheese in which one chooses the hole $\square_i$ is a $\mathbf{m}^{\uparrow i}$-SwissCheese, since one removes a hole $\square_i$. Therefore, there are m_i+1 ways to choose a hole $\square_i$. In this hole we put a term $S^{i-1}\,0$ of size i. Hence, among $l^v_{n,0:\mathbf{m}}$ SwissCheeses, there are $(m_i+1)\, l^v_{n-i,\mathbf{m}^{\uparrow i}}$ $0{:}\mathbf{m}$-SwissCheeses which are abstractions with binding in which a $\square_i$ has been replaced by the de Bruijn index $S^{i-1}\,0$. Hence by summing over i, the part of abstraction with binding contributes to $l^v_{n+1,0:\mathbf{m}}$ as:

$$\sum_{i=0}^{p-1} (m_i+1)\; l^v_{n-i,\mathbf{m}^{\uparrow i}}.$$

The subtle case of abstraction with binding is pictured in Figure 3. It works as follows. Consider the case where the $(0,1,1)$-SwissCheese $\lambda(\lambda(\square_2\,0)\,((\lambda S0)\,\square_1))$ is obtained from the $(1,2)$-SwissCheese $\lambda(\square_1\,0)\,((\lambda\square_1)\,\square_0)$ of Figure 1 (right) by abstraction with binding. Notice that $(0,1,1) = 0:(1,1)$ and that $(1,2) = (1,1)^{\uparrow 1}$. Focus on level 1 in $\lambda(\square_1\,0)\,((\lambda\square_1)\,\square_0)$. There are 2 holes at level 1, then 2 ways to choose a hole $\square_1$ at level 1, because 2 is the second index of $(1,2)$, which corresponds to level 1. Assume we choose the second hole from the left, the one in red. Put a (green) lambda on the top. Because of this lambda on the top, raise the levels of the other holes (the leftmost one becomes $\square_2$, the rightmost one becomes $\square_1$). Then replace the chosen hole $\square_1$ by $S0$. We get $\lambda(\lambda(\square_2\,0)\,((\lambda S0)\,\square_1))$.

We have the following recursive definitions of $l^v_{n+1,\mathbf{m}}$:

$$\begin{aligned}
l^v_{n+1,0:\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=0:\mathbf{m}} \sum_{k=0}^{n} l^v_{k,\mathbf{q}}\, l^v_{n-k,\mathbf{r}} + \sum_{i=0}^{p-1} (m_i+1)\; l^v_{n-i,\mathbf{m}^{\uparrow i}} \\
l^v_{n+1,(h+1):\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=(h+1):\mathbf{m}} \sum_{k=0}^{n} l^v_{k,\mathbf{q}}\, l^v_{n-k,\mathbf{r}}
\end{aligned}$$

Variable size 0

The only difference is that the inserted de Bruijn index has size 0. Therefore we have $(m_i+1)\, l^0_{n,\mathbf{m}^{\uparrow \mathbf{i}}}$ where we had $(m_i+1)\, l^v_{n-i,\mathbf{m}^{\uparrow \mathbf{i}}}$ for natural size. Hence the formulas:

$$\begin{aligned}
l^0_{n+1,0:\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=0:\mathbf{m}} \sum_{k=0}^{n} l^0_{k,\mathbf{q}}\, l^0_{n-k,\mathbf{r}} + \sum_{i=0}^{p-1} (m_i+1)\; l^0_{n,\mathbf{m}^{\uparrow i}} \\
l^0_{n+1,(h+1):\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=(h+1):\mathbf{m}} \sum_{k=0}^{n} l^0_{k,\mathbf{q}}\, l^0_{n-k,\mathbf{r}}
\end{aligned}$$

The sequence $l^0_{n,0^n}$ of the numbers of linear closed λterms is $0,1,0,5,0,60,0,1105,0,27120,0,828250$, which is sequence A062980 in the *On-line Encyclopedia of Integer Sequences* with 0's at even indices.

Variable size 1

The inserted de Bruijn index has size 1. We have $(m_i+1)\, l^1_{n-1,\mathbf{m}}$ where we had $(m_i+1)\, l^v_{n-i,\mathbf{m}}$ for natural size.

$$\begin{aligned}
l^1_{n+1,0:\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=0:\mathbf{m}} \sum_{k=0}^{n} l^1_{k,\mathbf{q}}\, l^1_{n-k,\mathbf{r}} + \sum_{i=0}^{p-1} (m_i+1)\; l^1_{n-1,\mathbf{m}^{\uparrow i}} \\
l^1_{n+1,(h+1):\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=(h+1):\mathbf{m}} \sum_{k=0}^{n} l^1_{k,\mathbf{q}}\, l^1_{n-k,\mathbf{r}}
\end{aligned}$$

As noticed by Grygiel et al. [2] (§ 6.1), there are no linear closed λ-terms of size $3k$ and $3k+1$. However for the values $3k+2$ we get the sequence: $1,5,60,1105,27120,\ldots$ which is again sequence A062980 of the *On-line Encyclopedia of Integer Sequences*.

4 Counting affine closed terms

We have just to add the case $n \neq 0$ *and abstraction without binding*. Since no index is added, the size increases by 1. The numbers are written $a^v_{n,\mathbf{m}}$, $a^0_{n,\mathbf{m}}$, $a^1_{n,\mathbf{m}}$, and $a_{n,\mathbf{m}}$ when the size does not matter. There are $a_{n,m}$ $(0:\mathbf{m})$-SwissCheeses of size n that are abstractions with no binding. We get the recursive formulas:

Natural size

$$\begin{aligned}
a^v_{n+1,0:\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=0:\mathbf{m}} \sum_{k=0}^{n} a^v_{k,\mathbf{q}}\, a^v_{n-k,\mathbf{r}} + \sum_{i=0}^{p-1} (m_i+1)\; a^v_{n-i,\mathbf{m}^{\uparrow i}} + a^v_{n,\mathbf{m}} \\
a^v_{n+1,(h+1):\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=(h+1):\mathbf{m}} \sum_{k=0}^{n} a^v_{k,\mathbf{q}}\, a^v_{n-k,\mathbf{r}}
\end{aligned}$$

Variable size 0

$$\begin{aligned}
a^0_{n+1,0:\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=0:\mathbf{m}} \sum_{k=0}^{n} a^0_{k,\mathbf{q}}\, a^0_{n-k,\mathbf{r}} + \sum_{i=0}^{p-1} (m_i+1)\; a^0_{n,\mathbf{m}\uparrow i} + a^0_{n,\mathbf{m}} \\
a^0_{n+1,(h+1):\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=(h+1):\mathbf{m}} \sum_{k=0}^{n} a^0_{k,\mathbf{q}}\, a^0_{n-k,\mathbf{r}}
\end{aligned}$$

The sequence is A287045 in the *On-line Encyclopedia of Integer Sequences*. It corresponds to the coefficients of the generating function $\mathscr{A}(z,0)$ where

$$\mathscr{A}(z,u) = u + z(\mathscr{A}(z,u))^2 + z\frac{\partial \mathscr{A}(z,u)}{\partial u} + z\mathscr{A}(z,u).$$

Variable size 1

$$\begin{aligned}
a^1_{n+1,0:\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=0:\mathbf{m}} \sum_{k=0}^{n-1} a^1_{k,\mathbf{q}}\, a^1_{n-k,\mathbf{r}} + \sum_{i=0}^{p-1} (m_i+1)\; a^1_{n-1,\mathbf{m}\uparrow i} + a^1_{n,\mathbf{m}} \\
a^1_{n+1,(h+1):\mathbf{m}} &= \sum_{\mathbf{q}\oplus\mathbf{r}=(h+1):\mathbf{m}} \sum_{k=0}^{n} a^1_{k,\mathbf{q}}\, a^1_{n-k,\mathbf{r}}
\end{aligned}$$

The sequence is A281270 in the *On-line Encyclopedia of Integer Sequences*. It corresponds to the coefficient of the generating function $\hat{\mathscr{A}}(z,0)$ where $\hat{\mathscr{A}}(z,u)$ is the solution of the functional equation:

$$\hat{\mathscr{A}}(z,u) = zu + z(\hat{\mathscr{A}}(z,u))^2 + z\frac{\partial \hat{\mathscr{A}}(z,u)}{\partial u} + z\hat{\mathscr{A}}(z,u).$$

References

[1] Nicolaas G. de Bruijn. Lambda calculus with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. *Proc. Koninkl. Nederl. Akademie van Wetenschappen*, 75(5):381–392, 1972.

[2] Katarzyna Grygiel, Pawel M. Idziak, and Marek Zaionc. How big is BCI fragment of BCK logic. *J. Log. Comput.*, 23(3):673–691, 2013.

[3] Pierre Lescanne. Quantitative aspects of linear and affine closed lambda terms. *ArXiv*, abs/1702.03085, 2017.