Formalizing Moessner's Theorem and Generalizations in NUPRL

Mark Bickford[a], Dexter Kozen[a], Alexandra Silva[b]

[a] *Computer Science Department, Cornell University, Ithaca, NY, USA*
[b] *Computer Science Department, University College London, London, United Kingdom*

Abstract

Moessner's theorem describes a procedure for generating a sequence of n integer sequences that lead unexpectedly to the sequence of nth powers 1^n, 2^n, 3^n, ... Several generalizations of Moessner's theorem exist. Recently, Kozen and Silva gave an algebraic proof of a general theorem that subsumes Moessner's original theorem and its known generalizations. In this note, we describe the formalization of this theorem that the first author did in NUPRL. On the one hand, the formalization remains remarkably close to the original proof. On the other hand, it leads to new insights in the proof, pointing to small gaps and ambiguities that would never raise any objections in *pen and pencil* proofs, but which must be resolved in machine formalization.

Keywords: Moessner's theorem, Pascal triangle, formalization, Nuprl

This short paper was first written in 2011-2012 and presented at an informal workshop in Braga, Portugal. Prof. Valença was in the audience and was fascinated by the simplicity and beauty of both the proof and formalisation. The last author was introduced to logic, functional programming, and formal methods in the undergraduate classes she took from Prof. Valença, who has always shown enthusiasm in explaining difficult concepts in a clear and simple way. We are happy to dedicate this paper to Prof. Valença on the occasion of his 70th birthday and wish him many happy returns.

Email addresses: markb@cs.cornell.edu (Mark Bickford), kozen@cs.cornell.edu (Dexter Kozen), alexandra.silva@ucl.ac.uk (Alexandra Silva)

Work carried out during a research visit to Cornell University.

1. Introduction

Proof assistants or interactive theorem provers are software tools used for formalizing properties and proofs of those properties. The holy grail of mechanized theorem proving is to give formalized, machine-checked mathematical proofs that are as close to the human written proofs as possible.

The NUPRL[2] proof development system is based on a formal account of deduction. Proofs are the main characters and are used not only to establish truth but also to denote evidence, including computational evidence in the form of programs (functional and distributed). The idea of a proof term is a key abstraction; it is a meaningful mathematical expression denoting evidence for truth.

In this note, we describe the formalization in NUPRL of the recent proof of Kozen and Silva [5] of Moessner's theorem and its generalizations. The process of formalization uncovered several idiosyncrasies that forced us to be much more precise in several points than we had been in the paper proof. The main challenge was to build all the needed mathematical background. Once this was in place, the formal statements and their proofs turned out to be relatively straightforward and close to the originals.

We will omit most of the details of the proofs, referring the reader to [5] for further details. Instead, we will walk the reader through the steps that had to be taken to obtain a full formalization and we will point out the unexpected differences with the proof in [5] and the challenges faced during the formalization. But first, let us describe the problem we will formalize.

Consider the following procedure for generating $n \geq 1$ infinite sequences of positive integers. To generate the first sequence, write down the positive integers $1, 2, 3, \ldots$, then cross out every nth element. For the second sequence, compute the prefix sums of the first sequence, ignoring the crossed-out elements, then cross out every $(n-1)$st element. For the third sequence, compute the prefix sums of the second sequence, then cross out every $(n-2)$nd element, and so on. For example, for $n = 4$,

1	2	3	~~4~~	5	6	7	~~8~~	9	10	11	~~12~~	13	14	15	~~16~~	17	18	19	~~20~~	21	22	23	~~24~~
1	3	~~6~~		11	17	~~24~~		33	43	~~54~~		67	81	~~96~~		113	131	~~150~~		171	193	~~216~~	
1	~~4~~			15	~~32~~			65	~~108~~			175	~~256~~			369	~~500~~			671	~~864~~		
1				16				81				256				625				1296			

[2]Pronounced "new pearl"

Moessner's theorem says that the final sequence is $1^n,\ 2^n,\ 3^n,\ \ldots$.

This construction is an interesting combinatorial curiosity that has attracted much attention over the years. The theorem was never proved by its eponymous discoverer [9]. The first proof was given by Perron [12]. The theorem has been the subject of several popular accounts [1, 3, 8].

In the construction of Moessner's theorem, the initial step size n is constant. What happens if we increase it in each step? Let us repeat the construction starting with a step size of one and increasing the step size by one each time. Thus, in the first sequence, we cross out 1, 3, 6, 10, ..., $\binom{k+1}{2}$, ...

```
1   2   ~~3~~   4   5   ~~6~~   7   8   9   ~~10~~   11   12   13   14   ~~15~~   16   17   18   19   20   ~~21~~
    2         6   ~~11~~        18  26  ~~35~~        46   58   71   ~~85~~       101  118  136  155  ~~175~~
              6                 24  ~~50~~           96  154  ~~225~~           326  444  580  ~~735~~
                                24                   120  ~~274~~              600104~~41624~~
                                                     120                       720~~1764~~
                                                                               720
```

Now the final sequence consists of the factorials 1, 2, 6, 24, 120, ... = 1!, 2!, 3!, 4!, 5!, ...

Let us now *increment the increment* by one in each step, thus incrementing the step size by 1, 2, 3, 4, ... in successive steps, crossing out 1, 4, 10, 20, ..., $\binom{k+2}{3}$, ...

```
1   2   3   ~~4~~   5   6   7   8   9   ~~10~~   11   12   13   14   15   16   17   18   19   ~~20~~ ...
    2   ~~5~~       10  16  23  31  ~~40~~        51   63   76   80   95 ...
    2               12  28  51  ~~82~~           133  196  272  352 ...
                    12  40  ~~91~~               224  420  692 ...
                    12  ~~52~~                   276  696 ...
                    12                          288  984 ...
                                                2881272 ...
                                                288 ...
                                                288 ...
                                                288 ...
```

The final sequence consists of the *superfactorials*

$$1,\ 2,\ 12,\ 288,\ \ldots = 1!,\ 2!1!,\ 3!2!1!,\ 4!3!2!1!,\ \ldots = 1!!,\ 2!!,\ 3!!,\ 4!!,\ \ldots\ .$$

The generalization of Moessner's theorem that handles these cases is known as *Paasche's theorem* [11].

Long [7, 8] discovered the following alternative procedure and generalization. Consider the figure illustrating the Moessner construction for $n = 4$ above. Breaking the figure into separate triangles and adding a row of 1's at the top, the first four triangles are

```
1   1   1   1   1        1   1   1   1   1        1   1   1   1   1        1   1   1   1   1
1   2   3   4            5   6   7   8            9  10  11  12           13  14  15  16
1   3   6               11  17  24                33  43  54               67  81  96
1   4                   15  32                    65 108                  175 256
1                       16                        81                      256
```

Call these the *level-n Moessner triangles*. The first triangle is the Pascal triangle. However, note that all the triangles satisfy the *Pascal property*: each interior element is the sum of the elements immediately above it and to its left.

Instead of a sequence of sequences of integers, Long described how to generate a single sequence of triangles. To generate the $(k+1)$st triangle from the kth, consider the nth northeast-to-southwest row. (For the second triangle above, this would be 1 8 24 32 16.) Let the first column of the next triangle be the prefix sums of this sequence (in our example, 1 9 33 65 81), and let the first row be a sequence of 1's. Complete the triangle using the Pascal property. Note that Long's construction of the triangles above is really a reformulation of Moessner's result, since from the triangles one can indeed read off horizontally the (construction of the) sequence of powers again.

Long [7, 8] also generalized Moessner's result to apply to the situation in which the first sequence is not the sequence of successive integers $1, 2, 3, \ldots$ but the arithmetic progression $a, a+d, a+2d, \ldots$. This corresponds to a sequence of triangles with $d, d, d, \ldots$ along the top and $d, a, a, a, \ldots$ as the first column of the first triangle. They showed that the final sequence obtained by the Moessner construction is $a \cdot 1^{n-1}, (a+d) \cdot 2^{n-1}, (a+2d) \cdot 3^{n-1}, \ldots$.

More recently, Hinze [2] and Niqui and Rutten [10] have given proofs involving concepts from functional programming, Hinze using calculational scans and Niqui and Rutten using coalgebra of streams. The proof of Hinze covers Moessner's and Paasche's result whereas Rutten and Niqui only provide a proof of the original Moessner theorem. Rutten and Niqui's proof has been formalized in Coq [6]. The Coq formalisation in [6] is fundamentally different than what we will see in this paper, as the proofs use very different techniques. It is a good illustration of the basic coinduction principles built in Coq while highlighting at the same time its limitations in terms of more expressive extensions (such as coinduction up-to or any coinductive hypothesis that does not strictly fall into the guardedness requirement) that are needed in more advanced coinductive proofs.

The proof Kozen and Silva presented in [5], and hence the formalization presented in the present paper, has the advantage of covering all the theorems

mentioned above and, furthermore, opening the door to new generalizations of Moessner's original result.

In the following, all the NUPRL code snippets (displayed in blue) are hyperlinks to complete proofs in the NUPRL library. The whole formalisation, which contains additional libraries for formal power series, can be accessed from `http://www.nuprl.org/wip/Standard2/power!series/`. We estimate the the whole project was a one man-month of effort.

2. Algebraic Representation: Formal Power Series

Kozen and Silva described Long's construction in terms of multidimensional generating functions, also known as formal power series in multiple variables. In this section, we will revisit all the steps of the proof, starting with setting basic definitions, building up to the main theorem (Theorem 2.4), which will have as corollaries Moessner's (Corollary 2.5), and Paasche's (Corollary 2.6) results. We will explain the very few details where the formalisation differs from the original pen-and-paper proof, but the goal of this section is to also highlight how NUPRL proofs stay very close to the original proof and enable the detection of minor omissions in the mathematical presentation.

The first step in the NUPRL formalization was to formalize the theory of formal power series. In NUPRL, there were already formalizations of basic algebraic structures, such as monoids, groups and rings (described in Paul Jackson's thesis [4]) and also of multisets (or bags). This made it possibly to incrementally build the theory of formal power series: a formal power series is represented as a map between monomials and coefficients taken from a ring. A monomial, in turn, is just a multiset of variables. The operations on formal power series, such as sum and convolution product, were then simply defined reusing existing operations on multisets. The details of this formalization are not important for understanding the rest of the paper and will therefore be omitted.

The triangles were represented as elements of the rational function field $\mathbb{Z}(x, y)$. For example, the *Pascal triangle* $\Delta = \Delta(x, y)$ is

$$\Delta(x, y) = \frac{1}{1 - (x + y)}. \tag{1}$$

In NUPRL, this is represented concisely by the statement:

```
Δ(x,y)  ==  (1÷•(1-(<{x}>+<{y}>)))
```

Note the close similarity to the formula (1) above.

To describe the operation of "completing the triangle using the Pascal property," we need some notation and lemmas. A power series in x, y is called *Pascal* if the coefficient of every interior monomial m (a monomial of positive degree in both x and y) is the sum of the coefficients of m/x and m/y. The following lemma from [5] characterizes the Pascal property of a power series:

Lemma 2.1 ([5]). *$f = f(x, y)$ is Pascal iff*

$$f = ((1 - x)f(x, 0) + (1 - y)f(0, y) - f(0, 0)) \cdot \Delta.$$

In NUPRL the formal statement of this lemma is very close to the original formulation:

```
∀[r:CRng]. ∀[x,y:Atom]. ∀[f:PowerSeries(r)].
   fps-Pascal(r;x;y;f)
   ⟺  f = (((((1-y)*f(x:=0))
                  +((1-x)*f(y:=0)))-f(x:=0)(y:=0))*Δ(x,y))
   supposing ¬(x = y)
```

Here we see how the formalization forces us to be more precise: when one writes x and y for the variable names, one is implicitly assuming that they are different. However, in the formalization this needs to be said explicitly. The same applies to quantification: in the formulation of Lemma 2.1 above, the symbol f refers to any formal power series, which is made precise in the $\forall$ quantification of the NUPRL code. Furthermore, note that we need not restrict ourselves to the ring of integers as in the original proof, but instead can generalize to any commutative ring, which we denote by `r:CRng`.

Before describing how the successive triangles are constructed, one more result is needed: that any two given series $g \in \mathbb{Z}(x)$ and $f \in \mathbb{Z}(y)$ with the same constant coefficient can be extended uniquely to a $p \in \mathbb{Z}(x, y)$ satisfying the Pascal property.

Lemma 2.2 ([5]). *If $g \in \mathbb{Z}(x)$, $f \in \mathbb{Z}(y)$, and $g(0) = f(0)$, then*

$$p = ((1 - x)g + (1 - y)f - f(0)) \cdot \Delta \tag{2}$$

is the unique $p \in \mathbb{Z}(x, y)$ such that (i) $p(x, 0) = g(x)$, (ii) $p(0, y) = f(y)$, *and* (iii) p *is Pascal.*

To formalize this, we first define the equation (2) as

```
Pascal-completion(r;f;g;x;y)
                    ==  ((1-y)*f)+(1-x)*g)-f(y:=0))*Δ(x,y)
```

In the original proof, a parameter p_0 was used in place of $f(0) = g(0)$ so that the statement of the theorem would be symmetric in g and f. Here we have used $f(0)$ (in NUPRL, `f(y:=0)`) instead. This breaks the symmetry but avoids having to define the extra parameter p_0. The term `f(y:=0)` is the power series one gets by setting all the `y` variables to 0.

We note that the use of variable `r` in the above definition, though at first sight unbounded, provides the type of the basic ring operations used in the definition, e.g. $+$. If one inspects the NUPRL code of the above equation, in the section **Definitions occuring in Statement** one will find that one of the functions used is e.g. `fps_add (f + g)` and inspecting this reveals:

```
(f+g) ==   λb.(+r f[b] g[b])
```

which explicitly states that power series addition uses the $+$ from the ring `r`, pointwise. The display form `(f + g)`, used in `Pascal-completion(r;f;g;x;y)`, does not show the parameter r, but it is really there.

Next, to finish the formalization of Lemma 2.2, we need to state and prove the uniqueness of p, which we are here denoting by `Pascal-completion(r;f;g;x;y)`:

```
∀[r:CRng]. ∀[f,g:PowerSeries(r)]. ∀[x,y:Atom].
    ((Pascal-completion(r;f;g;x;y)(x:=0) = f)
    ∧ (Pascal-completion(r;f;g;x;y)(y:=0) = g)
    ∧ fps-Pascal(r;x;y;Pascal-completion(r;f;g;x;y)))
    ∧ (∀h:PowerSeries(r)
              (fps-Pascal(r;x;y;h)
              ⇒ (h(x:=0) = f)
              ⇒ (h(y:=0) = g)
              ⇒ (h = Pascal-completion(r;f;g;x;y))))
    supposing (¬(1 = 0)) ∧ (¬(x = y)) ∧
              (f(x:=0) = f) ∧ (g(y:=0) = g) ∧ (f(y:=0) =
g(x:=0))
```

The first three conjuncts correspond to (i), (ii) and (iii) in Lemma 2.2 and the last conjunct to the statement of uniqueness.

Some remarks are in order for the code after the `supposing` clause. First, in the formalization above, we have not actually restricted f and g to be power series in one variable. Instead, we take any series in any number of

variables, then require that f and g satisfy the condition `(f(x:=0) = f)` $\wedge$ `(g(y:=0) = g)`, which is enough for the proof. This is a generalization of the original Lemma 2.2 above. Moreover, in the process of doing the proof, we discovered that the statement does not hold for the trivial ring, thus we need the additional assumption `(¬(1 = 0))`. The original proof was done in the ring of integers, in which this statement holds trivially.

Each successive level-n Moessner triangle is obtained from the previous by taking the homogeneous component of degree n, evaluating at $y = 1$, and multiplying by Δ. In other words, if we define inductively

$$h_0(x, y) = 1 \qquad\qquad h_{k+1}(x, y) = [h_k(x, 1) \cdot \Delta(x, y)]_n \,, \tag{3}$$

then the kth level-n Moessner triangle is $h_k(x, 1) \cdot \Delta$ and the final sequence in the Moessner construction is the lead coefficient of $h_k(x, 1)$ for $k = 1, 2, 3, \ldots$.

We first need to define the operation of *taking the homogeneous component of degree n*. Because of the formalization of power series using bags of monomials, this is a rather simple operation:

```
[f]_n  ==  λb.if (bag-size(b) =z n) then f b else 0 fi
```

Next, we formalize the operation of *evaluating at* $y = 1$:

```
[f]_n(y:=1)  ==
   λb.if 0 <z (#y in b) ∨b n <z bag-size(b)
      then 0
      else f[b + bag-rep(n - bag-size(b);y)] fi
```

We are now ready to state and formalize the main lemma of the paper.

Lemma 2.3 ([5]). *Let $h(x, y)$ be homogeneous of degree n and let $d \geq 0$. Then*

$$[h(x, 1) \cdot \Delta(x, y)]_{n+d} = (x + y)^d h(x, x + y).$$

In NUPRL, this was formalized as:

```
∀[r:CRng]. ∀[x,y:Atom]. ∀[h:PowerSeries(r)]. ∀[n,m:ℕ].
    [([h]_n(y:=1)*Δ(x,y))]_m  =  ([h]_n(y:=(x+y))*((x+y))^(m
- n))
     supposing (n ≤ m) ∧ (¬(x = y))
```

The main difference between this formalization and Lemma 2.3 is that we do not assume $h(x, y)$ to be homogeneous of degree n, but instead take its homogeneous component of degree n: `[h]_n`. Variable m in the formalization above corresponds to $n + d$ in Lemma 2.3.

This proof is a bit different from that of [5]. Instead, we proved a lemma that said that any two linear, uniformly continuous functions on power series that agree on monomials must agree on all power series. This allows the proof of the main lemma to be reduced to the easier case of monomials. One must also prove that all the operations on power series used in the lemma are uniformly continuous, but this is true of every function definable in constructive logic.

Next, we present the main theorem of the paper, which has as corollaries Moessner's theorem and the several generalizations mentioned in the introduction.

Theorem 2.4 ([5])**.** *Let h_k be the sequence defined by (3). For all $k \geq 0$,*

$$h_k(x, y) = \prod_{i=0}^{k-1} ((k - i)x + y)^{d(i)} \cdot h_0(x, kx + y).$$

The formalization in NUPRL is:

```
∀[r:CRng]. ∀[x,y:Atom].
   ∀[h:PowerSeries(r)]. ∀[d:ℕ ⟶ ℕ]. ∀[k:ℕ].
      Moessner(r;x;y;h;d;k) = ([h]_d0(y:=((k · r 1)*x + y))
         * Π(i ∈ upto(k)).((((k - i) · r 1)*x + y))^(d (i +
1)))
         supposing ¬(x = y)
```

Paasche's, Long's, and Moessner's theorems are now immediate consequences of Theorem 2.4. Paasche's second theorem on the superfactorials and Long's theorem are omitted here, but the construction is similar, and they are available in the NUPRL library.

Corollary 2.5 (Moessner's Theorem)**.** *If $h_0 = 1$, $d(0) = n$, and $d(k) = 0$ for $k \geq 1$, then the lead coefficient of $h_k(x, 1)$ is k^n for all $k \geq 1$.*

The formalization in NUPRL is:

```
∀[x,y:Atom].
   ∀[n:ℕ]. ∀[k:ℕ+].
      (Moessner(ℤ-rng;x;y;1;λi.if (i =z 0) then 0
                     if (i =z 1) then n else 0 fi
;k)[bag-rep(n;x)]
          = k^n)
      supposing ¬(x = y)
```

Corollary 2.6 (Paasche's Theorem)**.** *For $h_0 = 1$ and any sequence d, the lead coefficient of $h_k(x, 1)$ is*

$$\prod_{i=0}^{k-1}(k-i)^{d(i)}$$

for all $k \geq 0$. In particular, the sequences $d = 1, 1, 1, \ldots$ and $d = 1, 2, 3, \ldots$ yield the factorials and superfactorials, respectively.

The formalization in NUPRL is:

```
∀[x,y:Atom].
   ∀[d:ℕ ⟶ ℕ]. ∀[k:ℕ].
      (Moessner(ℤ-rng;x;y;1;λi.if (i =z 0) then 0
               else d (i - 1) fi ;k)[bag-rep(Σ(d i | i <
k);x)]
          = Π(k - i^d i | i < k))
      supposing ¬(x = y)

∀[x,y:Atom]. ∀[k:ℕ].
   (Moessner(ℤ-rng;x;y;1;λi.if (i =z 0) then 0
               else 1 fi ;k)[bag-rep(k;x)] = (k)!)
   supposing ¬(x = y)
```

3. Conclusion

We have described a machine formalization of the algebraic proof presented in [5] of Moessner's theorem and related theorems in NUPRL. Although the machine proof closely models the paper and pencil proof of [5], the very process of formalization reveals several implicit assumptions and highlights the need for engineering decisions that would not otherwise be apparent.

Bibliography

[1] J. H. Conway and R. K. Guy. Moessner's magic. In *The Book of Numbers*, pages 63–65. Springer-Verlag, 1996.

[2] R. Hinze. Scans and convolutions—a calculational proof of Moessner's theorem. In Sven-Bodo Scholz, editor, *Post-proceedings of the 20th International Symposium on the Implementation and Application of Functional Languages (IFL '08)*, volume 5836 of *Lecture Notes in Computer Science*. Springer-Verlag, 2009.

[3] R. Honsberger. *More Mathematical Morsels*. Math. Assoc. Amer., 1991.

[4] Paul Bernard Jackson. *Enhancing the* NUPRL *proof development system and applying it to computational abstract algebra*. PhD thesis, Cornell University, 1995.

[5] Dexter Kozen and Alexandra Silva. On Moessner's theorem. *Amer. Math. Monthly*, 120(2):131–139, February 2013.

[6] Robbert Krebbers, Louis Parlant, and Alexandra Silva. Moessner's theorem: an exercise in coinductive reasoning in Coq. Code available from https://github.com/robbertkrebbers/moessner.

[7] C. T. Long. On the Moessner theorem on integral powers. *Amer. Math. Monthly*, 73(8):846–851, October 1966.

[8] C. T. Long. Strike it out–add it up. *The Mathematical Gazette*, 66(438):273–277, December 1982.

[9] A. Moessner. Eine Bemerkung über die Potenzen der natürlichen Zahlen. *Sitzungsberichten der Bayerischen Akademie der Wissenschaften, Mathematischnaturwissenschaftliche Klasse 1952*, 29, March 1951.

[10] M. Niqui and J. Rutten. An exercise in coinduction: Moessner's theorem. Technical Report SEN-1103, CWI Amsterdam, 2011.

[11] I. Paasche. Eine Verallgemeinerung des moessnerschen Satzes. *Compositio Mathematica*, 12:263–270, 1954.

[12] O. Perron. Beweis des Moessnerschen Satzes. *Sitzungsberichten der Bayerischen Akademie der Wissenschaften, Mathematisch-naturwissenschaftliche Klasse 1951*, 4:31–34, May 1951.