An algorithm for proving identities with Riordan transformations
Extended abstract

Donatella Merlini and Renzo Sprugnoli
Dipartimento di Sistemi e Informatica
viale Morgagni 65, 50134, Firenze, Italia
merlini@dsi.unifi.it, renzo.sprugnoli@unifi.it

September 23, 2009

The problem we consider in the present paper is how to find the closed form of a class of combinatorial sums, if it exists. The problem is well known in the literature, and is as old as Combinatorial Analysis is, since we can go back at least to Euler's time. More recently, Riordan has tried to give a general approach to the subject, proposing a variety of methods, many of which are related to generating functions. Knuth [5] and Graham et al. [3] have shown that a clever use of properties of binomial coefficients and of other combinatorial quantities can produce beautiful proofs of combinatorial identities. Generating functions have been used by Wilf [12] in his "snake oil" method; Petkovšek, Wilf and Zeilberger [7] were able to find a general algorithm to treat sums of hypergeometric terms. The algorithm is mainly devised for computer usage and is based on previous works by Gosper [2] and Sister Celine, whose method is presented in Rainville [8, Chapter 14]. In the same line, Kauers [4] has proposed a method dealing also with Stirling numbers of both kinds. By using the theory of Riordan arrays, introduced by Shapiro et al. [9], Sprugnoli [10] developed a method consisting of transformations of generating functions, induced by a Riordan array.

A *proper Riordan array* is a pair of formal power series $D = (d(t), h(t))$, where $d(0) \neq 0$, $h(0) = 0$ and $h'(0) \neq 0$. If we maintain the only condition $h(0) = 0$ we obtain the general concept of a Riordan array; whenever possible, we will deal with proper Riordan arrays. The (proper) Riordan array D defines an infinite, lower triangular array:

$$d_{n,k} = [t^n]d(t)h(t)^k;$$

in other words, $d(t)h(t)^k$ is the generating function of column k of the Riordan array; column 0 has $d(t)$ as its generating function. Riordan arrays enjoy a lot of properties, but what relates them to combinatorial sums is the identity:

$$\sum_{k=0}^{n} d_{n,k} f_k = [t^n]d(t)f(h(t)), \tag{1}$$

which allows us to compute a combinatorial sum by performing a suitable transformation of the generating function $f(t)$ and then a coefficient extraction. The method we present in this paper can be formalized with an algorithm which returns a recurrence relation for the sequence $g_n = \sum_{k=0}^{n} d_{n,k} f_k$, similarly to what happens with the Zeilberger algorithm described in [7]. If the recurrence relation can be solved, then we have a constructive proof of a combinatorial identity. Otherwise, the recurrence can be used to verify that the identity holds or, also, to find an asymptotic approximation of the sum (see, e.g., Wimp e Zeilberger [13]).

The following general transformations related to binomial coefficients hold:

$$\sum_k \binom{n+ak}{m+bk} f_k = [t^n]\frac{t^m}{(1-t)^{m+1}} f\left(\frac{t^{b-a}}{(1-t)^b}\right) \qquad b > a \qquad (A)$$

$$\sum_k \binom{n+ak}{m+bk} f_k = [t^m](1+t)^n f(t^{-b}(1+t)^a) \qquad b < 0 \qquad (B).$$

Euler's or binomial transformation corresponds to case (A) with $m = 0,\ a = 0$ and $b = 1$.

We can summarize the aim of the present paper in the following way. It is not always possible to find the explicit generating function for the sequence f_k or also it may be difficult to extract the coefficient of t^n from the formal power series $d(t)f(h(t))$ (although in this latter situation one can possibly find an asymptotic formula for the sum). In these cases, the direct application of the Riordan array method can fail, and we must look for other approaches. Luckily, there are several occasions in which the Riordan array transformations can be applied without knowing explicitly the form of the involved generating functions. This fact enriches the theory of Riordan arrays in the context of constructive evaluation of combinatorial sums.

In order to be concrete, let us start with a combinatorial sum of the form $\sum_{k=0}^{n} d_{n,k} f_k$, where $(d_{n,k})_{n,k\in\mathbb{N}}$ is the Riordan array $(d(t),\, h(t))$ and $f(t)$ is the (generally unknown) generating function of the sequence $(f_k)_{k\in\mathbb{N}}$. We suppose that (f_k) satisfies a linear recurrence relation with non-constant polynomial coefficients:

$$a_j(n)f_{n+j} + a_{j-1}(n)f_{n+j-1} + \cdots + a_0(n)f_n = b(n) \qquad \forall n \in \mathbb{N}; \tag{2}$$

it is known that this is equivalent to a suitable differential equation in $f(t)$ and, on the contrary, given a linear differential equation in $f(t)$, it corresponds to a linear recurrence for the sequence $(f_k)_{k\in\mathbb{N}}$.

The passage from a recurrence relation to the corresponding differential equation (and vice versa) requires the application of the *principle of identity* for generating functions: $\mathcal{G}(f_k) = \mathcal{G}(g_k)$ if and only if $f_k = g_k\ \forall k \in \mathbb{N}$; therefore it is mandatory that the recurrence relation holds for every natural number, 0 included. As a rule of thumb, it is more convenient to express the recurrence relation for indices $k,\ k+1,\ \ldots,\ k+j$ rather then for $k,\ k-1,\ \ldots,\ k-j$.

If $f(t)$ is the generating function of a sequence $(f_k)_{k\in\mathbb{N}}$, the *shift rule* (see, e.g., Merlini et al. [6]) for the generating function operator can be easily generalized to:

$$\mathcal{G}(f_{k+j}) = \frac{f(t) - f_0 - f_1 t - \cdots - f_{j-1}t^{j-1}}{t^j}.$$

The generalized shift rule and the following formulas are the bases for passing from a linear recurrence to the corresponding differential equation and, vice versa, for returning from the differential equation to the recurrence relation. For the *coefficient of* operator the following properties hold:

$$[t^k]f(t) = f_k; \qquad [t^k]f'(t) = (k+1)f_{k+1};$$

$$[t^k]f''(t) = (k+2)(k+1)f_{k+2}; \qquad [t^k]f'''(t) = (k+3)(k+2)(k+1)f_{k+3}.$$

$\mathcal{G}(f_k) = f(t)$	$\mathcal{G}(f_{k+1}) = \dfrac{f(t) - f_0}{t}$	$\mathcal{G}(f_{k+2}) = \dfrac{f(t) - f_0 - f_1}{t^2}$
$\mathcal{G}(kf_k) = tf'(t)$	$\mathcal{G}(kf_{k+1}) = f'(t) - \mathcal{G}(f_{k+1})$	$\mathcal{G}(kf_{k+2}) = \dfrac{f'(t) - f_1}{t} - 2\mathcal{G}(f_{k+2})$
$\mathcal{G}(k^2 f_k) = t^2 f''(t) + tf'(t)$	$\mathcal{G}(k^2 f_{k+1}) = t^2 f''(t) - \mathcal{G}(kf_{k+1})$	$\mathcal{G}(k^2 f_{k+2}) = f''(t) - 3\dfrac{f'(t) - f_1}{t} + 4\mathcal{G}(f_{k+2})$

Table 1: Some rules for the operator $\mathcal{G}$

For practical purposes, it is convenient to have some explicit formulas for the operator $\mathcal{G}$. We list them in Table 1 and an algorithm is easily devised for computing $\mathcal{G}(k^s f_{k+p})$ when s, p are given.

Now, let us come to the central point of the method, that is how to apply the Riordan transformation to a differential equation. This is simply done by transforming the generating function and its derivatives. The general method can be formalized with the algorithm illustrated in Table 2.

INPUT A linear recurrence relation (2) for the sequence $(f_k)_{k\in\mathbb{N}}$ and a Riordan array $(d_{n,k})_{n,k\in\mathbb{N}} = (d(t), h(t))$ for the transformation defined by $g(t) = d(t)f(h(t))$, where $f_k = [t^k]f(t)$.
OUTPUT A linear recurrence relation for the sequence $(g_n)_{n\in\mathbb{N}}$, where $g_n = [t^n]g(t)$.

1. Apply the *generating function* operator $\mathcal{G}$ to the recurrence relation (2), obtaining a (differential) equation in $f(t)$ of order J.
2. Apply the transformation $(d(t), h(t))$:
   (a) Apply the substitution $t \to h(t)$ in the differential equation.
   (b) Apply the substitutions for $f^{(j)}(h(t))$ by using the relations:
$$\frac{d^j}{dt^j}f(h(t)) = \frac{d^j}{dt^j}\frac{g(t)}{d(t)}, \qquad j = 0, \cdots, J.$$
3. Apply the *coefficient of* operator $[t^n]$ to the (differential) equation obtained in step 2 obtaining a recurrence relation for $(g_n)_{n\in\mathbb{N}}$.
4. Compute the necessary initial conditions by using the relation $g_n = \sum_{k\geq 0} d_{n,k}f_k$.

Table 2: The algorithm for computing a recurrence relation for $g_n = \sum_{k\geq 0} d_{n,k}f_k$.

Although Riordan arrays are a general methodology for computing combinatorial sums, in order to define properly the algorithm, it is necessary to restrict the class of formal power series involved in the transformations. In general we assume that: 1) $d(t), f(t)$ are holonomic functions, i.e., they satisfy a linear differential equation with polynomial coefficients; 2) $h(t)$ is an algebraic function, i.e., $P(h(t), t) = 0$ for some bivariate polynomial $P(x, y)$. Under these conditions, by [11, Th. 6.4.9 and Th. 6.4.10], the transformation $d(t)f(h(t))$ is holonomic. The algorithm requires a number of steps which grows polynomially with the order of the differential equation satisfied by the generating function of the sequence $(f_k)_{k\in\mathbb{N}}$.

As an example, let us consider the Chu and Yan's identity in [1, Th. 1]:
$$\sum_{k=0}^{n}\binom{n}{k}\frac{(-1)^k}{(x+k)_{m+1}} = \frac{(m+n)!}{m!(x)_{m+n+1}}.$$
The recurrence relation to which apply the algorithm is as follows:
$$f_k = \frac{(-1)^k}{(x+k)_{m+1}} \qquad \text{or} \qquad f_{k+1} = -\frac{x+k}{x+k+m+1}f_k$$
and can be rewritten as:
$$(x+m)f_{k+1} + (k+1)f_{k+1} + xf_k + kf_k = 0 \qquad \text{with} \qquad f_0 = \frac{1}{(x)_{m+1}}.$$
Then we have the following steps:

1. Pass to the generating functions:
$$(x+m+xt)f(t) + (t^2+t)f'(t) = (x+m)f_0$$
2. Apply the binomial transformation and simplify:
$$(x+m-t(m+1))g(t) + t(1-t)g'(t) = (x+m)f_0.$$

3. Extract the coefficient of t^n, with $n > 0$:

$$xg_n + mg_n - mg_{n-1} - g_{n-1} + ng_n - (n-1)g_{n-1} = 0 \qquad \forall n > 0;$$

$$(x + m + n)g_n = (m + n)g_{n-1}.$$

4. Compute initial conditions:

$$xg_0 + mg_0 = (x + m)f_0 \qquad \text{or} \qquad g_0 = f_0 = \frac{1}{(x)_{m+1}}.$$

By unfolding the recurrence we find the desired result.

As another example, we consider identity (6.1) in Chu and Yan's paper [1]:

$$\sum_{k=0}^{n} \binom{n}{k}\binom{n+k}{k}\frac{(-1)^k}{x+k} = \frac{(1-x)_n}{(x)_{n+1}}.$$

The sum can be written in a form suitable for the application of the Riordan transformation (A):

$$\sum_{k=0}^{n} \binom{n+k}{2k}\binom{2k}{k}\frac{(-1)^k}{x+k} = [t^n]\frac{1}{1-t}f\left(\frac{t}{(1-t)^2}\right)$$

where $f(t)$ is the generating function:

$$f(t) = \mathcal{G}\left(\binom{2k}{k}\frac{(-1)^k}{x+k}\right).$$

In order to apply the Riordan transformation method, it is not necessary to know explicitly this function, but we need to know the substitutions we have to perform. By successive differentiations we get:

$$f\left(\frac{t}{(1-t)^2}\right) = (1-t)g(t); \qquad f'\left(\frac{t}{(1-t)^2}\right) = \frac{(1-t)^3}{1+t}\left((1-t)g'(t) - g(t)\right);$$

$$f''\left(\frac{t}{(1-t)^2}\right) = \frac{(1-t)^5}{(1+t)^3}\left(2(2+t)g(t) - 2(2t+3)(1-t)g'(t) + (1+t)(1-t)^2g''(t)\right).$$

We find:

$$f_{k+1} = \binom{2k+2}{k+1}\frac{(-1)^{k+1}}{x+k+1} = -\frac{4k^2 + (4x+2)k + 2x}{x(k+1) + (k+1)^2}f_k$$

that is:

$$x(k+1)f_{k+1} + (k+1)^2f_{k+1} + 4k^2f_k + (4x+2)kf_k + 2xf_k = 0 \qquad \text{with} \qquad f_0 = \frac{1}{x}.$$

Then we proceed in the usual way:

1. Pass to the generating function:

$$xf'(t) + f'(t) + tf''(t) + 4tf'(t) + 4t^2f''(t) + 2(2x+1)tf'(t) + 2xf(t) = 0$$

2. Apply the Riordan transformation and simplify:

$$(x-1)g(t) + (xt - 3t + x + 1)g'(t) - t(1-t)g''(t) = 0.$$

3. Extract the coefficient of t^n with $n > 0$:

$$(x-1)g_n + (1-x)(n+1)g_{n+1} + (x-3)ng_n - n(n-1)g_n + (n+1)ng_{n+1} = 0,$$

which simplifies to:

$$(x + n + 1)g_{n+1} = (n + 1 - x)g_n \qquad \text{or} \qquad g_n = \frac{n-x}{x+n}g_{n-1}.$$

4. Compute initial conditions: $g_0 = f_0 = 1/x$.

By unfolding the recurrence we obtain the desired result.

References

[1] W. Chu and Q. Yan. Combinatorial identities on binomial coefficients and harmonic numbers. *Utilitas Mathematica*, 75:51–66, 2008.

[2] R. W. Gosper. Decision procedure for indefinite hypergeometric summation. *Proc. Nat. Acad. Sci. USA*, 75:40–42, 1978.

[3] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics.* Addison-Wesley, New York, 1989.

[4] M. Kauers. Summation algorithms for Stirling number identities. *Journal of Symbolic Computation*, 42(10):948–970, 2007.

[5] D. E. Knuth. *The art of computer programming. Vol. I.* Addison-Wesley, 1968.

[6] D. Merlini, R. Sprugnoli, and M. C. Verri. The method of coefficients. *American Mathematical Monthly*, 114:40–57, 2007.

[7] M. Petkovšek, H. S. Wilf, and D. Zeilberger. *A=B.* AK Peters, Natick, MA, 1996.

[8] R. D. Rainville. *Special functions.* MacMillan, New York, 1960.

[9] L. W. Shapiro, S. Getu, W.-J. Woan, and L. Woodson. The Riordan group. *Discrete Applied Mathematics*, 34:229–239, 1991.

[10] R. Sprugnoli. Riordan arrays and combinatorial sums. *Discrete Mathematics*, 132:267–290, 1994.

[11] R. P. Stanley. *Enumerative Combinatorics*, volume 2. Cambridge University Press, Cambridge, 1999.

[12] H. S. Wilf. *Generatingfunctionology.* Academic Press, Boston, 1990.

[13] J. Wimp and D. Zeilberger. Resurrecting the asymptotics of linear recurrences. *Journal of Mathematical Analysis and Applications*, 111:162–176, 1985.