Algorithms Seminar 2002–2004,
F. Chyzak (ed.), INRIA, (2005), pp. 7–10.

Available online at the URL
http://algo.inria.fr/seminars/.

Effective Scalar Product of Differentiably Finite Symmetric Functions

Frédéric Chyzak
Algorithms Project, INRIA (France)

March 11, 2003

Summary by Marni Mishna

Introduction

Many enumerative problems can be expressed using the scalar product of symmetric functions. As we shall see, we can set up expressions for the generating functions of objects which possess a certain kind of regularity as a scalar product. Two examples of this are k-regular graphs (graphs in which each vertex is of degree k), and secondly we have a class of semi-standard Young tableaux in which each entry appears k times. The generating series for these are respectively given by:

$$R_k(t) = \left\langle \left(\exp \sum_{i\geq 1} p_i/i\right)[e_2], \exp(h_k t)\right\rangle, \qquad Y_k(t) = \left\langle \left(\exp \sum_{i\geq 1} p_i/i\right)[e_1+e_2], \frac{1}{1-h_k t}\right\rangle.$$

Gessel [2] proved that this operation preserves some notion of D-finiteness, which implies that these generating functions should satisfy a linear differential equation with polynomial coefficients. This seminar, and [1] introduce algorithms to determine these differential equations, and prove that the algorithm is correct, and terminates.

1. Symmetric Functions

A *partition* of a positive integer n is a decreasing sequence of integers $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_k)$ whose sum is n. This is denoted $\lambda \vdash n$. A partition is either written as a vector or in power notation, for example $(7,7,4,4,1) = \langle 7^2 4^2 1\rangle$ are both partitions of 23. A *symmetric function* is a sum of monomials in a some variable set, such that it is invariant under any permutation of that variable set. We can write any symmetric function as a sum of *monomial symmetric functions*, defined for the variable set $\{x_1, x_2, \ldots\}$ with respect to some partition λ as

$$m_\lambda := \sum_{\sigma\in\mathcal{S}_{\mathbb{N}\setminus\{0\}}} (r_1!\, r_2! \cdots)^{-1}\, x_{\sigma(1)}^{\lambda_1} \cdots x_{\sigma(k)}^{\lambda_k}.$$

For example, $m_{(3,2,2)} = x_1^3x_2^2x_3^2 + x_3^3x_2^2x_1^2 + x_4^3x_1^2x_3^2 + \cdots$. We also have the *elementary symmetric functions*, $e_n = m_{\langle 1^n\rangle}$, and $e_\lambda = e_{\lambda_1}\cdots e_{\lambda_k}$; the homogeneous symmetric functions $h_n = \sum_{\lambda\vdash n} m_\lambda$, and $h_\lambda = h_{\lambda_1}\cdots h_{\lambda_k}$; and *power symmetric functions* $p_n = m_{(n)} = x_1^n + x_2^n + \ldots$, $p_\lambda = p_{\lambda_1}\cdots p_{\lambda_k}$. There is much to be said of the ring of symmetric functions, and the interested reader is pointed to Stanley [3, Chapter 7] for the intimate details. The essential here is that symmetric functions can always be written in terms of the power sum symmetric functions, and thus we have the liberty of describing our algorithms and definitions for symmetric functions as elements of $\Lambda = \mathbb{Q}[p_1, p_2 \ldots]$. In fact, we shall take this one step further and interest ourselves with the ring of series of symmetric functions, $\hat{\Lambda} = \mathbb{Q}[[p_1, p_2, \ldots]]$.

The scalar product of symmetric functions is the bilinear product defined by $\langle p_\lambda, p_\mu\rangle = z_\lambda \delta_{\lambda,\mu}$, with $\lambda = \langle 1^{r_1} 2^{r_2} \cdots \rangle$ giving $z_\lambda = (1^{r_1} r_1!)(2^{r_2} r_2!)\cdots$. It permits the extraction of the coefficients in symmetric function:

$$[x_1^{\lambda_1} x_2^{\lambda_2} \dots x^{\lambda_k}]\, \phi = [m_\lambda]\, \phi = \langle h_\lambda, \phi\rangle.$$

This is precisely the use we have in mind.

To conclude this all-too-brief introduction of symmetric functions, we describe one type of composition that turns out to be quite useful here: *plethysm*, written $f[g]$. We can most easily define it using the power sum symmetric functions. It is defined by $p_n[\psi(p_1, p_2, \dots)] = \psi(p_n, p_{2n}, \dots)$, along with $(\phi_1 + c\phi_2)[\psi] = \phi_1[\psi] + c\phi_2[\psi]$ and $(\phi_1 \cdot \phi_2)[\psi] = \phi_1[\psi] \cdot \phi_2[\psi]$.

2. D-Finite Functions and Holonomic Modules

The second key ingredient is D-finite functions. A function ϕ is said to be *differentiably finite* or just simply *D-finite* in $\mathbb{K}[[x_1, \dots, x_r]]$ if and only if the $\partial_1^{\alpha_1} \cdots \partial_r^{\alpha_r} \phi$ generate a finite dimensional vector space over $\mathbb{K}(x_1, \dots, x_r)$. In this case, ϕ is determined by a system of linear differential equations.

There are algorithms to make effective the closure of D-finite functions under $+$, $\times$, derivation, algebraic substitution, indefinite and definite integration, hadamard product and diagonals.

In order to treat symmetric functions, however, we first need to expand this definition to functions with an infinite number of variables. The function $\phi(x_1, x_2, \dots)$ is D-finite in $K[[x_1, x_2, \dots]]$ if for any r, $\phi(x_1, \dots, x_r, 0, \cdots)$ is D-finite in $K[[x_1, x_2, \dots, x_r]]$. This case does not enjoy all of the closure properties of the previous, nonetheless we have closure under $+$, $\times$, ∂_i, extension of coefficients, rational substitution, and exponentials of polynomials. We say that a symmetric function $\phi \in \hat{\Lambda}$ is D-finite if it is D-finite in $\mathbb{Q}[[p_1, p_2, \dots]]$. For example, the famous sums of symmetric functions, $\sum_n h_n = \prod_{i\geq 1}(1 + x_i) = \exp\left(\sum_n p_n/n\right)$ is D-finite under this definition.

In the symmetric case we also have closure under *Kronecker* (or tensor) product ($p_\lambda * p_\mu = \delta_{\lambda,\mu} z_\lambda p_\lambda$), and closure under plethysm with a polynomial. The property we are most interested here is that if ϕ and ψ are both D-finite symmetric functions in $K[t_1, t_2, \dots, t_n]$, and ϕ is a function of at most a finite number of power sum symmetric functions then $\langle \phi, \psi\rangle$ is a D-finite function of $k[t_1, t_2, \dots, t_n]$ [2].

2.1. The Weyl algebra.

We will present and prove our algorithm in the context of the *Weyl algebra* A_p of linear differential operators. This algebra is particularly useful for manipulating differential equations with polynomial coefficients. It is in this algebra that we can make many D-finite closure properties effective. We define $A_p := \mathbb{C}\langle p_1, \dots, p_n, \partial_1, \dots, \partial_n; \mathcal{R}_p\rangle$, with $\partial_i = d/dp_i$ and relations $\mathcal{R}_p : \partial_i p_j = p_j \partial_i + \delta_{i,j}$, and $p_i p_j - p_j p_i = \partial_i \partial_j - \partial_j \partial_i = 0$. We likewise consider additional variables, for example t and ∂_t, in $A_{p,t}$.

We shall denote left ideals and modules respectively by $I_\phi = \operatorname{ann}_{A_{p,t}} \phi = \{\, L \in A_{p,t} \mid L\phi = 0 \,\}$ and $A_{p,t}\phi = \{\, L\phi \mid L \in A_{p,t} \,\} \simeq A_{p,t}/I_\phi$. We filter A_p by total degree. First define $F_d = \{\, P \in A_p \mid \deg P \leq d \,\}$. Clearly, $F_d F_{d'} \subset F_{d+d'}$ $F_d \subset F_{d+1}$, and $\bigcup_{d\geq 0} F_d = A_p$. The module $A_p\phi$ is *holonomic* when $\dim_{\mathbb{C}} F_d\phi = O(d^n)$. The function ϕ is *holonomic* if $A_p\phi$ is holonomic. There is a key theorem of Kashiwara and Takayama which says that ϕ is holonomic if and only if ϕ is D-finite.

3. The Scalar Product of Symmetric Functions

3.1. Adjunction.

The magic operation that allows us to manipulate the scalar product is its adjunction, or adjoint operator. For P and Q in A_p, $\langle P\phi, \psi\rangle = \langle \phi, P^{\mathrm{adj}}\psi\rangle, (P^{\mathrm{adj}})^{\mathrm{adj}} = P$, and

$(PQ)^{\mathrm{adj}} = Q^{\mathrm{adj}} P^{\mathrm{adj}}$. So for the *usual* scalar product of functions, $(f,g) \mapsto \int fg$, $\langle \partial_i \phi, \psi \rangle = -\langle \phi, \partial_i \psi \rangle$ implies that $p_i^\star = p_i$ and $\partial_i^\star = -\partial_i$, a bi-product of integration by parts. It is easy to calculate that in the case of the symmetric scalar product:

$$\langle p_i \phi, \psi \rangle = \langle \phi, i\partial_i \psi \rangle \quad \text{implies} \quad p_i^\diamond = i\partial_i \quad \text{and} \quad \partial_i^\diamond = i^{-1} p_i.$$

If M is a left module, then M^{adj} is a right module. These two modules have the same support and addition, however, we define multiplication this way for any$P \in A_p$, and any $m \in M^{\mathrm{adj}}$, $mP := (P^{\mathrm{adj}})m$. So, if $M = A_p\phi$, $M^{\mathrm{adj}} = \phi^{\mathrm{adj}} A_p$ with $\phi^{\mathrm{adj}} = \phi$, then $I_{\phi^{\mathrm{adj}}} = (I_\phi)^{\mathrm{adj}}$, and $(A_p\phi)^{\mathrm{adj}} \simeq A_p / I_{\phi^{\mathrm{adj}}}$.

3.2. **Calculating with the scalar product.** The algorithm also works in the more general cases, but for the sake of clarity and its good sister brevity we consider a unique auxiliary t variable, and we impose $\partial_t \phi = 0$.

The important link between annihilators of ϕ and ψ and of $\langle \phi, \psi \rangle$ is the following equation. Imagine a $W = (P^\diamond SU + TQ)$ with $S \in A_p$, $T \in A_{p,t}$, $U \in A_t$, $P \in I_\phi \cap A_p$ and $Q \in I_\psi$. Then,

$$\big\langle \phi, W\psi \big\rangle = \big\langle \phi, (P^\diamond SU + TQ)\psi \big\rangle = \langle S^\diamond P\phi, U\psi \rangle + \langle \phi, TQ\psi \rangle = \langle S^\diamond \cdot 0, U\psi \rangle + \langle \phi, T \cdot 0 \rangle = 0.$$

If by chance, $W \in \mathbb{C}\langle t, \partial_t \rangle$, since $\partial_t \langle \phi, \psi \rangle = \langle \phi, \partial_t \psi \rangle$, and $t\langle \phi, \psi \rangle = \langle t\phi, \psi \rangle = \langle \phi, t\psi \rangle$, then in fact, $W\langle \phi, \psi \rangle = \big\langle \phi, W\psi \big\rangle = 0$. So, if we calculate generators for $\big((\mathrm{ann}_{A_p}\, \phi)^\diamond A_t + (\mathrm{ann}_{A_{p,t}}\, \psi)\big) \cap \mathbb{C}\langle t, \partial_t \rangle$ any element in this intersection will annihilate the scalar product, that is, will represent a differential equation satisfied by the scalar product, which is PRECISELY WHAT WE WANT! Now, we must add that not all annihilators of the scalar product have to be of this form. However, that there are any is sufficient for our purposes.

The idea of the calculation is to iteratively construct elements in truncations of $(I_\phi)^\diamond$ and I_ψ, and at each step search for some linear combination which is in the ring A_t (for example, using our good friend, Gröbner bases.)

3.3. **Algorithm.** The inputs are ϕ D-finite in $\mathbb{C}[[p_1, \ldots, p_n]]$, and ψ D-finite in $\mathbb{C}[[t, p_1, \ldots, p_n]]$.

1. Calculate the Gröbner bases $(\mathcal{G}_\phi)^\diamond$ for $(\mathrm{ann}\, \phi)^\diamond$ and $\mathcal{G}_\psi$ for $(\mathrm{ann}\, \psi)$ with respect to the same order.
2. set $B = \{\}$;
3. iterate over monomials α from $[p_1, \ldots, p_n, \partial_1, \ldots, \partial_n, \partial_t]$ in an increasing order;
   (a) set α_ϕ to the remainder from *left* division of α by $(\mathcal{G}_\phi)^\diamond$;
   (b) set α_ψ to the remainder from *right* division of α by $\mathcal{G}_\psi$;
   (c) put both $\alpha - \alpha_\phi$ and $\alpha - \alpha_\psi$ in B and reduce to eliminate the p_i and the ∂_i by linear algebra;
   (d) If B contains an element P in t and ∂_t only, stop and output this element.

3.4. **Why does it work?** We build a helper intermediate space $S := (A_{p,t}\phi)^\diamond \otimes_{A_p[t]} (A_{p,t}\psi)$, which has a natural injective map onto $\langle A_{p,t}\phi, A_{p,t}\psi \rangle$, and in which we have the relations $(P\phi)^\diamond \otimes \psi = (\phi^\diamond P^\diamond) \otimes \psi = \phi^\diamond \otimes (P^\diamond \psi)$. In fact, $S \simeq \big(A_{p,t}^\diamond \otimes_{A_p[t]} A_{p,t}\big)/K$ for $K = I_\phi^\diamond \otimes_{A_p[t]} A_{p,t} + A_{p,t} \otimes_{A_p[t]} = I_\psi \,\mathrm{ann}_{A_{p,t}^\diamond \otimes_{A_p[t]} A_{p,t}} (\phi^\diamond \otimes \psi)$. The algorithm progressively constructs a basis for the vector space K, and then using linear algebra determines a basis for $K \cap \mathbb{C}\langle t, \partial_t \otimes 1 + 1 \otimes \partial_t \rangle$. This maps to $\big((\mathrm{ann}_{A_p}\, \phi)^\diamond A_t + (\mathrm{ann}_{A_{p,t}}\, \psi)\big) \cap \mathbb{C}\langle t, \partial_t \rangle$.

3.5. **Will it finish?** To prove termination, we first prove that S is infact a holonomic module whenever ϕ and ψ are holonomic. Essentially, this is done by building a module which injectively maps onto it, and the holonomy of this module is easily established, in part by passing through the usual scalar product. Since S is a holonomic module, the intersection $K \cap \mathbb{C}\langle t, \partial_t \otimes 1 + 1 \otimes \partial_t \rangle$ cannot reduce to $\{0\}$ because of a restriction on the dimension of S. The algorithm eventually produces every element of K, and thus the terminating condition will be satisfied.

4. **Applications to Combinatorics**

FINALLY, we return to the enumeration problem. Basically, we encode all graphs by their degree sequence. The sum of this encoding over all graphs, ϕ_G, gives a particularly nice D-finite symmetric series!

$$\phi_G = \sum_{g\in G} \prod_{\{i,j\}\in g} x_i x_j = \prod_{i<j}(1+x_i x_j) = \left(\exp \sum_{i\geq 1} \frac{p_i}{i}\right)[e_2].$$

To determine the number $r_{n,k}$ of graphs on n vertices with all valencies equal to k, we EXTRACT THE COEFFICIENT of $x_1^k x_2^k \cdots x_n^k$ in this giant series.

$$R_k(t) := \sum_{n\geq 0} r_{n,k} \frac{t^n}{n!} = \sum_{k\geq 0} \big\langle \phi_G, h_{\langle k^n\rangle}\big\rangle \frac{t^n}{n!} = \big\langle \phi_G, \sum_{n\geq 0} h_{\langle k^n\rangle} \frac{t^n}{n!}\big\rangle = \big\langle \phi_G, \exp(h_k t)\big\rangle.$$

This gives a scalar product that we can apply the algorithm to:

$$R_k(t) = \left\langle \exp\left(\sum_{\substack{i\leq k \\ i \text{ pair}}} (-1)^{i/2} \frac{p_i^2}{2i} + \frac{p_i}{i} + \sum_{\substack{i\leq k \\ i \text{ impair}}} \frac{p_i^2}{2i}\right), \exp\left(t \sum_{\lambda\vdash k} \frac{p_\lambda}{z_\lambda}\right)\right\rangle.$$

The current implementation of the algorithm can determine the differential equation satisfied by $R_k(t)$ for $t = 1, \ldots, 4$.

We can do a similar treatment to get $Y_k(t)$. Once this is done for a few values of k, the following conjecture arises.

Conjecture 1. *The number $y_{n,k}$ of k-uniform Young Tableaux of size kn has asymptotic growth like*

$$y_{n,k} \sim \frac{1}{\sqrt{2}} \left(\frac{e^{k-2}}{2\pi}\right)^{k/4} n!^{k/2-1} \left(\frac{k^{k/2}}{k!}\right)^n \frac{\exp(\sqrt{kn})}{n^{k/4}}, \qquad n\to\infty.$$

The scalar product is closely related to the Kronecker product and using a variation of the algorithm we have the following.

Proposition 1. *The following identity holds, where s_λ is the Schur function indexed by λ,*

$$\left(\sum_\lambda s_\lambda\right) * \left(\sum_\lambda s_\lambda\right) = \exp\left(\sum_{n\geq 1} \frac{p_{2n-1}}{(2n-1)(1-p_{2n-1})}\right)\left(\prod_{n\geq 1}(1-p_n^2)\right)^{-1/2}.$$

Bibliography

[1] Chyzak (Frédéric), Mishna (Marni), and Salvy (Bruno). – Effective scalar products of D-finite symmetric functions. *Journal of Combinatorial Theory, Series A*, 2005. – In press.

[2] Gessel (Ira M.). – Symmetric functions and P-recursiveness. *Journal of Combinatorial Theory, Series A*, vol. 53, n° 2, 1990, pp. 257–285.

[3] Stanley (Richard P.). – *Enumerative combinatorics. Vol. 2.* – Cambridge University Press, Cambridge, 1999, *Cambridge Studies in Advanced Mathematics*, vol. 62, xii+581p.