SOME APPLICATIONS AND TECHNIQUES FOR GENERATING FUNCTIONS

P.Massazza - N.Sabadini
Dip. Scienze dell'Informazione - Università di Milano

Abstract

In this paper we apply generating functions techniques to the problem of deciding whether two probabilistic finite state asynchronous automata define the same events. We prove that the problem can be solved by an efficient parallel algorithm, in particular showing that it is in the class DET. Furthermore, we develop some methods for studying properties of generating functions, in particular from the point of view of the algebricity.

1. Introduction

Generating functions have been applied successfully in combinatorics, probability theory and formal languages theory. In combinatorial theory, they represent one of the basic techniques for dealing with problems of enumeration [St][DRS][Go]. In formal language theory, Schuetzenberger developped a theory of generating functions in a finite number of non commuting variables to solve problems in context free languages (CFL) [CS].

In particular, in some cases generating functions techniques give an analytic method for approaching the problem of determining whether a language L belongs to a given class C. Informally, the idea is to associate to every language in C a generating function and to state properties that these functions must satisfy. If the generating function of L does not satisfy these properties then L does not belong to C. For example, it is well known from a classical result due to [CS] that every unambiguous CFL has an algebraic generating function. So, it is possible to prove that some CFL's are inherently ambiguous, by showing that their generating functions are not algebraic: this method has been successfully used in [Fl] to solve some open problems on the ambiguity of CFL's. Some conditions for the applicability of this method to classes of trace languages have been recently given in [BS2].

Furthermore, generating functions techniques can be used to solve decision problems on languages. For example, in [BPS] it is proved that the equivalence problem for some classes of unambiguous regular trace languages is decidable, by showing that two languages coincide iff a suitable rational generating function is identically zero.

Hence, it is possible to individuate two research topics:
1) to find new and significant applications of generating functions techniques to decision problems,
2) to develop suitable methods for studying properties of classes of generating functions (for example, a natural question is whether the generating functions of CFL's lie in a particular class of trascendental functions [Fl] or whether the algebricity of the generating function for subclasses of CFL's is decidable).

In this paper we consider both directions: as a main application in the first one, we extend the notion of finite state asynchronous automaton (introduced in [Zi] in order to characterize the class of recognizable trace languages) to the probabilistic case and we study equivalence problems on the events defined by these automata. In particular, we introduce a notion of distance between events, showing that this distance function is computable. Furthermore we prove that the problem of deciding equality between events is in the class DET.

In the second direction, we study the algebricity problem for generating functions. Although the general problem of determining whether the generating function of a CFL is algebraic is undecidable [BS1], nevertheless for a suitable subclass of languages it is possible to develop proper techniques. In particular, we study the problem of deciding whether the linear space of the generating functions, whose coefficients are the solutions of linear homogeneous recurrence equations (having polynomial coefficients), only contains algebraic functions. We develop an algorithm for this problem, based on Klein's method for solving differential equations of the second order having polynomial coefficients.

2. Preliminary definitions on generating functions

In this section we recall some basic definitions and results about generating functions of languages. For all the definitions on formal languages we refer to a classical book as [Sa].

Given a finite alphabet Σ, we associate with a language $L \subseteq \Sigma^*$ its *enumeration* or *counting sequence* $\{a_n\}$, defined by $a_n = \#\{x \in L \mid |x| = n\}$. This sequence univocally defines $F_L(z) = \sum_{n=0}^{\infty} a_n z^n$, called the *generating function* of the language L. This function is an analytic function in a neighbourhood of the origin and its convergence radius is at least $1/\#\Sigma$. In this paper we are interested in *algebraic* analytic functions; we recall that a complex function f(z) is algebraic if there exists a finite sequence of polynomials $q_0(x),\dots, q_d(x) \in \mathbf{C}[x]$ (where $\mathbf{C}$ is the complex field) such that for every $z \in \mathbf{C}$ $\sum_{j=0}^{d} q_j(z)f(z)^j = 0$. The degree of the algebraic function is the least integer d such that the above relation holds. Useful necessary conditions for a function to be algebraic are given in [Fl]; in particular we recall the following result of Comtet [Com]:

Fact 2.1- Let $f(z) = \sum_{n=0}^{\infty} c_n z^n$ be an algebraic function of degree d. Then there exist an integer n° and a finite sequence of polynomials $p_0(x),\dots, p_q(x) \in \mathbf{Z}[x]$ such that:

1) $p_q(x) \neq 0$
2) the degree of $p_j(x) < d$ for every j
3) for every $n \geq n^\circ$ $p_0(n)c_n + p_1(n)c_{n-1} + \dots + p_q(n)c_{n-q} = 0$.

3. Preliminary definitions on trace languages

In this section we recall some basic definitions about traces and trace languages, introduced by Mazurkiewicz [Ma] in order to formally describe the behaviour of concurrent systems, and extensively studied by many authors [see AR for a review]. Given a finite alphabet Σ and a concurrency relation on Σ (i.e. a symmetric and irreflexive relation $C \subseteq \Sigma \times \Sigma$) we consider the *concurrent alphabet* $<\Sigma, C>$ and the *free*

partially commutative monoid $F(\Sigma, C)$ generated by $<\Sigma, C>$, defined as the initial object in the category of monoids generated by Σ and satisfying, besides the usual monoid axioms, the set of commutativity laws $\{ab = ba \mid aCb\}$. From standard results $F(\Sigma, C)$ is the quotient structure $F(\Sigma, C) = \Sigma^*/=_C$, where $=_C$ is the least congruence on Σ^* extending C. A *trace* t is an element of $F(\Sigma, C)$ which can be interpreted as an equivalence set of words, a *trace language* is a subset of $F(\Sigma, C)$; given a string $x \in \Sigma^*$ we will denote by $[x]_C$ the equivalence class of x (i.e. the trace generated by x) and by $|x|$ its length. So, given a language $L \subseteq \Sigma^*$, the trace language $T = [L]_C$ generated by L is defined as $T = \{t = [x]_C \mid x \in L\}$.
The notion of *regular trace language* (introduced by Mazurkiewicz [Ma]), is defined as follows: T is regular iff $T = [L]_C$ and L is a regular language on Σ. We denote by $R(\Sigma, C)$ the class of regular trace languages. An interesting subclass of the class of regular trace languages is the class $R_1(\Sigma, C)$ of *deterministic* or *unambiguous* languages [BMS], defined as the class of trace languages T for which there exists a regular language $L \subseteq \Sigma^*$ such that it holds the following:
a) $T= [L]_C$ b) $\forall x \in \Sigma^*$ $(\#\{[x]_C \cap L\} \leq 1)$.
Given a trace language $T \subseteq F(\Sigma, C)$, we define the *generating function* $f_T(z)$ of T as :

$$f_T(z) = \sum_{k=0}^{\infty} \#\{t \mid t \in T, |t| = k\}z^k.$$

An analysis of some properties of generating functions of regular trace languages has been carried out in [BS2].
Finally, we present two simple examples of solution of decision problems over languages by means of generating functions techniques. Let us consider the following problems:

Problem 1:

Instance: A context free grammar G on Σ

Question: to decide whether $L_G = \Sigma^*$, where L_G is the language generated by G

Problem 2:

Instance: A regular grammar R on Σ for a concurrent alphabet $<\Sigma, C>$

Question: to decide whether $T_R = F(\Sigma, C)$, where $T_R = [L_R]_C$ and L_R is the language generated by R

It is well known that these problems are undecidable [Sa][AH]. Let $F_{L_G}(z)$ and $F_{T_R}(z)$ be the generating functions of L_G and T_R; it is immediate to prove that:

1) $L_G = \Sigma^*$ iff $F_{L_G}(z) = \dfrac{1}{1 - \#\Sigma z}$

2) $T_R = F(\Sigma,C)$ iff $F_{T_R}(z) = \dfrac{1}{\sum_k (-)^k C_k z^k}$, where C_k is the number of k-cliques in the concurrency relation [BBMS].

We recall that if G is unambiguous then $F_{L_G}(z)$ is algebraic and if L_R generates T_R deterministically then $F_{T_R}(z)$ is rational; in these cases the problems consist of verifying the equality of algebraic (rational) functions. Hence, we can conclude that the previous problems restricted to unambiguous CFL's and deterministic regular trace languages become decidable.

4. Probabilistic Asynchronous Automata

In this section we introduce the notion of probabilistic finite state asynchronous automaton (PFSAA), which can be considered as an interesting type of stochastic Petri Nets [Mo] We study the equivalence problem for this model using suitable generating functions techniques. In particular we prove that the "distance" between behaviours of PFSAA's with rational probabilities is computable and that the "equality" of behaviours is decidable by means of an efficient parallel algorithm.

Def.4.1- A *probabilistic finite state asynchronous automaton* (PFSAA) with n processes is a tuple $A = <P_1, P_2, \ldots, P_n, \Delta, F>$ where:

- $\forall i\ 1 \leq i \leq n\ P_i = <\Sigma_i, S_i, s_{i_0}>$ is the i-th process, Σ_i being a finite non empty alphabet, S_i denoting a finite set of (local) states s.t. $S_i \cap S_j = \varnothing$ for $j \neq i$, $s_{i_0} \in S_i$ standing for the initial state of P_i,
- $F \subseteq \prod_{i=1,n} S_i$ is the set of final states,
- $\Delta = \{\delta_\sigma \mid \sigma \in \bigcup_{i=1,n} \Sigma_i\}$ is a set of stochastic transition matrices (we recall that a stochastic matrix is a square matrix with nonnegative real entries and with row sums equal to 1).

Let us consider the following notations:

- $\Sigma = \bigcup_{i=1,n} \Sigma_i$,
- $\mathrm{Proc} = \{1, \ldots, n\}$,
- $Q = \prod_{i=1,n} S_i$ (the set of global states of A),
- $q_o = (s_{1o}, \ldots, s_{no})$ (the initial state of A).

$P(\sigma) = \{i \in \mathrm{Proc} \mid \sigma \in \Sigma_i\}$ is the set of processes which can "execute" an action $\sigma \in \Sigma$. For every action $\sigma \in \Sigma$ the set Δ contains exactly one stochastic matrix δ_σ whose indices are the elements of $\prod_{i \in P(\sigma)} S_i$; furthermore, $\delta_\sigma((s_{j_1}, \ldots, s_{j_k}), (s'_{j_1}, \ldots, s'_{j_k}))$ gives the probability that the automaton, while executing the action σ, modifies the local states of the processors which can execute the action from $(s_{j_1}, \ldots, s_{j_k})$ to $(s'_{j_1}, \ldots, s'_{j_k})$.

For each symbol σ in Σ, we extend δ_σ to the matrix δ'_σ, which has as indices the elements of $\prod_{i \in \mathrm{Proc}} S_i$. In order to do this, we introduce the following notation: let $v = (s_1, \ldots, s_n)$ be an element of $\prod_{i \in \mathrm{Proc}} S_i$, we denote by $\mathrm{Pro}_\sigma(s_1, \ldots, s_n)$ the element of $\prod_{i \in P(\sigma)} S_i$ obtained from v by erasing the states of the processors which cannot execute the action σ. Now the extended matrix δ'_σ can be defined as follows:

$\delta'_\sigma((s_1, \ldots, s_n), (s_1', \ldots, s_n'))$ = <u>if</u> $\forall\ i \notin P(\sigma)\ s_i = s_i'$
<u>then</u> $\delta_\sigma(\mathrm{Pro}_\sigma(s_1, \ldots, s_n), \mathrm{Pro}_\sigma(s_1', \ldots, s_n'))$
<u>else</u> 0.

It is clear that if actions α and β operate on disjoint sets of processes then they may be executed independently, so the *concurrency (independency) relation* of A is defined as $C_A = \{(\alpha,\beta) \in \Sigma x \Sigma \mid P(\alpha) \cap P(\beta) = \varnothing\}$ and A defines a concurrent alphabet $<\Sigma, C_A>$ in a natural way. It is immediate to observe the following fact:

Fact 4.1- Given two symbols $\alpha, \beta \in \Sigma$, $\alpha C_A \beta$ implies that $\delta'_\alpha \circ \delta'_\beta = \delta'_\beta \circ \delta'_\alpha$, where $\circ$ denotes the usual product of matrices.

As a consequence, if the words $x_1 \ldots x_n$, $x'_1 \ldots x'_n$ are in the same trace (interpreted as equivalence class of words), then $\delta'_{x_1} \circ \ldots \circ \delta'_{x_n} = \delta'_{x'_1} \circ \ldots \circ \delta'_{x'_n}$.
Furthermore, we observe that the concurrency relation defined by A can be interpreted as a graph and that to every *clique* $C = \{\sigma_1, \ldots, \sigma_s\}$ of C_A we can associate the matrix $\delta'(C) = \delta'_{\sigma_1} \circ \ldots \circ \delta'_{\sigma_n}$.
Given a PFSAA A, let π and μ denote the stochastic vectors which have as indices the elements of $\prod_{i \in Proc} S_i$, defined as:
$\pi(s_1, \ldots, s_n) =$ <u>if</u> $(s_1, \ldots, s_n) = (s_{1\,0}, \ldots, s_{n\,0})$ <u>then</u> 1 <u>else</u> 0
$\mu(s_1, \ldots, s_n) =$ <u>if</u> $(s_1, \ldots, s_n) \in F$ <u>then</u> 1 <u>else</u> 0
(π and μ are the characteristic vectors of the initial state and of final states of the automaton).
Now we can define the *behaviour* of the PFSAA A with input $t = [x_1 \ldots x_n]_{C_A} \in F(\Sigma, C_A)$, as the probability that A, starting from the initial state, reaches a final one; this probability is obtained as $\pi \circ \delta'_{x_1} \circ \ldots \circ \delta'_{x_n} \circ \mu_T$, where μ_T is the transposed vector of μ. More formally we have:

Def.4.2- Given a PFSAA A, the *event* realized by A is the function $S_A : F(\Sigma, C_A) \rightarrow [0,1]$ where $S_A(t) = \pi \circ \delta'_{x_1} \circ \ldots \circ \delta'_{x_n} \circ \mu_T$ and $[x_1 \ldots x_n]_{C_A} = t$.

Now, given two PFSAA A_1 and A_2 on the same concurrent alphabet $<\Sigma, C>$, we are interested in the Equivalence Problem stated as follows:

Equivalence Problem for PFSAA (EPFSAA)
Given a concurrent alphabet $<\Sigma, C>$:
Instance: two PFSAA A_1, A_2 on $<\Sigma, C>$
Question: $S_{A_1} = S_{A_2}$?

A more general question consists of defining a suitable notion of distance between the events realized by two PFSAA's, and of giving a procedure to compute it.

Def.4.3 - $d(S_{A_1},S_{A_2}) = \mathrm{Sup}_n \dfrac{\sum_{|t|=n}\left(S_{A_1}(t) - S_{A_2}(t)\right)^2}{F_n}$, where F_n is the number of traces in $F(\Sigma, C)$.

The corresponding problem is:

Distance between events

Instance: two PFSAA A_1, A_2 on $<\Sigma, C>$

Question: to compute $d(S_{A_1}, S_{A_2})$.

In order to solve this problem, we consider the following generating function in the complex variable z:

$$f_{A_1,A_2}(z) = \sum_{k=0}^{\infty} \sum_{|t|=k} \left(S_{A_1}(t) - S_{A_2}(t)\right)^2 z^k.$$

As a first result, by using formal power series in partially commutative variables [La], we show that $f_{A_1,A_2}(z)$ is a rational function.

We recall some basic definitions on formal power series in partially commutative variables: let $<A, +, ., 1>$ be a ring with unity (not necessarily commutative) and $F(\Sigma, C)$ the free partially commutative monoid, then a *formal power series in partially commutative variables* is a function $\psi : F(\Sigma, C) \rightarrow A$, represented as the formal sum $\sum_{t\in F(\Sigma,C)} \psi(t)t$. The set of formal power series on $F(\Sigma, C)$ form a ring $A[[F(\Sigma, C)]]$, by considering the usual operations of sum (+) and Cauchy product (.):

$(\phi + \psi)(t) = \phi(t) + \psi(t)$

$(\phi \,.\, \psi)(t) = \Sigma_{xy=t}\, \phi(x)\psi(y)$

with the identity **1** defined as follows:

$\mathbf{1}(t) =$ <u>if</u> $t = [\varepsilon]_C$ <u>then</u> 1 <u>else</u> 0 (being ε the empty word of Σ^*).

Given the concurrent alphabet $<\Sigma, C>$, let *C* be the set of cliques of C (we recall that every clique in *C* individuates a monomial in which every two symbols commute). We will use the following result (it can be immediately obtained by applying the techniques used to prove the Moebius inversion formula):

Lemma 4.1- Let $\varphi : F(\Sigma, C) \rightarrow <A, ., 1>$ be a monoid morphism, then it holds:

$$\left(\sum_t \varphi(t)t\right)\left(\sum_{c\in C} (-)^{|c|}\varphi(c)c\right) = \mathbf{1}.$$

In particular, by considering the ring **Z** of integers and the morphism $\varphi(t) = 1$, we obtain the usual Moebius inversion formula:

$$\left(\sum_t t\right)\left(\sum_{c\in C} (-)^{|c|}c\right) = \mathbf{1}.$$

Now, we can state our main result:

Theor.4.1- Given two PFSAA A_1, A_2 on a concurrent alphabet $<\Sigma, C>$, the associated generating function $f_{A_1,A_2}(z)$ is a rational function in z.

Proof (outline): Let M_N be the monoid of NxN square matrices with real components with the usual product operation, and $\varphi : F(\Sigma, C) \rightarrow M_N$ be a monoid morphism. We observe that the formal series $\Sigma_t\varphi(t).t$ can be interpreted as a formal series on the ring $<M_N, +, .>$ or as a matrix with formal series as components. Given two 1xN vectors π, μ with real components, it holds:

$$\sum_t(\pi\varphi(t)\mu_T)t = \pi\left(\sum_t\varphi(t)t\right)\mu_T$$

By lemma 4.1 and by considering the morphism from the ring of formal power series on $<\Sigma, C>$ and the ring of series in the variable z induced by the substitutions $\sigma \rightarrow z$ ($\sigma \in \Sigma$), we obtain:

$$\left(\sum_0^\infty\left(\sum_{|t|=n}\varphi(t)\right)z^n\right)\left(\sum_{c\in C}(-)^{|c|}\varphi(c)z^{|c|}\right) = \mathbf{1},$$

and finally

$$\sum_n\left(\sum_{|t|=n}\pi\varphi(t)\mu_T\right)z^n = \pi\left(\sum_{c\in C}(-)^{|c|}\varphi(c)z^{|c|}\right)^{-1}\mu_T. \qquad (*)$$

Given two PFSAA A_1,A_2 on $<\Sigma, C>$, we denote by π_i, μ_i, $\delta'_i(\sigma)$ (for $\sigma \in \Sigma$), respectively, the characteristic vector of initial states, of final states (with dimensions $1xn_i$) and the stochastic (n_ixn_i) matrices associated to A_i ($i = 1, 2$), where n_i is the number of global states of the automaton.

Let be:

$$\pi = (\pi_1, \pi_2) \otimes (\pi_1, \pi_2) \qquad \mu = (\mu_1, \mu_2) \otimes (\mu_1, -\mu_2)$$

$$\forall\sigma \in \Sigma \quad \varphi(\sigma) = (\delta'_1(\sigma) \oplus \delta'_2(\sigma)) \otimes (\delta'_1(\sigma) \oplus \delta'_2(\sigma))$$

where $\oplus$ and $\otimes$ are the usual direct sum and Kronecker's product, and (π_1, π_2) is the $(1x(n_1+n_2))$ vector obtained by joining the vectors π_1, π_2.

Let us consider the monoid morphism $\varphi : F(\Sigma, C) \rightarrow M_N$ defined as before, where $N = (n_1 + n_2)^2$, and a trace $t = [x_1...x_p]_C$; by using the usual properties of $\oplus$ and $\otimes$ we obtain:

$\pi\varphi(t)\mu_T = \pi\varphi(x_1)...\varphi(x_p)\mu_T = (\pi_1\delta'_1(x_1)...\delta'_1(x_p)\mu_{1T} - \pi_2\delta'_2(x_1)....\delta'_2(x_p)\mu_{2T})^2$
$= (S_{A_1}(t) - S_{A_2}(t))^2$

Hence $\sum_0^\infty\left(\sum_{|t|=n}\pi\varphi(t)\mu_T\right)z^n = f_{A_1,A_2}(z)$ and from relation (*) the thesis follows ($\sum_{c\in C}(-)^{|c|}\varphi(c)z^{|c|}$ is a matrix whose components are polynomials in the variable z).

Now, given two PFSAA's A_1, A_2 on $F(\Sigma, C)$, let be $G_n = \sum_{|t|=n} \left(S_{A_1}(t) - S_{A_2}(t)\right)^2$; we recall that F_n is the number of traces of length n. If the associated stochastic matrices have rational number as entries, since $\Sigma_{0,\infty} G_n.z^n$ and $\Sigma_{0,\infty} F_n.z^n$ are rational functions with rational coefficients, then we can conclude that there exist polynomials with algebraic coefficients $p_1,..,p_m$, $q_1,..,q_p$ and algebraic numbers $a_1,..,a_m$, $b_1,..,b_p$ such that $G_n = \sum_{k=1}^{m} p_k(n)a_k^n$, $F_n = \sum_{j=1}^{p} q_j(n)b_j^n$.

Hence, there is an integer w such that $\mathrm{Lim}_{k\to\infty} \frac{G_{kw+s}}{F_{kw+s}} = r_s$ ($s = 1,..., w$), where r_s is an algebraic number. This is the main step which allows to conclude:

Fact.4.2 - The distance function d for PFSAA's, whose associated stochastic matrices have rational number as entries, is computable.

The previous result implies that the problem EPFSAA for PFSAA's with rational entries is decidable.

More generally, by observing that $f_{A_1,A_2}(z) = 0$ iff $S_{A_1} = S_{A_2}$, we can reduce the problem of deciding the equivalence of PFSAA to the problem of determining whether the rational function $f_{A_1,A_2}(z) = \pi\left(\sum_{c\in C} (-)^{|c|}\varphi(c)z^{|c|}\right)^{-1} \mu_T$ is identically zero.

Let F be a subfield of **R** whose elements admit a representation such that, firstly, the operations of sum and product are total recursive, secondly the equality is decidable (for example, we could consider the field of rational (or algebraic) numbers). The following fact holds:

Fact 4.3 - The Equivalence problem for PFSAA defined by stochastic matrices, with probability values in F, is decidable.

We are now interested in a classification of EPFSAA from the complexity point of view: we consider the case of automata such that the associated stochastic matrices have rational numbers as entries. We will prove that in this case EPFSAA is "efficiently parallelizable"; in order to do that we recall the definitions of the hierarchy $\{NC^k\}$ and of the class DET [Co]:

Def.4.4 - For every integer k, NC^k is the set of functions computable by a family of uniform boolean circuits of depth $O(\log^k n)$ and size $n^{O(1)}$. The class NC is defined as: $NC = \cup_k NC^k$.

Uniform boolean circuits are considered as a standard model of parallel computations and are widely studied in literature [Co]; size and depth of a circuit are the number of nodes and the length of the longest path from input to output nodes respectively. It is well known that NC is contained in the class P of problems solvable in polynomial sequential time; although it seems reasonable that the inclusion is proper, to prove this is an open problem. A central notion for investigating the structure of the hierarchy $\{NC^k\}$ and the complexity of problems that lie between NC^1 and NC^2 is the NC^1-reducibility [Co]. In particular we are interested in the following class:

Def.4.5- DET is the class of the functions belonging to NC^1 which are reducible to computing the determinant of a nxn matrix of n-bits integers.

Fact 4.4- $NC^1 \subseteq DET \subseteq NC^2$.

In [Co] it is proved that the problem of computing the inverse of a nxn matrix A with n-bits integer entries is in DET; furthermore, the problem remains in DET even if the matrix entries are polynomials with a fixed number of variables (or rational functions) over the rationals. Using this result, it is easily proved the following:

Fact 4.5- EPFSAA is in the class DET if the stochastic matrices associated to the automata have rational number as entries.

5. Generating functions and algebricity

In this section we study the problem of determining whether the function $F_L(x) = \sum a_n x^n$ is algebraic or not, $\{a_n\}$ being the counting sequence for a language L; in particular we consider counting sequences satisfying linear recurrence equations with polynomial coefficients:

P-recursive

$$\sum_{k=0}^{d} a_{n-k} P_k(n) = 0$$

As a matter of fact the interest in this type of conditon is motivated by the following observations:
1) the coefficients of every algebraic function verify an equation of this type [Com],
2) there exists a sufficiently large class of CFL's which have generating functions whose coefficients are given through such an equation [Fl].

Examples
5.1a) The language $L_1 = \{x \in \{a,b\}^* \mid n_a(x) = n_b(x)\}$, where $n_\sigma(x)$ is the number of occurrences of the symbol σ in the string x, has generating function

$$F_{L_1}(x) = \sum_{n=0}^{\infty} \binom{2n}{n} x^{2n} \quad ;$$

taking $\phi(x) = \sum_{n=0}^{\infty} \binom{2n}{n} x^n$ $(F_{L_1}(x) = \phi(x^2))$ then $na_n = 2(2n-1)a_{n-1}$

5.2a) the language $L_2 = \{x \in \{a,b,c\}^* \mid n_a(x) = n_b(x) = n_c(x)\}$ has generating function

$$F_{L_2}(x) = \sum_{n=0}^{\infty} \binom{3n}{n}\binom{2n}{n} x^{3n} = \sum_{n=0}^{\infty} \frac{3n!}{(n!)^3} x^{3n} \ ;$$

taking $\varphi(x) = \sum_{n=0}^{\infty} \frac{3n!}{(n!)^3} x^n$ $(F_{L_2}(x) = \varphi(x^3))$ then $n^2 a_n = 3(3n-1)(3n-2)a_{n-1}$

The problem of deciding algebricity for a single function seems to be very difficult: in the following we study a simpler problem which gives sometime an answer to the previous one.

Problem (space-algebricity):

Instance: a linear homogeneous recurrence equation with polynomial coefficients

$$\sum_{k=0}^{d} a_{n-k} P_k(n) = 0$$

Question: does the linear space of the generating functions $F(x) = \sum a_n x^n$,where $\{a_n\}$ satisfy the recurrence equation, contains only algebraic functions?

The following result allows us to transform the problem in that of studying equations on generating functions:

Fact 5.1- Given a linear homogeneous recurrence equation with polynomial coefficients, let $\{a_n\}$ be a solution (regardless of the initial conditions). Then $F(x)= \sum a_n x^n$ satisfies a linear homogeneous differential equation with polynomial coefficients, and viceversa.

The proof of this fact is constructive and it is based on two simple correspondences: the first one is that which exists between the displacement operator (on the space of the numerical successions) and the multiplication by a monomial x^k (on the space of the generating functions), the second one being that between the multiplication by n and the operator xD (D stands for the derivation operator).

Examples

5.1b) Taking $\phi(x) = \sum_{n=0}^{\infty} a_n x^n$ then $x\phi'(x) = \sum_{n=0}^{\infty} n a_n x^n$,

$$x(x\phi(x))' = \sum_{n=1}^{\infty} n a_{n-1} x^n \quad , \quad x\phi(x) = \sum_{n=1}^{\infty} a_{n-1} x^n .$$

The differential equation for example 5.1a is $(4x - 1)\phi(x)' + 2x\phi(x) = 0$

5.2b) Taking $\varphi(x) = \sum_{n=0}^{\infty} a_n x^n$ then $x(x\varphi'(x))' = \sum_{n=0}^{\infty} n^2 a_n x^n$,

$$x(x(x\varphi(x))')' = \sum_{n=1}^{\infty} n^2 a_{n-1} x^n.$$

Recalling example 5.2a we obtain $(27x^2 - x)\varphi''(x) + (54x - 1)\varphi'(x) + 6\varphi(x) = 0$.

It can be noted that the above theorem provides us with an easy proof of Comtet's theorem since all the algebraic functions satisfy differential equations with polynomial coefficients.

We recall here Klein's method for solving differential equations of the second order having polynomial coefficients; this is the starting point for an algorithm for testing algebricity.
The basic idea is that, given a differential equation of order m on the complex field with a fundamental system of integrals $w_1,\ldots,w_m$, the effect caused by a description of a closed path enclosing one or more singularities is to replace the system with one of the type

$$\begin{gathered} w_1' = a_{11}w_1 + \ldots + a_{1m}w_m \\ \cdot \\ \cdot \\ w_m' = a_{m1}w_1 + \ldots + a_{mm}w_m \end{gathered}$$

that is w' = S(w) where S is a linear substitution.
The set of linear substitutions (obtained by considering infinitely many contours passing through a fixed point) forms a group with respect to the composition. Moreover it is possible to prove the following

Fact 5.2- The group of linear substitutions associated with a linear differential equation (with polynomial coefficients) is finite <u>iff</u> the space of solutions only contains algebraic functions.

Klein gives a method for determining linear differential equations of the second order with algebraic integrals, by associating them with the finite groups of linear substitutions of two variables:

$$w_1' = \alpha w_1 + \beta w_2,\ w_2' = \gamma w_1 + \delta w_m$$
$$(w_1', w_2') = S(w_1, w_2).$$

In such a case he studies the finite groups of homographic substitutions by taking

$$s = \frac{w_1}{w_2},\ S = \frac{w_1'}{w_2'} = \frac{\alpha s + \beta}{\gamma s + \delta}$$

and he succeeds in computing the orders of the groups, identifying only five cases. His results can be summarized in the following theorem [Fo]:

Theor. 5.1- (Klein) Given a linear homogeneous differential equation of the second order with polynomial coefficients

$$A(z)\frac{d^2w}{dz^2} + B(z)\frac{dw}{dz} + C(z)w = 0,$$

his integrals are algebraic functions if and only if,taking $p(z) = \frac{B(z)}{A(z)}$, $q(z) = \frac{C(z)}{A(z)}$, one has:

1) $p(z) = \frac{1}{u}\frac{du}{dz}$ where u is an algebraic function of z;

2) the invariant $I(z) = q(z) - \frac{1}{4}p^2(z) - \frac{1}{2}\frac{dp}{dz}$ is either of the form

$$(*)\quad \frac{1}{4}\left[\frac{1-\frac{1}{\nu_2^2}}{Z^2} + \frac{1-\frac{1}{\nu_1^2}}{(Z-1)^2} + \frac{\frac{1}{\nu_1^2}+\frac{1}{\nu_2^2}-\frac{1}{\nu_3^2}-1}{Z(Z-1)}\right]\left(\frac{dZ}{dz}\right)^2 + \frac{1}{2}\{Z,z\}$$

or of the form

$$(**)\quad \frac{1}{4}\,\frac{1-\frac{1}{N^2}}{Z^2}\left(\frac{dZ}{dz}\right)^2 + \frac{1}{2}\{Z,z\}$$

where Z is a rational function of z, N is an integer, and$\{Z,z\} = \frac{Z'''}{Z'} - \frac{3}{2}\left(\frac{Z''}{Z'}\right)^2$ is the Schwarzian derivative. ν_1,ν_2,ν_3 are integers which are related to the orders of the groups of substitutions, their values being determined for each case.

case	ν_1	ν_2	ν_3
1	N	N	-
2	2	2	N
3	2	3	3
4	2	3	4
5	2	3	5

(*) for the cases 2,3,4,5
(**) for the case 1

6. An algorithm for the algebricity

We outline here an algorithm which is based on the results of the theorem 5.1. The main goal is to test the existence of a rational function Z(z) satisfying equations (*) or (**): this can be done by explicitly constructing Z(z).

Let Z(z) and Z(z) - 1 be in the form

$$Z(z) = A\,\frac{\prod_{i=1}^{m}(z-a_i)^{\alpha_i}}{\prod_{j=1}^{n}(z-c_j)^{\gamma_j}},\quad Z(z) - 1 = B\,\frac{\prod_{i=1}^{m}(z-b_i)^{\beta_i}}{\prod_{j=1}^{n}(z-c_j)^{\gamma_j}}\quad \text{A,B constants}$$

then by an analysis of the poles of the second order in the expression for I(z) (the invariant cannot have poles of order greater than 2) we obtain

$$(*)\ I(z) = \sum_{i=1}^{m} \frac{\frac{1}{4}\left(1 - \frac{\alpha_i^2}{v_2^2}\right)}{(z - a_i)^2} + \sum_{i=1}^{m} \frac{\frac{1}{4}\left(1 - \frac{\beta_i^2}{v_1^2}\right)}{(z - b_i)^2} + \sum_{j=1}^{n} \frac{\frac{1}{4}\left(1 - \frac{\gamma_j^2}{v_3^2}\right)}{(z - c_j)^2} - \sum_{k=1}^{p} \frac{\frac{1}{2}\tau_k + \frac{1}{4}\tau_k^2}{(z - t_k)^2} + \ldots$$

$$(**)\ I(z) = \sum_{i=1}^{m} \frac{\frac{1}{4}\left(1 - \frac{\alpha_i^2}{N^2}\right)}{(z - a_i)^2} + \sum_{j=1}^{n} \frac{\frac{1}{4}\left(1 - \frac{\gamma_j^2}{N^2}\right)}{(z - c_j)^2} - \sum_{k=1}^{p} \frac{\frac{1}{2}\tau_k + \frac{1}{4}\tau_k^2}{(z - t_k)^2} + \ldots$$

where t_k is a zero of $Z'(z)$ of multiplicity $\tau_k$which is not a repeated zero of $Z(z)$.

The algorithm works in the following steps:
1) it computes the coefficients of the poles of the second order in $I(z)$,
2) it guesses a correspondence between the values found (which must be rational numbers) and those in the symbolic expression at the right side of (*) or (**). If such a correspondence does not exist it halts with a negative answer (This means that there is no rational function $Z(z)$ satisfying the conditions of the theorem 5.1, hence the space of solutions contains at least one function which is not algebraic),
3) for each existing correspondence (it might not be unique - so it has to try for all the cases) - i.e. for each set of acceptable values for $\alpha_i,\beta_i,\gamma_j,\tau_k$ - it constructs an algebraic equation in the variables z,a_i,b_i,g_j,t_k by manipulating the equations (*) or (**),
4) it evaluates each algebraic equation at p+1 points where p is the degree of the equation in the variable z , obtaining a set of systems of algebraic equations in the variables a_i,b_i,g_j,t_k,
5) it tests whether at least one system admits a solution: if this is the case then the space of the solutions only contains algebraic integrals otherwise not.

Example

6.1a) For the equation $(27x^2 - x)\varphi''(x) + (54x - 1)\varphi'(x) + 6\varphi(x) = 0$ we have

$$p(x) = \frac{54x - 1}{27x^2 - x},\ q(x) = \frac{6}{27x^2 - x}.$$

The first condition is easily satisfied (take $u = 27x^2 - x$), then we compute the invariant $I(x)$.

$$I(x) = \frac{648x^2 - 24x + 1}{x^2(27x - 1)^2}$$

In this case the coefficients of the poles of the second order turn out to be 1, and this suffices to state that the space of the solutions doesn't contain only algebraic integrals, since for each case there is no way for an equation in $\alpha_i,\beta_i,\gamma_j$ to hold.

A preliminary version of this algorithm has been implemented in the muMATH system. At the moment it handles differential equations having polynomial coefficients of limited degree and it does not deal with the last step. The first limitation is of technical reason

(steps 1 through 4 have a complexity which is a polynomial in the maximum degree of the coefficients), while the second one is related to the inherent difficulty of the problem (from the complexity point of view): a possible solution could be that of using techniques based on Groebner bases which seem to work well in this case.
It should be noted that whenever the answer of the algorithm is negative, it might be the case that one of the integrals still be algebraic, so that we cannot decide for the ambiguity of the given language.

7. Generating functions and contour integration

In this section we give a technique which is sometimes useful to establish properties of generating functions. This method works well when there is a suitable relation between the coefficients of the generating function we want to study and those of another generating function being known.
In this setting we study generating functions in two variables, which are a natural extension of the univariate case, and we consider them as analytic functions. The basic result is in the following

Fact 7.1- Let $F(x,y) = \sum_{i,j} F_{ij}x^iy^j$ be an analytic function in the neighbourhood of the origin. Consider any function of the form

$$\phi(x) = \sum_i F_{hi\,i}x^{ki} \quad (h, k \text{ integers}),$$

then we have

$$\phi(x) = \frac{1}{2\pi i}\oint_c F(x^\alpha s^\beta, x^\gamma s^\delta)\frac{ds}{s}$$

where α, β, γ, δ are integers such that $h = \frac{-\delta}{\beta}$, $k = \alpha h + \gamma$, and c is a circle centered at the origin.

Example
7.1a) It is quite easy to show that the language $L_3 = \{x \in \{a,b,c\}^* \mid n_a(x) = n_b(x)\}$ has the generating function (in two variables)

$$F_{L_3}(x,y) = \sum_{i,j} F_{ij}x^iy^j = \sum_{i,j} F_{2i\,j}x^{2i}y^j = \frac{1}{\sqrt{(1-y)^2 - 4x^2}},$$

where $F_{2i\,j} = \#\{x \in \{a,b,c\}^* \mid n_a(x) = n_b(x) = i, n_c(x) = j\}$.

It is then immediate to see that $F_{L_2}(x) = \sum_{i=0}^{\infty} F_{2i\,i}x^{3i}$, therefore by fact 7.1 we have

$$F_{L_2}(x) = \frac{1}{2\pi i}\oint_c F_{L_3}\left(xs, xs^{-2}\right)\frac{ds}{s} = \frac{1}{2\pi i}\oint_c \frac{s}{\sqrt{\left(s^2-x\right)^2 - 4x^2s^6}}\,ds$$

and finally by taking $s = \sqrt{z}$ we obtain

$$F_{L_2}(x) = \frac{1}{4\pi i} \oint_c \frac{1}{\sqrt{(z-x)^2 - 4x^2 z^3}}\, dz$$

that is an elliptic integral of the first type. We can conclude that the generating function of L_2 is not algebraic since it has transcendental values at algebraic points (see for example [Sc]).

Acknowledgements
We warmly thank Alberto Bertoni for pointing us this topic and for stimulating discussions.
This work has been partially supported by MPI (Ministero della Pubblica Istruzione) funds.

References

[AR] IJ.J.Aalbersberg, G.Rozenberg, *Theory of traces*, tec. Rep. 86-16, Inst. of Appl.Math. and Comp. Sci. University of Leiden, 1986

[AH] IJ.J.Aalbersberg, H.J.Hogeboom, *Decision problems for regular trace languages*, Proc.14th ICALP, Lect.Not.Comp.Sci. 267, 251-259, Springer, 1987

[BBMS] A.Bertoni, M.Brambilla, G.Mauri, N.Sabadini, *An application of the theory of free partially commutative monoids: asymptotic densities of trace languages*, Lect. Not. Comp. Sci. 118, 205-215, Springer, 1981

[BMS1] A.Bertoni, G.Mauri, N.Sabadini, *Unambiguous regular trace languages*, Algebra Combinatorics and Logica in Computer Science, Colloquia Mathematica Societatis J. Bolyai, vol.42, 113-123, North Holland, Amsterdam 1985

[BPS] D.Bruschi, G.Pighizzini, N.Sabadini, *On the existence of the minimum asynchronous automaton and on decision problems for unambiguous regular trace languages*, Proc. STACS 88, Lect.Notes Comp.Sci. 294, pp.334-346, Springer, 1988

[BS1] A.Bertoni,N.Sabadini, *Algebricity of the generating function for context free languages*, Internal Rep. Dip.Scienze Informazione, Univ.Milano, 1985

[BS2] A.Bertoni,N.Sabadini, *Generating functions of trace languages*, Bulletin of EATCS 35, 1988

[Com] L.Comtet, *Calcul pratique des coefficient de Taylor d'une fonction algebrique*, Einsegnement Math.10, 267-270, 1964

[Co] S.A.Cook, *A taxonomy of problems with fast parallel algorithms*, Information and control 64, 2-22, 1985

[CS] N.Chomsky, M.Schuetzenberger, *The algebraic theory of context free languages*, Comp. Prog. and Formal Systems, North Holland,118-161, 1963

[DRS] P.Doubilet,G.C.Rota,R.Stanley, *On the foundations of combinatorial theory : the idea of generating functions*, VI° Berkeley Symp.Math.Stat.Prob.2, 267-318

[Fo] A.R.Forsyth, *Theory of differential equations*, Dover Publications, New York, 1959

[Fl] P.Flajolet, *Analytic models and ambiguity of context-free languages*, TCS.49, 283-309, 1987

[Go] J.Goldman, *Formal languages and enumeration*, Journal of Comb.Theory, series A 24, 318-338, 1978

[La] G.Lallement, *Semigroups and combinatorial applications*, J.Wiley and sons, New York, 1979

[Ma] A.Mazurkiewicz, *Concurrent program schemes and their interpretations*, DAIMI Rep.PB-78, Aarhus Univ., 1977

[Mo] H.K.Molloy, *On the integration of delay and throughput measures in distributed processing modes*, Ph.D. Thesis, Univ. of California, Los Angeles, 1981

[Pa] A.Paz, *Introduction to probabilistic automata*, Academic Press, New York London, 1971

[Sa] A.Salomaa, *Formal languages*, Academic Press, New York London, 1973

[Sc] T.Schneider, *Introduction aux nombres transcendants*, Gauthier-Villars, Paris, 1959

[St] P.Stanley, *Generating functions*, in Studies in Combinatorics, vol.17, ed. G.C.Rota, 100-141,1978

[Zi] W.Zielonka, *Notes on asynchronous automata*, RAIRO Inf.Théor.vol. 21 n.2, 99-135, 1987