An Efficient Representation for Large Arrays of Rational Expressions*

Eugene W. Stark
Department of Computer Science
State University of New York at Stony Brook
Stony Brook, NY 11794 USA
stark@cs.sunysb.edu

October 1, 2010

Abstract

The paper describes a method of representing an n-dimensional array of rational expressions as an $(n + 1)$-dimensional array of scalars and shows how this representation readily supports the implementation of various common array operations. A feature of the representation is that it can take advantage of memory-saving schemes for encoding large, sparse scalar arrays using multi-terminal binary decision diagrams (MTBDDs). The representation involves taking partial fraction expansions of the entries of the array; refining the factorizations of the denominator polynomials as required rather than presupposing the existence of a complete factorization at the outset.

1 Introduction

A *rational expression* over a coefficient field $\mathcal{K}$ is an element of the field of fractions of the ring of polynomials over $\mathcal{K}$. More concretely, a rational expression is an equivalence class $\ulcorner p/q \urcorner$ of *formal quotients* p/q of polynomials, with $q \neq 0$, under the equivalence relation that relates p/q and p'/q' whenever $pq' = qp'$. Computing with rational expressions can be accomplished [Hor72, Col75] by using as canonical representations the formal quotients p/q such that q is monic and p and q are coprime. These representations are manipulated using the familiar rules for fractions.

If it is desired to compute with arrays (vectors, matrices, 3D-arrays, *etc.*) of rational expressions, then this can be done using standard array storage techniques with rational expressions as the elements of the arrays. Just as for arrays of ordinary numeric

*This research was supported in part by the National Science Foundation under Grant CCR-9988155 and the Army Research Office under Grants DAAD190110003 and DAAD190110019. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, the Army Research Office, or other sponsors.

values, either "dense" representations, in which every element is represented explicitly, or "sparse" representations, in which zero elements are left implicit, are possible. However, for some applications there are drawbacks inherent in this way of representing arrays of rational expressions. For example, consider what has to be done in order to compute the sum of two n-element vectors R and R' of rational expressions. For each i with $1 \leq i \leq n$ we must compute the sum of the ith elements $R_i = \ulcorner p_i/q_i \urcorner$ and $R'_i = \ulcorner p'_i/q'_i \urcorner$. This calculation could be done, say, by finding a common multiple $m_i = q_i q'_i$ of q_i and q'_i, computing the sum $r_i = p_i q'_i + p'_i q_i$, and then cancelling common factors to obtain a canonical representation for the rational expression $\ulcorner r_i/m_i \urcorner$. In case many entries of R and R' have the same denominators, similar calculations would have to be repeated many times. One way of avoiding some of this repeated work would be to represent a vector R of rational expressions as a pair (P, q), where P is a vector of the numerator polynomials and q is a single denominator polynomial common to all the entries. In this case, to add (P, q) and (P', q') we need only compute the common denominator qq' once. This possibly saves some work, but it is of course still necessary to multiply each element of P by q' to obtain Pq', multiply each element of P' by q to obtain $P'q$, form the sum $Pq' + P'q$, and then cancel common factors from $Pq' + P'q$ and qq' to obtain the result. In addition, placing all entries of a large vector over a single common denominator would tend to produce numerator polynomials with large coefficients and unnecessarily high degrees.

Another disadvantage of using a standard array representation with rational expressions as elements is that this approach cannot take full advantage of some of the techniques that have been developed for obtaining and manipulating highly compact representations for huge arrays having either highly sparse or repetitive structure. For example, such arrays can arise when a large system (*e.g.* a Markov chain represented by its transition matrix) is specified as a composition of loosely interacting component subsystems. In this case, composing the transition matrices of the components to obtain the transition matrix for the large system is an operation that can be described in terms of the Kronecker (tensor) product of matrices. In practice, this often results in a transition matrix for the composite system that, although huge in dimension (the dimensions of the matrices for the component subsystems combine multiplicatively), has a highly repetitive structure due to the loose interaction between the components. Also, if there is just one initial state of interest, then often many of the states of the composite system are unreachable, which leads to a highly sparse system representation once the irrelevant entries have been zeroed.

Multiterminal binary decision diagrams (MTBDDs, also known as *algebraic binary decision diagrams* or ABDDs) [BFG+93, CFM+97], are data structures that have been exploited, for example, to obtain highly compact representations of huge transition matrices obtained from system descriptions. An MTBDD is a decision graph whose nodes represent functions $f : \{0, 1\}^d \to V$ taking sequences of boolean values to elements of an arbitrary set V of data values. The graph is organized into levels that correspond to the length d of the argument sequences for the functions represented. For example, if $d = 0$ (*i.e.* the function f takes no arguments and is simply an element v of V), then the corresponding node in the graph is a leaf node (at level 0) which is labeled by v. If $d > 0$, then the node for f, which occurs at level d of the graph, has two outgoing edges (labeled by 0 and 1) leading to child nodes at level $d - 1$. These child nodes represent the functions $f_0 : \{0,1\}^{d-1} \to V$ and $f_1 : \{0,1\}^{d-1} \to V$ obtained, respectively, from f by fixing the first element of the argument sequence:

$$f_0(\sigma) = f(0,\sigma) \qquad\qquad f_1(\sigma) = f(1,\sigma).$$

MTBDDs are are maintained in *reduced* form, which means that there is at most one node in a graph representing any particular function f. This leads to the possibility of *structure sharing* between the various descendants of of a node, which makes it possible for a function f to have a very compact representation in some cases. In particular, this can happen if the values of f occur in some sort of regular pattern and the number of distinct values taken on by f is small compared to the size 2^d of its domain. Manipulations of MTBDDs are programmed in a recursive style, in which operations on higher-level nodes are reduced to the computation of the same operations on lower-level nodes, and in which caching (which exploits reduceness in an essential way) is used to avoid exponential explosion that would otherwise result due to repeated treatment of nodes accessible via multiple paths in the diagram.

MTBDDs can provide a compact representation for sparse and regularly structured arrays, by regarding an array containing 2^d elements drawn from V as a function $f : \{0,1\}^d \to V$. Many important operations on arrays can then be programmed in the recursive style required for efficient manipulation of MTBDDs. In fact, this type of representation has been exploited very successfully by tools [HMKS99, KNP02a, KNP02b] that can build representations for transition matrices of huge Markov chains and to solve the associated systems of linear equations that define such quantities as steady-state probabilities. The Kronecker product of arrays having dimensions that are powers of two can be performed highly efficiently on MTBDD representations, by a construction that essentially amounts to concatenation of decision diagrams. Matrix/vector and matrix/matrix multiplication can also be performed on MTBDD representations, in time quadratic in the number of nodes of the argument diagrams (assuming an unbounded cache in which to store already-computed results). Matrix/vector multiplication, in particular, is the main operation needed in order to solve systems of linear equations using iterative methods.

In this paper, we describe a technique that permits an array of rational expressions to be represented using a scalar array having one additional dimension. That is, a one-dimensional array (*i.e.* a vector) of rational expressions would be represented using a matrix of scalars, a two-dimensional array (*i.e.* a matrix) would be represented using a three-dimensional array of scalars, and so on. In brief, the rational expressions making up the original array are represented as linear combinations $\sum_{i=1}^{m} \sum_{j=1}^{d_i-1} a_{i,j} \ulcorner p_{i,j}/q_i \urcorner$, of *basis expressions* $\ulcorner p_{i,j}/q_i \urcorner$, where $\langle q_1, q_2, \ldots, q_m \rangle$ is a pairwise coprime sequence of *denominator polynomials* with $\deg(q_i) = d_i$, and for each i the sequence $\langle p_{i,1}, p_{i,2}, \ldots, p_{i,d_i} \rangle$ is a linearly independent set of *numerator polynomials*. For each element of the original array, the corresponding coefficient vector $\langle a_{i,j} : 1 \leq i \leq m, 1 \leq j \leq d_i \rangle$ is stored using the additional dimension. The coefficients themselves are obtained by *partial fraction expansion* of the original array elements. The idea of using partial fraction expansion as a representation for rational expressions has been proposed recently by Fateman [Fat06], however he assumes the use of a complete expansion with irreducible denominator polynomials. Here, we do not make any explicit effort to maintain the entries in fully expanded form, which would require factoring completely the denominator of each array entry. Instead, what we do is to take a "lazy" approach in which additional factorizations are discovered and introduced as needed during operations such as addition, when two matrices to be added must be "transformed to a common basis" before the result can be computed.

Besides making it possible to employ existing compact storage representations such as MTBDDs, the representation we describe permits some operations that would otherwise require expensive element-by-element processing of an array to be performed just on the basis elements $\ulcorner p_{i,j}/q_i \urcorner$. For example, multiplying each element of an array by a scalar c can be performed simply by replacing each $p_{i,j}$ by $cp_{i,j}$. Another example is *translation by a scalar* c, which replaces each element r of an array by the rational expression $r(x+c)$ obtained by substituting $x+c$ for the indeterminate x. This operation preserves partial fraction expansions and can be performed simply by applying the translation operation to each basis element $\ulcorner p_{i,j}/q_i \urcorner$. If the number of basis elements is small compared to the total number of entries in the array, then a significant savings in time is realized.

The techniques in this paper were developed for use in manipulating systems descriptions represented using the probabilistic input/output automaton (PIOA) model [WSS97, SS98], and implemented in the context of the "PIOATool" [ZCS03] system. In the PIOA model, transition matrices whose entries are rational expressions rather than scalars arise in the representation of "open" systems, which are those whose environments have not yet been completely specified. The operation of composing additional components with an open system involves: (1) applying translation operations to the transition matrix of the system; and (2) forming the Kronecker product of the resulting translated matrices with scalar matrices describing the new component. In PIOATool, the matrices of rational expressions are represented, and the required operations on matrices implemented, using an underlying MTBDD-based representation for scalar arrays.

The remainder of this paper is organized as follows. In Section 2 we recall some necessary concepts and notation. In Section 3 we introduce the idea of a "basis of formal quotients" (BFQ), which is a sequence of rational expressions having certain independence properties. The set of all linear combinations of the elements of a BFQ is a finite-dimensional subspace of the space of all rational expressions, and once a BFQ has been specified each element of this subspace can be specified by giving a corresponding coordinate vector. In Section 4 we show how to represent an array of rational expressions as an array of coordinate vectors with respect to a specified BFQ, and in Section 5 we discuss how to carry out various common operations in terms of this representation.

2 Preliminaries

Polynomials Let $\mathcal{K}$ be a field. A *polynomial* p over $\mathcal{K}$ is a finite formal sum

$$p = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0,$$

where the coefficients a_i are elements of $\mathcal{K}$ and the x^i are formal powers of the "indeterminate" x. The *degree* of a nonzero polynomial p is the largest value i for which the coefficient a_i of x^i is nonzero. By convention, we define the degree of the zero polynomial to be 0. We use $\deg(p)$ to denote the degree of polynomial p. A polynomial p of degree n is *monic* if the coefficient of x^n is 1.

Polynomials over $\mathcal{K}$ form a commutative ring, which we denote by $\mathcal{K}[X]$, under the usual definitions for the arithmetic operations. We say that polynomial q *divides* polynomial p, and we write $q|p$, if there exists a polynomial r such that $qr = p$. In this case, q and r are called *factors* of p. The *nontrivial* factors of q are those other than 1 and q. A polynomial is *irreducible* if it is not equal to 1 and it has no nontrivial factors. The set of *irreducible factors* of q is the set of all irreducible polynomials that divide q. We write $p \lesssim q$ if every irreducible factor of p is also an irreducible factor of q (though not necessarily with any particular relationship between the multiplicities), and we write $p \sim q$ if $p \lesssim q$ and $q \lesssim p$; that is, when p and q have the same sets of irreducible factors. Polynomials p and q are *coprime* if they are not equal and have no nontrivial common factors. The commutative ring $\mathcal{K}[X]$ is a *Euclidean domain*, which entails a number of properties we shall use:

- Every polynomial other than 0 or 1 can be expressed uniquely as a product of powers of its irreducible factors.
- Every two polynomials have a *greatest common divisor* (GCD), which can be computed by the Euclidean algorithm, as well as a *least common multiple* (LCM). A GCD of two polynomials is unique up to scaling by element of K; thus if we impose the requirement that a GCD be monic, then the GCD is uniquely determined. The same holds for the LCM.
- The *Bézout lemma* holds: If q and q' are coprime then there exist polynomials p and p' such that $p'q + pq' = 1$.

We use $\mathcal{K}_{<n}[X]$ to denote the set of polynomials of degree strictly less than n. Note that, besides being a commutative ring, $\mathcal{K}[X]$ has the structure of a $\mathcal{K}$-vector space, with each $\mathcal{K}_{<n}[X]$ as an n-dimensional subspace. A *basis* for $\mathcal{K}_{<n}[X]$ is, as usual, a subset B of $\mathcal{K}_{<n}[X]$ that is linearly independent and spans $\mathcal{K}_{<n}[X]$. The *natural basis* for $\mathcal{K}_{<n}[X]$ is the sequence consisting of the n powers:

$$\langle x^{n-1}, x^{n-2}, \ldots, x^2, x, 1\rangle$$

of the indeterminate x, listed in decreasing order of their exponents.

Rational Expressions As stated at the outset, a *rational expression* over $\mathcal{K}$ is an element of the field $\mathcal{K}[[X]]$ of fractions over $\mathcal{K}[X]$. Each rational expression is an equivalence class $\ulcorner p/q \urcorner$ of *formal quotients* p'/q' of polynomials, with $q' \neq 0$, under the equivalence relation that relates p/q and p'/q' whenever $pq' = qp'$. Clearly, if $r \neq 0$ then $\ulcorner p/q \urcorner = \ulcorner pr/qr \urcorner$, hence rational expressions are preserved by the usual operations of "introduction or cancellation of nonzero common factors". It follows from this and the existence of GCDs that each rational expression has a unique canonical representation as a formal quotient p/q, where q is nonzero and monic and p and q are coprime. If p/q is any formal quotient, then we refer to p as the *numerator* and q as the *denominator*. The *numerator* and *denominator* of a rational expression $\ulcorner p/q \urcorner$ are the numerator and denominator of its canonical representative.

In this paper, we shall for the most part only consider *proper* rational expressions, which are those rational expressions $\ulcorner p/q \urcorner$ that satisfy $\deg(p) < \deg(q)$. This is not a significant limitation, because an arbitrary rational expression can always be represented as the sum of a polynomial and a proper rational expression. In the sequel the term "rational expression" shall be used to mean "proper rational expression" without further comment. The *degree* of a proper rational expression $\ulcorner p/q \urcorner$ is defined to be the degree of the polynomial q. Let $\mathcal{K}_{<n}[[X]]$ denote the subset of $\mathcal{K}[[X]]$ consisting of all rational expressions of degree strictly less than n. Besides being a field, the set $\mathcal{K}[[X]]$ has the structure of a $\mathcal{K}$-vector space and $\mathcal{K}_{<n}[[X]]$ is an n-dimensional subspace of $\mathcal{K}[[X]]$.

3 Bases of Formal Quotients

Suppose, for $1 \leq i \leq m$, that p_i and q_i are nonzero polynomials, such that q_i is monic and $\deg(p_i) < \deg(q_i)$ for $1 \leq i \leq m$. Then each formal quotient p_i/q_i represents a nonzero proper rational expression $\ulcorner p_i/q_i \urcorner$ for $1 \leq i \leq m$. We will call such a sequence $S = \langle p_1/q_1, p_2/q_2, \ldots, p_m/q_m \rangle$ a *sequence of formal quotients* (SFQ, for short). If S is a SFQ, then we write $|S|$ to denote the length m of the sequence S, and we write $\mathrm{span}(S)$ to denote the subspace of $\mathcal{K}[[X]]$ spanned by the corresponding sequence of rational expressions $\langle \ulcorner p_1/q_1 \urcorner, \ulcorner p_2/q_2 \urcorner, \ldots \ulcorner p_m/q_m \urcorner \rangle$. A *coordinate vector with respect to S* for a rational expression $e \in \mathrm{span}(S)$ is a sequence $\langle a_1, a_2, \ldots, a_m \rangle$ of elements of $\mathcal{K}$, such that $e = \sum_{i=1}^{m} a_i \ulcorner p_i/q_i \urcorner$. If the sequence $\langle \ulcorner p_1/q_1 \urcorner, \ulcorner p_2/q_2 \urcorner, \ldots \ulcorner p_m/q_m \urcorner \rangle$ of rational expressions is linearly independent (hence forms a basis for $\mathrm{span}(S)$), then we will call S a *basis of formal quotients* (BFQ, for short). If S is a BFQ, then clearly every rational expression $e \in \mathrm{span}(S)$ has a unique coordinate vector $\langle a_1, a_2, \ldots, a_m \rangle$ with respect to S.

Call a SFQ $S = \langle p_1/q_1, p_2/q_2, \ldots, p_m/q_m \rangle$ *homogeneous* if all the polynomials q_i are identical. For an arbitrary SFQ S and polynomial q, we use $S(q)$ to denote the homogenous subsequence of S that consists of all the p_i/q_i in S for which $q_i = q$. We call S *segregated* (resp. *strongly segregated*) if it has the following property:

- If p/q and p'/q' are any two elements of S, then either q and q' are coprime or else $q \sim q'$ (resp. $q = q'$).

Clearly, if e is an arbitrary rational expression, and if p/q is an arbitrary representative of e with $a \in \mathcal{K}$ the leading coefficient of q, then the singleton sequence $S = \{(a^{-1}p)/(a^{-1}q)\}$ is a homogeneous, strongly segregated SFQ such that $e \in \mathrm{span}(S)$. So it is always possible to view individual rational expressions as elements of $\mathrm{span}(S)$ for some strongly segregated SFQ S.

The following result is the main tool underlying our representation for arrays of rational expressions. It enables us to take an arbitrary SFQ S and obtain a strongly segregated SFQ S' whose span includes the span of S.

Theorem 1. *Suppose S is a SFQ. Then there exists a strongly segregated SFQ S' such that* $\operatorname{span}(S) \subseteq \operatorname{span}(S')$*. Moreover, there is an algorithm that, given S, computes S' and in addition computes the matrix M of a linear transformation that takes each coordinate vector of a rational expression e with respect to S to a coordinate vector of that same rational expression e with respect to S'.*

Proof. The proof proceeds in two steps:

1. First, we show how to compute from S a segregated SFQ S'' such that $\operatorname{span}(S) \subseteq \operatorname{span}(S'')$. The computation proceeds by repeatedly applying a *segregation procedure* to eliminate instances of formal quotients p_i/q_i and p_j/q_j where q_i and q_j are neither coprime nor have the same sets of irreducible factors. The segregation procedure is described in Section 3.1 below.

2. Second, we show how to compute from S'' a strongly segregated SFQ S' such that $\operatorname{span}(S') = \operatorname{span}(S'')$. This computation proceeds by applying a *homogenization procedure*, to replace pairs of formal quotients p_i/q_i and p_j/q_j, where $q_i \sim q_j$, by new pairs p'_i/q and p'_j/q, where $q'_i \sim q \sim q'_j$. The homogenization procedure is described in Section 3.2 below.

□

We now turn to a detailed description of the segregation and homogenization procedures.

3.1 Segregation

In this section we show that, given a SFQ S, there exists a segregated SFQ S' such that $\operatorname{span}(S) \subseteq \operatorname{span}(S')$. The main result is Lemma 2, which gives a *segregation procedure* that, if applied repeatedly to a SFQ, will eventually bring about the segregation property. The segregation procedure makes use of *partial fraction expansion*, which we discuss first.

Lemma 1 (Partial Fraction Expansion)**.** *Suppose r is a rational expression, represented by the formal quotient p/q. Suppose further that we have a nontrivial factorization $q = q_1 q_2$, where q_1 and q_2 are coprime. Then there exist polynomials p_1 and p_2 such that*

$$\ulcorner p/q \urcorner = \ulcorner p_1/q_1 \urcorner + \ulcorner p_2/q_2 \urcorner .$$

Proof. If q_1 and q_2 are coprime, then by the Bézout lemma, there exist polynomials p'_1 and p'_2 such that $1 = p'_2 q_1 + p'_1 q_2$. Then taking $p_1 = pp'_1$ and $p_2 = pp'_2$ gives $p = p_2 q_1 + p_1 q_2$. It follows that $\ulcorner p/q \urcorner = \ulcorner p_1/q_1 \urcorner + \ulcorner p_2/q_2 \urcorner$, as asserted. □

Lemma 2. *Suppose S is a SFQ. Then there exists a segregated SFQ S' such that* $\operatorname{span}(S) \subseteq \operatorname{span}(S')$.

Proof. Suppose $S = \langle p_1/q_1, p_2/q_2, \ldots, p_m/q_m \rangle$. If S is segregated, then there is nothing to prove. Otherwise, there exist i and j such that q_i and q_j are neither coprime nor do we have $q_i \lesssim q_j$. Then q_i and q_j have some irreducible factor in common, but q_i also has an irreducible factor that is not a factor of q_j. We may therefore write $q_i = rs$, where r is a product of powers of those irreducible factors that q_i and q_j have in common, and s is a product of powers of those irreducible factors of q_i that are not also factors of q_j. Clearly r and s are coprime, and we may arrange that both r and s are monic. By Lemma 1, we have a partial fraction expansion

$$\ulcorner p_i/q_i \urcorner = \ulcorner p_r/r \urcorner + \ulcorner p_s/s \urcorner.$$

Let S' be obtained from S by replacing the formal quotient p_i/q_i by the two-element sequence $p_r/r, p_s/s$. Clearly $\mathrm{span}(S) \subseteq \mathrm{span}(S')$. This procedure is repeated until it is no longer possible to perform it, at which point a segregated SFQ is obtained.

To see that the above procedure must eventually terminate, we associate with each SFQ $\langle p_1/q_1, p_2/q_2, \ldots, p_m/q_m\rangle$ the finite multiset of nonnegative integers

$$\{\deg(q_1), \deg(q_2), \ldots, \deg(q_m)\}.$$

Define a lexicographic ordering that compares two such multisets by first comparing the multiplicities of the largest value occuring in either of the two multisets, then comparing the multiplicities of the second-largest value, and so on. Each iteration of the segregation procedure replaces a formal quotient p_i/q_i, by two new formal quotients p_r/r and p_s/s, where $\deg(r) < \deg(q_i)$ and $\deg(s) < \deg(q_i)$, hence produces a strict decrease with respect to the lexicographic ordering. Since the ordering is clearly well-founded, termination is ensured. □

As an example of the segregation procedure, consider the SFQ $S = \langle p_1/q_1, p_2/q_2\rangle$, where

$$\frac{p_1}{q_1} = \frac{1}{x^3 - 5x^2 + 8x - 4} \qquad \frac{p_2}{q_2} = \frac{1}{x^2 - 5x + 6}$$

Now q_1 and q_2 have the common factor $x-2$, which induces the nontrivial factorization $q_1 = (x-1)(x-2)^2$. Applying partial fraction expansion to $\ulcorner p_1/q_1 \urcorner$, we obtain

$$\ulcorner p_1/q_1 \urcorner = \ulcorner p_3/q_3 \urcorner + \ulcorner p_4/q_4 \urcorner$$

where

$$\frac{p_3}{q_3} = \frac{1}{x-1} \qquad \frac{p_4}{q_4} = \frac{-x+3}{(x-2)^2}.$$

We replace p_1/q_1 by $p_3/q_3, p_4/q_4$. Next, q_2 and q_4 have the common factor $x-2$, which yields the nontrivial factorization $q_2 = (x-2)(x-3)$. Applying partial fraction expansion to $\ulcorner p_2/q_2 \urcorner$ gives

$$\ulcorner p_2/q_2 \urcorner = \ulcorner p_5/q_5 \urcorner + \ulcorner p_6/q_6 \urcorner,$$

where

$$\frac{p_5}{q_5} = \frac{-1}{x-2} \qquad \frac{p_6}{q_6} = \frac{1}{x-3}.$$

so we replace p_1/q_2 by $p_5/q_5, p_6/q_6$, resulting in the SFQ

$$S' = \langle \frac{1}{x-1}, \frac{-x+3}{(x-2)^2}, \frac{-1}{x-2}, \frac{1}{x-3}\rangle.$$

Since all pairs (q, q') of denominators are either coprime or satisfy $q \sim q'$, the SFQ S' is segregated and the procedure terminates.

The procedure given in the proof of Lemma 2 requires that we be able to obtain a nontrivial factorization $q_i = rs$, where r has only irreducible factors of q_i that also occur in q_j and s has only irreducible factors of q_i that do not also occur in q_j. Although this could be accomplished by completely factoring q_i and q_j into irreducible factors (actually, only a squarefree factorization would be required), this is unnecessary, as the next results show.

Lemma 3. *Let p and q be nonzero polynomials, with p monic. Then the following are equivalent:*

1. $\gcd(p, q^k) = p$ *for some* $k > 0$.
2. $p|q^k$ *for some* $k > 0$.
3. $p \lesssim q$.

Proof.

(1) implies (2). Obvious.

(2) implies (3). If $p|q^k$, then every irreducible factor of p is also a factor of q, hence $p \lesssim q$.

(3) implies (1). If $p \lesssim q$, then every irreducible factor of p is also a factor of q, hence taking k to be the greatest multiplicity of any of the irreducible factors of p and using the assumption that p is monic gives $\gcd(p, q^k) = p$. □

Lemma 4. *Let p and q be arbitrary nonzero polynomials. Then there exists $k > 0$ such that* $\gcd(p, q^{k+1}) = \gcd(p, q^k)$.

Proof. It suffices to take k to be the greatest multiplicity of any of the irreducible factors of of p. □

The preceding lemmas show that we can carry out the segregation procedure in the proof of Lemma 2 as follows: At each iteration, we search systematically through all pairs (i, j) with $1 \le i, j \le m$ and $i \ne j$, looking for a pair for which $\gcd(q_i, q_j) \ne 1$. If there are no such pairs, all q_i and q_j are coprime for $i \ne j$ and the algorithm terminates. If (i, j) is such a pair, we next obtain a positive integer k such that $\gcd(q_i, q_j^{k+1}) = \gcd(q_i, q_j^k)$. Lemma 4 shows that such a k always exists, hence we can find one by searching the positive integers in increasing order. Let $g = \gcd(q_i, q_j^k)$. There are now two possibilities:

1. $g = q_i$. In this case, $q_i \lesssim q_j$ by Lemma 3, so we go on to the next pair (i, j).
2. $g \ne q_i$. In this case, q_i has some irreducible factor that does not also divide q_j. Since we have already ensured that $g \ne 1$, we obtain a nontrivial factorization $q_i = g(q_i/g)$, as required by the segregation procedure.

In our example, when considering p_1/q_1 and p_2/q_2, we first compute $\gcd(q_1, q_2) = x - 2$, then $\gcd(q_1, (q_2)^2) = (x-2)^2$, and then $\gcd(q_1, (q_3)^3) = (x-2)^2 = \gcd(q_1, (q_2)^2)$, so we take $g = (x-2)^2$ and obtain the factorization $q_1 = (q_1/g)g = (x-1)(x-2)^2$.

It is also possible, at each application of the segregation procedure, to compute a corresponding change of coordinates matrix. In particular, suppose S' is obtained from S by replacing the formal quotient p_i/q_i by the sequence $p_r/r, p_s/s$ as in the proof of Lemma 2. Let M be the $(|S| \times |S'|)$-dimensional matrix having the following block form:

$$M = \begin{pmatrix} I_{i-1} & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & I_{m-i} \end{pmatrix}$$

where I_{i-1} is the $(i-1)$-dimensional identity matrix, I_{m-i} is the $(m-i)$-dimensional identity matrix, and the zeroes denote zero matrices of suitable dimensions. Clearly, M is the matrix of a change of coordinates (applied on the right of row vectors) from S to S'. Such matrices arising in each iteration of the algorithm can be composed by multiplication to obtain a final overall change of coordinates matrix.

In our example, the overall change of coordinates matrix is computed as follows:

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

To make the segregation procedure completely algorithmic, it remains to be shown how, given p and the factorization $q = rs$, to compute the polynomials p_r and p_s in the partial fraction expansion $\ulcorner p/q \urcorner = \ulcorner p_r/r \urcorner + \ulcorner p_s/s \urcorner$. Although more efficient algorithms for doing this are known (*e.g.* [Xin04]) for concreteness we show how, with one matrix inversion, to obtain matrices M'_r and M'_s that can be used to obtain the coefficients of p_r and p_s, respectively from the coefficients of p. Thus, once matrices M'_r and M'_s have been computed for a particular factorization $q = rs$, we can use these same matrices to obtain partial fraction expansions for all p_k/q_k with $q_k = q$. The following result makes this precise.

Lemma 5. *Suppose $q = rs$, where r and s are monic and coprime. Let*

$$T_r : \mathcal{K}_{<\deg(s)}[X] \to \mathcal{K}_{<\deg(q)}[X] \qquad \text{and} \qquad T_s : \mathcal{K}_{<\deg(r)}[X] \to \mathcal{K}_{<\deg(q)}[X]$$

be the linear transformations "multiply by r" and "multiply by s", respectively. Then the linear transformation:

$$T : \mathcal{K}_{<\deg(r)}[X] \oplus \mathcal{K}_{<\deg(s)}[X] \to \mathcal{K}_{<\deg(q)}[X],$$

defined by $T(p_r \oplus p_s) = p_r s + p_s r$, is nonsingular. It therefore has an inverse T', which can be expressed as the direct sum $T'_s \oplus T'_r$ of transformations T'_r and T'_s that satisfy $p = (T'_r p)s + (T'_s p)r$, hence $\ulcorner p/q \urcorner = \ulcorner (T'_r p)/r \urcorner + \ulcorner (T'_s p)/s \urcorner$ for any polynomial $p \in \mathcal{K}_{<\deg(q)}[X]$. Moreover, there is an algorithm for computing the matrices of T'_r and T'_s with respect to the natural bases for $\mathcal{K}_{<\deg(q)}[X]$, $\mathcal{K}_{<\deg(r)}[X]$, and $\mathcal{K}_{<\deg(s)}[X]$, *Proof.* Let $d_r = \deg(r)$, $d_s = \deg(s)$, and $d_q = \deg(q) = \deg(r) + \deg(s)$. Since r and s are coprime, by the Bézout lemma, there exist polynomials r' and s' such that $1 = rs' + sr'$. Thus, for any polynomial $p \in \mathcal{K}_{<d_q}[X]$ we have $p = r(s'p) + s(r'p)$. In other words, for any polynomial $p \in \mathcal{K}_{<d_q}[X]$ there exist polynomials p_s and p_r such that $p = rp_s + p_r s$. Since $rp_s + sp_r = r(p_s - ts) + (p_r + rt)s$ for any polynomial t, we may always arrange that $p_r \in \mathcal{K}_{<d_r}[X]$ by choosing t so that rt cancels the coefficients of degree d_r or higher in p_r. Moreover, p_r and p_s are uniquely determined by p if we require that $p_r \in \mathcal{K}_{<d_r}[X]$ (and necessarily also $p_s \in \mathcal{K}_{<d_s}[X]$). For, suppose we have $rp_s + sp_r = p = rp'_s + sp'_r$, where $p_r, p'_r \in \mathcal{K}_{<d_r}[X]$ and $p_s, p'_s \in \mathcal{K}_{<d_s}[X]$. Then $r(p_s - p'_s) = (p'_r - p_r)s$, hence $r|(p'_r - p_r)$. But since 0 is the only polynomial of degree strictly less than d_r that r divides, it must be that $p'_r - p_r = 0$; in other words, $p'_r = p_r$. Once we know $p'_r = p_r$, it follows immediately that $p_s = p'_s$.

From the reasoning of the preceding paragraph, we may conclude that the linear transformation T is a bijection. It is therefore nonsingular, and when expressed in the form $T'_r \oplus T'_s$, the inverse transformation T' satisfies $p = (T'_r p)s + (T'_s p)r$ for all $p \in \mathcal{K}_{<d_q}[X]$.

We turn now to the computation of the matrices of the transformations T'_r and T'_s. Suppose $r = a_{d_r}x^{d_r} + a_{d_r-1}x^{d_r-1} + \ldots + a_1 x + a_0$. With respect to the natural bases for $\mathcal{K}_{<d_s}[X]$ and $\mathcal{K}_{<d_q}[X]$, the matrix of T_r (acting on the right of row vectors) is the $(d_s \times (d_r + d_s))$-dimensional matrix M_r whose k-th row (starting from 1) is obtained by taking the taking the $(d_r + d_s)$-dimensional row vector

$$(a_{d_r}\ a_{d_r-1}\ \ldots\ a_1\ a_0\ 0\ 0\ \ldots\ 0)$$

and shifting it to the right by $k - 1$ columns, filling with zeroes at the left. The matrix of T_s is a $(d_r \times (d_r + d_s))$-dimensional matrix M_s constructed similarly. The transformation T is represented by the block matrix (known as the *Sylvester Matrix*)

$$\begin{pmatrix} M_r \\ M_s \end{pmatrix}.$$

Inverting this matrix yields the matrix of T', which can be written in the form

$$\begin{pmatrix} M'_s & M'_r \end{pmatrix}.$$

to obtain the matrices M'_s and M'_r of the transformations T'_s and T'_r, respectively. Thus, given the coefficient vector of p with respect to the natural basis for $\mathcal{K}_{<\deg(q)}[X]$, we can obtain the coefficient vectors of p_r and p_s simply by multiplying by M'_r and M'_s, respectively. □

In our running example, to compute the partial fraction expansion

$$\ulcorner 1/(x-1)(x-2)^2 \urcorner = \ulcorner p_r/(x-1) \urcorner - \ulcorner p_s/(x-2)^2 \urcorner$$

we form the matrices

$$M_r = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{pmatrix} \qquad M_s = \begin{pmatrix} 1 & -4 & 4 \end{pmatrix}$$

and

$$T = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & -4 & 4 \end{pmatrix}$$

whose inverse is

$$T' = \begin{pmatrix} 0 & 4 & 1 \\ -1 & 4 & 1 \\ -1 & 3 & 1 \end{pmatrix}$$

giving

$$M'_s = \begin{pmatrix} 0 & 4 \\ -1 & 4 \\ -1 & 3 \end{pmatrix} \qquad M'_r = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

so that the coefficient vector of p_r is $(0\ 0\ 1)M'_r$ or (1) (i.e. $p_r = 1$) and the coefficient vector of p_s is $(0\ 0\ 1)M'_s$ or $(-1\ \ 3)$ (i.e. $p_s = -x + 3$).

3.2 Homogenization

In this section, we show how, given a segregated SFQ S, to obtain a strongly segregated SFQ S' such that $\mathrm{span}(S') = \mathrm{span}(S)$. The main result is Lemma 6, which gives a *homogenization procedure* which if applied repeatedly to a segregated SFQ will eventually bring about the strong segregation property.

Lemma 6. *Suppose S is a segregated SFQ. Then there exists a strongly segregated SFQ S' such that* $\mathrm{span}(S') = \mathrm{span}(S)$.

Proof. Suppose $S = \langle p_1/q_1, p_2/q_2, \ldots, p_m/q_m \rangle$. If S is strongly segregated, then there is nothing to prove. Otherwise, there exist i and j such that $q_i \sim q_j$ but $q_i \neq q_j$. Let $l = \mathrm{lcm}(q_i, q_j)$, let $p'_i = p_i(l/q_i)$ and $p'_j = p_j(l/q_j)$. Let S' be obtained from S by replacing the formal quotients p_i/q_i and p_j/q_j by the formal quotients p'_i/l and p'_j/l, respectively. Now, $\ulcorner p'_i/l \urcorner = \ulcorner p_i(l/q_i)/l \urcorner = \ulcorner p_i/q_i \urcorner$ and $\ulcorner p'_j/l \urcorner = \ulcorner p_j(l/q_j)/q_j \urcorner = \ulcorner p_j/q_j \urcorner$. Thus, S' represents the same sequence of rational expressions as does S, so that $\mathrm{span}(S') = \mathrm{span}(S)$. Moreover, since $q_i \sim q_j$, and hence l has the same set of irreducible factors as q_i and q_j, it follows that for all k, if q_k and l have a nontrivial common factor, then q_k and q_i also have a nontrivial common factor. Thus $q_k \sim q_i$ by the assumption that S is segregated, and hence $q_k \sim l$ as well. Since this holds for all k, it follows that S' is again segregated. The above procedure is repeated until it is no longer possible to do so, at which point a strongly segregated SFQ S' is obtained. Note that each iteration results in a strict reduction in the number of pairs i, j, such that $q_i \sim q_j$ but $q_i \neq q_j$, so termination is guaranteed. $\square$

In our example, starting with the segregated SFQ

$$S = \langle \frac{p_3}{q_3}, \frac{p_4}{q_4}, \frac{p_5}{q_5}, \frac{p_6}{q_6} \rangle = \langle \frac{1}{x-1}, \frac{-x+3}{(x-2)^2}, \frac{-1}{x-2}, \frac{1}{x-3} \rangle$$

we observe that $q_4 \sim q_5$, so we replace p_5/q_5 by $p_5(x-2)/q_4$, resulting in the strongly segregated SFQ

$$S = \langle \frac{1}{x-1}, \frac{-x+3}{(x-2)^2}, \frac{-x+2}{(x-2)^2}, \frac{1}{x-3} \rangle.$$

Note that the homogenization procedure does not change the sequence of rational expressions determined by the SFQ, hence the overall change of coordinates matrix associated with the procedure is the identity matrix.

3.3 Obtaining a BFQ

Once we have used the segregation procedure to produce a strongly segregated SFQ S, a BFQ S' can be obtained by applying a change of coordinates that transforms the numerators of each homogeneous subsequence $S(q)$ into elements of the natural basis. The following lemmas show precisely how to do this.

Lemma 7. *Suppose $S = \langle p_1/q_1, p_2/q_2, \ldots, p_m/q_m \rangle$ is a strongly segregated SFQ. Then S is a BFQ if and only if for each i with $1 \le i \le m$ the homogeneous subsequence $S(q_i)$ is a BFQ.*

Proof. It is clear that if S is a BFQ then each homogeneous subsequence $S(q_i)$ is also a BFQ. Conversely, suppose each $S(q_i)$ is a BFQ. We claim that if $S(q_i) \neq S(q_{i'})$ for some i and i', then no nonzero rational expression can at once be a linear combination of the rational expressions represented by $S(q_i)$ and a linear combination of the rational expressions represented by $S(q_{i'})$. In other words, $q_i \neq q_{i'}$ implies $\mathrm{span}(S(q_i)) \cap \mathrm{span}(S(q_{i'})) = \{0\}$ for all i, i'. But then it is immediate that the sequence of rational expressions represented by the elements of S is linearly independent if and only if the sequence of rational expressions represented by each homogeneous subsequence $S(q_i)$ is linearly independent; that is, if and only if each $S(q_i)$ is a BFQ.

To prove the claim, suppose $S(q_i) \neq S(q_{i'})$. Since S is strongly segregated by hypothesis, we have that q_i and $q_{i'}$ are coprime. Now, any nontrivial linear combination of rational expressions $\ulcorner p_{i,j}/q_i \urcorner$, where $p_{i,j}/q_i \in S(q_i)$, is equal to $\ulcorner p/q_i \urcorner$ for the same linear combination p of the polynomials $p_{i,j}$. Moreover, since $\deg(p_{i,j}) < \deg(q_i)$ holds for each formal quotient $p_{i,j}/q_i \in S(q_i)$, it follows that $\deg(p) < \deg(q_i)$ as well. Therefore, q_i must have some irreducible factor that is not also a factor of p. Similarly, any nontrivial linear combination of rational expressions $\ulcorner p_{i',j'}/q_{i'} \urcorner$, where $\ulcorner p_{i',j'}/q_{i'} \urcorner \in S(q_{i'})$, has the property that there is some irreducible factor of $q_{i'}$ that is not also a factor of p'. Since q_i and $q_{i'}$ are coprime, they have disjoint sets of irreducible factors, hence it is impossible to have $pq_{i'} = p'q_i$ unless p and p' are both zero. Consequently, the rational expressions $\ulcorner p/q_i \urcorner$ and $\ulcorner p'/q_{i'} \urcorner$ can only be equal if they are both zero. $\square$

Lemma 8. *Suppose $S = \langle (p_1, q), (p_2, q), \ldots, (p_m, q) \rangle$ is a homogeneous SFQ. Define*

$$S' = \langle (x^{d-1}, q), (x^{d-2}, q), \ldots, (1, q) \rangle,$$

where $d = \deg(q)$. Then S' is a BFQ such that $\mathrm{span}(S) \subseteq \mathrm{span}(S')$. Moreover, the matrix having the coefficient vectors of the p_j as its rows is the matrix of a transformation of coordinates from S to S'.

Proof. Obvious. □

In our example, we have

$$S = \langle \frac{1}{x-1}, \frac{-x+3}{(x-2)^2}, \frac{-x+2}{(x-2)^2}, \frac{1}{x-3} \rangle.$$

The homogeneous subsequences $S(x-1)$ and $S(x-3)$ already have elements of the natural basis for their numerators. Consider now the homogeneous subsequence

$$S((x-2)^2) = \langle \frac{-x+3}{(x-2)^2}, \frac{-x+2}{(x-2)^2} \rangle.$$

The matrix

$$M = \begin{pmatrix} -1 & 3 \\ -1 & 2 \end{pmatrix}.$$

is the matrix of a change of coordinates from $S((x-2)^2)$ to the BFQ

$$\langle \frac{x}{(x-2)^2}, \frac{1}{(x-2)^2} \rangle.$$

The overall change of coordinates is from the SFQ S to the BFQ

$$S' = \langle \frac{1}{x-1}, \frac{x}{(x-2)^2}, \frac{1}{(x-2)^2}, \frac{1}{x-3} \rangle$$

and it is given by the matrix

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 3 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

4 Representing Arrays

We now consider how to apply the results of the preceding section to obtain a representation for arrays of rational expressions.

Suppose we wish to represent an n-dimensional array E of rational expressions. The representation comprises:

1. A sequence $Q = \langle q_1, q_2, \ldots, q_m \rangle$ of pairwise coprime nonzero polynomials.
2. An $(n+1)$-dimensional array A of scalars that stores, for each element of the original array E, the coordinate vector of that element with respect to the BFQ

$$\begin{aligned} \langle\ & x^{d_1-1}/q_1, \ldots, 1/q_1, \\ & x^{d_2-1}/q_2, \ldots, 1/q_2, \\ & \ldots, \\ & x^{d_m-1}/q_m, \ldots, 1/q_m\ \rangle. \end{aligned}$$

   If $d_i = \deg(q_i)$ for $1 \le i \le m$ then each coordinate vector will contain $d_1 + d_2 + \ldots + d_m$ coefficients.

This representation has the advantage that the numerator polynomials do not need to be stored or explicitly manipulated, and that having the natural basis polynomials as the implied numerators automatically guarantees independence.

The issue of course arises as to how to initially construct an array representation of the above form given a list of n rational expressions that are to be the array elements. This can be done in a recursive fashion; at each stage dividing the given sequence in half, forming the array representation of each of the two subsequences, and then combining the two representations into a single representation for the whole sequence. For the base case of the recursion, we need to be able to construct an array representation from a single element list $\langle p/q \rangle$. Let $d = \deg(q)$ and suppose $p = a^{d-1}x^{d-1} + a^{d-2}x^{d-2} + \ldots + 1$. Since the sequence

$$x^{d-1}/q, x^{d-2}/q, \ldots 1/q$$

is obviously a BFQ, we may represent the 1-element array $\langle \ulcorner p/q \urcorner \rangle$ as the one-element sequence of polynomials $\langle q \rangle$ together with an array A whose single element is the vector $(a_{d-1}, a_{d-2}, \ldots, a_0)$ of coefficients of the polynomial p.

In the general case, we recursively construct array representations (Q_1, A_1) and (Q_2, A_2), which we then wish to combine into a single representation (Q, A). In order to do this, it is necessary to *amalgamate* the BFQs defined by the sequences Q_1 and Q_2 into a single BFQ defined by the sequence of polynomials Q. The following result describes formally how this can be done.

Lemma 9. *Suppose* $Q_1 = \langle q_{1,1}, q_{1,2}, \ldots, q_{1,m_1} \rangle$ *and* $Q_2 = \langle q_{2,1}, q_{2,2}, \ldots, q_{2,m_2} \rangle$ *are sequences of pairwise coprime nonzero polynomials that represent BFQs* S_1 *and* S_2*, respectively. Then we can compute a sequence* $\langle q_1, q_2, \ldots, q_m \rangle$*, representing a BFQ* S*, such that* $\mathrm{span}(S_1) \subseteq \mathrm{span}(S)$ *and* $\mathrm{span}(S_2) \subseteq \mathrm{span}(S)$*. Moreover, we can compute matrices* M_1 *and* M_2 *of a change of coordinates from* S_1 *and* S_2*, respectively, to* S*.*

Proof. Concatenate BFQs S_1 and S_2 to obtain SFQ S'. The change of coordinates from S_1 and S_2 to S' are given, respectively, by matrices M'_1 and M'_2 with the block forms:

$$M'_1 = \begin{pmatrix} I & 0 \end{pmatrix} \qquad M'_2 = \begin{pmatrix} 0 & I \end{pmatrix}.$$

Apply Lemma 2 to S' to obtain a segregated SFQ S'' such that $\mathrm{span}(S') \subseteq \mathrm{span}(S'')$, together with an associated change of coordinates matrix from S' to $\mathrm{span}(S'')$. Apply Lemma 6 to S'' to obtain a strongly segregated SFQ S''' such that $\mathrm{span}(S'') = \mathrm{span}(S''')$, together with an associated change of coordinates matrix from S'' to S'''. Finally, apply Lemma 8 to S''' to obtain a BFQ S such that $\mathrm{span}(S''') \subseteq \mathrm{span}(S)$, together with an associated change of coordinates matrix from S''' to S. Multiply matrices M'_1 and M'_2 by the change of coordinates matrices from each of the subsequent steps to obtain overall change of coordinates matrices M_1 and M_2 from S_1 and S_2, respectively, to S. □

For example, suppose we wish to create a two-element vector from the list:

$$\langle \frac{1}{x^3 - 5x^2 + 8x - 4}, \frac{1}{x^2 - 5x + 6} \rangle.$$

We recursively create one-element vectors from the lists:

$$\langle\frac{1}{x^3-5x^2+8x-4}\rangle \qquad \langle\frac{1}{x^2-5x+6}\rangle.$$

The first of these is represented by (Q_1, A_1), where Q_1 is $\langle x^3-5x^2+8x-4\rangle$ and A_1 is $\langle(0\ 0\ 1)\rangle$. The second is represented by (Q_2, A_2), where Q_2 is $\langle x^2-5x+6\rangle$ and A_2 is $\langle(0\ 1)\rangle$. Next, we amalgamate the sequences Q_1 and Q_2. This produces the pairwise coprime sequence

$$Q = \langle x-1, (x-2)^2, x-3\rangle$$

corresponding to the BFQ

$$S = \langle\frac{1}{x-1}, \frac{x}{(x-2)^2}, \frac{1}{(x-2)^2}, \frac{1}{x-3}\rangle$$

and the change of coordinates matrices

$$M_1 \quad = \quad \begin{pmatrix} 1 & 0 & 4 & 0 \\ 1 & -1 & 4 & 0 \\ 1 & -1 & 3 & 0 \end{pmatrix}$$

and

$$M_2 \quad = \quad \begin{pmatrix} 0 & -2 & 4 & 3 \\ 0 & -1 & 2 & 1 \end{pmatrix}.$$

So the array A is

$$\langle\begin{pmatrix} 1 & -1 & 3 & 0 \end{pmatrix}, \begin{pmatrix} 0 & -1 & 2 & 1 \end{pmatrix}\rangle$$

corresponding to the expansions

$$\frac{1}{x^3-5x^2+8x-4} = \frac{1}{x-1} + \frac{-x}{(x-2)^2} + \frac{3}{(x-2)^2}$$

and

$$\frac{1}{x^2-5x+6} = \frac{-x}{(x-2)^2} + \frac{2}{(x-2)^2} + \frac{1}{x-3}.$$

Note that the amalgamation operation is performed just once on Q_1 and Q_2 each time two representations (Q_1, A_1) and (Q_2, A_2) are combined. Once this has been done and we have obtained Q and the corresponding change of coordinates matrices M_1 and M_2, the coordinate vectors that are the entries of the result array A are obtained by multiplying each entry of A_1 by M_1 and each entry of A_2 by M_2. If A_1 and A_2 are regarded as matrices having these coordinate vectors as their rows, then this calculation amounts to constructing the block matrix

$$A = \begin{pmatrix} A_1M_1 \\ A_2M_2 \end{pmatrix},$$

where A_1M_1 and A_2M_2 are matrix products.

5 Operations on Arrays

We now consider various operations that can be performed on arrays of rational expressions represented as described in the preceding section.

5.1 Structural Operations

Arrays that have special structure, such as identity matrices or arrays that are constant along one or more of their dimensions, can be constructed by building scalar arrays with a related structure. For example, an n-vector of rational expressions, all of which are zero, can be represented as $(\langle 1\rangle, A)$, where A is an $(n \times 0)$-matrix of scalars (*i.e.* all the rows are empty coordinate vectors). A constant n-vector having rational expression $e = \ulcorner p/q \urcorner$ in all its entries can be represented as $(\langle q\rangle, A)$ where A is a $(n \times \deg(q))$-matrix, each row of which contains the coefficients of the polynomial p. Similarly, an n-vector having a single nonzero entry $e = p/q$ can be represented as $(\langle q\rangle, A)$, where A is an $(n \times \deg(q))$-matrix having a single nonzero row containing the coefficients of p. Compact representations of arrays of all these forms are readily constructed using MTBDD-based scalar arrays as the underlying data structure.

Because an array E of rational expressions is represented in terms of a scalar array A having one additional dimension, structural operations on arrays of rational expressions are easily implemented in terms of corresponding operations on scalar arrays. For example, if E is a 2-dimensional array of rational expressions (*i.e.* a matrix), then extracting a row or column of E amounts to extracting a 2-dimensional slice of the underlying 3-dimensional scalar array A. Similarly, small arrays can be pasted together horizontally or vertically to form larger arrays by first applying amalgamation and then performing a corresponding pasting operation on scalar arrays. These operations can be efficiently implemented using MTBDDs for arrays whose size in each dimension is a power of 2. This restriction does not typically pose any hardship in practice, because the "wasted space" incurred by rounding the sizes of arrays up to the next higher power of two only costs a few additional nodes in an MTBDD representation.

5.2 Scaling

If array E of rational expressions is represented by (Q, A), and if $c \in \mathcal{K}$ is a scalar, then the scaled array cE is represented by (Q, cA). Although this appears to imply the need to scale each element of A individually, this can be avoided if the underlying matrix representation maintains a scale factor separately from the matrix elements. Some kinds of BDD-based matrix representations (*e.g.* [CT96]) naturally support this.

Alternatively, we can avoid operating on every entry of A if we generalize the (Q, A) representation slightly so that we can scale the underlying BFQ. For example, if we permit Q to contain non-monic polynomials and make the appropriate adjustments to accomodate this generalization, then we could represent the array cE by $(c^{-1}Q, A)$ for c nonzero. The case $c = 0$ would then be handled as a special case that returns the representation of an identically zero array.

5.3 Addition and Subtraction

Suppose arrays E_1 and E_2 (of the same dimension) are represented by (Q_1, A_1) and (Q_2, A_2), respectively, where Q_1 represents the BFQ S_1 and Q_2 represents the BFQ S_2. In order to compute the sum $E_1 + E_2$ or difference $E_1 - E_2$, first amalgamate Q_1 and Q_2 to obtain Q, together with change of coordinates matrices M_1 and M_2. The sum $E_1 + E_2$ is now represented by $(Q, A_1M_1 + A_2M_2)$ and the difference $E_1 - E_2$ by $(Q, A_1M_1 - A_2M_2)$.

5.4 Zero and Equality Testing

If array E of rational expressions is represented by (Q, A), then E is identically zero if and only if A is an identically zero array of scalars. So zero-testing for arrays of rational expressions reduces to zero-testing for scalar arrays. Testing whether two arrays of rational expressions E_1 and E_2 are equal can be accomplished by computing the difference $E_1 - E_2$ and testing whether it is identically zero.

5.5 Translation

If $e = \ulcorner p/q \urcorner$ is a rational expression, and $a \in \mathcal{K}$, then the *translation* of e by a is the rational expression $e\{x + a\} = \ulcorner p\{x + a\}/q\{x + a\}\urcorner$, where $p\{x + a\}$ and $q\{x + a\}$ denote the result of substituting $x + a$ for the indeterminate x in the polynomials p and q, respectively. The fact that partial fraction expansions are preserved by translation makes it possible to efficiently perform translation on representations (Q, A), as we now show.

Lemma 10. *If polynomial q is irreducible, then so is $q\{x + a\}$, for any $a \in \mathcal{K}$.*

Proof. Suppose $q\{x+a\}$ has a nontrivial factorization rs. Then $q = q\{x+a\}\{x-a\} = (rs)\{x - a\} = (r\{x - a\})(s\{x - a\})$, yielding a factorization of q that is nontrivial because translation preserves degree. Thus, reducibility of $q\{x+a\}$ implies reducibility of q, or equivalently, irreducibility of q implies irreducibility of $q\{x + a\}$. □

Lemma 11. *If q_1 and q_2 are coprime, then so are $q_1\{x + a\}$ and $q_2\{x + a\}$, for any $a \in \mathcal{K}$.*

Proof. If r is a nontrivial common factor of both $q_1\{x + a\}$ and $q_2\{x + a\}$, then $r\{x - a\}$ is a nontrivial common factor of both $q_1 = q_1\{x + a\}\{x - a\}$ and $q_2 = q_2\{x + a\}\{x - a\}$. Thus, if q_1 and q_2 have no nontrivial common factors, neither do $q_1\{x + a\}$ and $q_2\{x + a\}$. □

Corollary 12. *If $S = \langle p_1/q_1, p_2/q_2, \ldots, p_m/q_n\rangle$ is a strongly segregated BFQ, then so is*

$$S\{x + a\} = \langle p_1\{x + a\}/q_1\{x + a\}, \ldots, p_m\{x + a\}/q_m\{x + a\}\rangle.$$

Moreover, if rational expression e has coordinate vector $\langle a_1, a_2, \ldots, a_m\rangle$ with respect to S, then $e\{x + a\}$ has that same coordinate vector with respect to $S\{x + a\}$.

From the above results, if array E of rational expressions is represented by (Q, A), then the translation $E\{x + a\}$ of E is represented by $(Q\{x + a\}, A)$. That is, translation of a vector of rational expressions can be accomplished simply by translating the denominator polynomials of the underlying BFQ, without having to treat each entry of the array individually.

5.6 Evaluation

If $e = \ulcorner p/q \urcorner$ is a rational expression, and $a \in \mathcal{K}$ is such that the value $q(a)$ of polynomial q at a is nonzero, then the *value* $e(a)$ of e at a is given by the quotient $p(a)/q(a) \in \mathcal{K}$. The value $E(a)$ at a of an array E of rational expressions is the scalar array whose entries are the values at a of the corresponding entries of E. If array E of rational expressions is represented by (Q, A), where $Q = \langle q_1, q_2, \ldots, q_m \rangle$, and $a \in \mathcal{K}$ is such that $q_i(a) \neq 0$ for all $1 \leq i \leq m$, then the value $E(a)$ of E at a is given by the product:

$$A \cdot \langle q_1(a)^{-1}, q_2(a)^{-1}, \ldots, q_m(a)^{-1} \rangle^{\mathrm{t}}.$$

Actually, the requirement that $q_i(a) \neq 0$ for all $1 \leq i \leq m$ is too strong, and it can be weakened to the following: $q_i(a) \neq 0$ for all $1 \leq i \leq m$ *for which the corresponding slice A_i is nonzero*. (The slice A_i is the scalar array consisting of the ith coordinates of each entry of A.) This is an important consideration, because as operations are performed on arrays of rational expressions there can tend to accumulate in Q polynomials q_i that are unnecessary because the corresponding slices A_i are identically zero. "Garbage collection" of these unnecessary elements of Q from time to time may be useful in reducing time and space requirements.

5.7 Contraction

If E is an array of rational expressions, represented by (Q, A), then the sum of all the entries of E is the rational expression whose coordinate vector with respect to Q is the vector obtained by summing all the coordinate vectors that are the entries of A. If an MTBDD-based representation is used for A, then a recursive algorithm can be used to compute this coordinate vector efficiently (*i.e.* in time on the order of the number of nodes of the MTBDD representation of A, rather than the total number of entries of A).

5.8 Kronecker Product

The Kronecker (or tensor) product of an m-element array A and an n-element array B is the mn-element array $A \otimes B$ whose entries are all possible products of an entry from A and an entry from B. If A and B happen to be matrices, then we can think of $A \otimes B$ as a block matrix that consists, for each position (i, j) in A, of a copy of B that has been scaled by the entry $a_{i,j}$ at that position of A. Kronecker products arise naturally when forming parallel compositions of various kinds of automata represented in terms of transition matrices. If (as as can be done with MTBDD-based implementations) the underlying representation of scalar arrays admits an efficient method for computing

Kronecker products, then we can take advantage of this to efficiently compute the Kronecker product $C \otimes E$ of a scalar array C and an array E of rational expressions. In particular, if E is represented by (Q, A), then $C \otimes E$ is represented by $(Q, C \otimes A)$.

To compute the Kronecker product of two arrays of rational expressions is somewhat more difficult. Suppose E_1 and E_2 are two arrays of rational expressions, represented by (Q_1, A_1) and (Q_2, A_2), respectively. Let

$$S_1 = \langle p_{1,1}/q_{1,1}, p_{1,2}/q_{1,2}, \ldots, p_{1,m_1}/q_{1,m_1} \rangle$$

and

$$S_2 = \langle p_{2,1}/q_{2,1}, p_{2,2}/q_{2,2}, \ldots, p_{2,m_2}/q_{2,m_2} \rangle$$

be the BFQs corresponding to Q_1 and Q_2, respectively. If e_1 and e_2 are entries of E_1 and E_2, respectively, then $e_1 = \sum_{i=1}^{m_1} a_{1,i} \ulcorner p_{1,i}/q_{1,i} \urcorner$ and $e_2 = \sum_{j=1}^{m_2} a_{2,j} \ulcorner p_{2,j}/q_{2,j} \urcorner$. Thus $e_1 e_2$, the corresponding entry of $E_1 \otimes E_2$ satisfies

$$e_1 e_2 = \sum_{i=1}^{m_1} \sum_{j=1}^{m_2} a_{1,i} a_{2,j} \ulcorner p_{1,i} p_{2,j} / q_{1,i} q_{2,j} \urcorner .$$

In other words, if V_1 and V_2 are the coordinate vectors of e_1 and e_2 with respect to BFQs S_1 and S_2, respectively, then $V_1 \otimes V_2$ is a coordinate vector of $e_1 e_2$ with respect to the SFQ $S_1 \otimes S_2$.

The above observations lead to the following method for computing $E_1 \otimes E_2$. First, compute the array $A = A_1 \otimes A_2$, organized as an array of coordinate vectors obtained by taking all possible Kronecker products of an entry of A_1 and an entry of A_2. Then the entries of A are coordinate vectors of the entries of $E_1 \otimes E_2$ with respect to the SFQ $S_1 \otimes S_2$. Next, apply the segregation procedure to $S_1 \otimes S_2$ to obtain a pairwise coprime sequence of polynomials Q representing a strongly segregated BFQ S, together with the associated change of coordinates matrix M from $S_1 \otimes S_2$ to S. Finally, compute AM to obtain the representation (Q, AM) of $E_1 \otimes E_2$. Note that all we really need is the sequence Q, the array A and the matrix M, and these can be obtained without explicitly computing either $S_1 \otimes S_2$ or S.

If the sequence Q_1 contains polynomials that have irreducible factors that are mostly distinct from those of Q_2, then the resulting sequence Q will tend to be on the order of the product of the lengths of Q_1 and Q_2. On the other hand, if the irreducible factors are mostly the same between the two sequences, then Q will tend to be similar in length to the lengths of Q_1 and Q_2. Although the intermediate array A seems relatively "large" in terms of number of entries, in fact in an MTBDD-based representation it can be arranged (by placing scalar "weights" at the nodes of the decision diagram, so that the value in each entry of an array is obtained by taking the product of the weights of each of the nodes along a path to a leaf node in the MTBDD), that the number of nodes in the representation of A need only be the sum of the number of nodes in A_1 and the number of nodes in A_2. So if A_1 and A_2 have compact representations, A will as well.

5.9 Hadamard Product

The Hadamard or (entrywise) product of two arrays of the same shape (*i.e* same number and sizes of dimensions) is the array obtained by multiplying the corresponding elements of the given arrays. If E is an array of rational expressions, represented by (Q, A), and C is an array of scalars of the same shape as E, then the Hadamard product $C \Box E$ is represented by $(Q, C \Box A)$, assuming we treat A as an array of coordinate vectors. Alternatively, if we treat A as an array of scalars (with an additional dimension beyond those of E), then the same result can be computed as $\bar{C} \Box A$, where $\bar{C}$ is the array of scalars of the same shape as A that is obtained by replicating the entries of C along the additional dimension. This idea is efficient and simple to implement using MTBDDs.

The Hadamard product of two arrays of rational expressions can be computed using ideas similar to those already explained for Kronecker product. Suppose E_1 and E_2 are two arrays of rational expressions of the same shape, represented by (Q_1, A_1) and (Q_2, A_2), respectively. Let S_1 and S_2 be the BFQs corresponding to Q_1 and Q_2, respectively. To compute $E_1 \Box E_2$, first compute the array A whose entries are the coordinate vectors obtained by taking the Kronecker products of corresponding entries of A_1 and A_2. The entries of A are coordinate vectors of $E_1 \Box E_2$ with respect to the SFQ $S_1 \otimes S_2$. Then, compute the sequence Q and change of coordinates matrix M as was done for Kronecker product, and finally obtain the representation (Q, AM) for $E_1 \Box E_2$.

5.10 Matrix Product

The product of an $(m \times p)$-matrix A and an $(p \times n)$-matrix B can be computed as follows: First, expand A to an $(m \times p \times n)$-array $\bar{A}$ by replicating the existing entries along the new dimension. Next, expand B to an $(m \times p \times n)$-array $\bar{B}$ in a similar way. Then, form the Hadamard product $C = \bar{A} \Box \bar{B}$. Finally, reduce C to an $(m \times n)$-matrix by summing along the "middle" dimension. Although this is not the usual way that matrix product is described, in the context of an MTBDD-based implementation it is a reasonable way to compute it. Note that we need not limit ourselves to (two-dimensional) matrices; the same scheme serves to describe the product of arrays with arbitrary numbers of dimensions, with an arbitrary set of "overlapping" dimensions, save only that the arrays are conformant in the sense that corresponding overlapping dimensions have the same size. This basic idea can be applied directly to compute products of various combinations of arrays of scalars and and arrays of rational expressions, including dot product, matrix/vector product, and matrix/matrix product.

5.11 Conversion to Traditional Representation

Suppose the n-dimensional array E of rational expressions is represented by the pair (Q, A), where $Q = \langle q_1, \ldots, q_m \rangle$. For $1 \leq i \leq m$, let $d_i = \deg(q_i)$, and let $d = d_1 + \ldots + d_m$. Then the array A can be regarded as a sequence of n-dimensional "slices":

$$A_{1,1}, \ldots, A_{1,d_1}, A_{2,1}, \ldots, A_{2,d_2}, \ldots, A_{m,1}, \ldots, A_{m,d_m}$$

so that the array E is given by:

$$E = \sum_{i=1}^{m} \sum_{j=0}^{d_i - 1} A_{i,j} \frac{x^j}{q_i}. \quad (1)$$

To convert from the (Q, A) representation to a traditional matrix representation is simply a matter of evaluating expression (1) using the traditional representation.

5.12 Improper Rational Expressions

We have assumed up to this point that arrays to be represented contained only proper rational expressions, for which the degree of the numerator is strictly less than the degree of the denominator. However, it is not difficult to extend the representation and algorithms to handle improper rational expressions as well. One easy extension, which hardly requires any changes, is to allow rational expressions for which the degree of the numerator is less than or equal to the degree of the denominator. Since every such rational expression is equivalent to a scalar plus a proper rational expression, all that needs to be done to handle such expressions is to reserve one entry of the coordinate vector as the coefficient of x^0 (*i.e.* 1).

More generally, an arbitrary improper rational expression can be expressed as the sum of an "integral part," which is a polynomial, and a "fractional part," which is a proper rational expression. To handle such expressions we simply need to add to the representation (Q, A) a nonnegative integer d that gives the degree of the highest positive power of x, and to reserve $d + 1$ elements of the coefficient vector for the coefficients of $x^d, x^{d-1}, \ldots, x^1, x^0$. It is not necessary to represent these polynomials explicitly, although the computations of the various change of coordinates matrices have to be modified to take into account the presence of these additional basis elements.

6 Conclusion

We have shown how to represent n-dimensional arrays of rational expressions as $(n + 1)$-dimensional arrays of scalars, in a way that can take advantage of memory-saving MTBDD-based representations for large sparse scalar arrays. A variety of common operations on arrays of rational expressions map efficiently to this representation. The algorithms described here were implemented by the author in the Standard ML programming language, and applied successfully as part of a package [ZCS03] for manipulating representations of large sparse probabilistic systems. In this package, matrices of rational expressions were used to represent systems that are "open" in the sense of having environments that are not yet specified. The implementation relied on an underlying MTBDD-based representation for scalar arrays. Three particular operations on matrices of rational expressions were central to this application, and served as the motivation for devising the representation described in this paper: (1) translation of the entries of a matrix of rational expressions by a common scalar, (2) Kronecker product of a matrix of scalars and a matrix of rational expressions, and (3) Evaluation of the entries of a matrix of rational expressions on a common argument, to obtain a matrix of scalars.

References

[BFG+93] R. Iris Bahar, Erica A. Frohm, Charles M. Gaona, Gary D. Hachtel, Enrico Macii, Abelardo Pardo, and Fabio Somenzi. Algebraic decision diagrams and their applications. In *IEEE /ACM International Conference on CAD*, 1993.

[CFM+97] E. Clarke, M. Fujita, P. McGeer, K. McMillan, J. Yang, and X. Zhao. Multi-terminal binary decision diagrams: An efficient data structure for matrix representation. *Formal Methods in System Design*, 10(2/3):149–169, 1997.

[Col75] G. E. Collins. Computer algebra of polynomials and rational functions. *American Mathematical Monthly*, 80:aug–sept, 1975.

[CT96] G. Ciardo and M. Tilgner. On the use of Kronecker operators for the solution of generalized stochastic Petri nets. Technical Report 96-35, ICASE Report, 1996.

[Fat06] Richard J. Fateman. Rational function computing with poles and residues, 2006.

[HMKS99] H. Hermanns, J. Meyer-Kayser, and M. Siegle. Multi-terminal binary decision diagrams to represent and analyse continuous time Markov chains. *In Proc. 3rd International Workshop on the Numerical Solution of Markov Chains*, pages 188–207, 1999.

[Hor72] Ellis Horowitz. Algorithms for rational function arithmetic operations. In *STOC '72: Proceedings of the fourth annual ACM symposium on Theory of computing*, pages 108–118, New York, NY, USA, 1972. ACM.

[KNP02a] M. Kwiatkowska, G. Norman, and D. Parker. PRISM: Probabilistic symbolic model checker. In T. Field, P. Harrison, J. Bradley, and U. Harder, editors, *Proc. 12th International Conference on Modeling Techniques and Tools for Computer Performance Evaluation (TOOLS'02)*, volume 2324 of *LNCS*, pages 200–204. Springer-Verlag, 2002.

[KNP02b] Marta Kwiatkowska, Gethin Norman, and David Parker. Probabilistic symbolic model checking with prism: A hybrid approach. In *International Journal on Software Tools for Technology Transfer (STTT*, pages 52–66. Springer, 2002.

[SS98] E. W. Stark and S. Smolka. Compositional analysis of expected delays in networks of probabilistic I/O automata. In *Proc. 13th Annual Symposium on Logic in Computer Science*, pages 466–477, Indianapolis, IN, June 1998. IEEE Computer Society Press.

[WSS97] S.-H. Wu, S. A. Smolka, and E. W. Stark. Composition and behaviors of probabilistic I/O automata. *Theoretical Computer Science*, 176(1-2):1–38, 1997.

[Xin04] Guoce Xin. A fast algorithm for partial fraction decompositions, August 2004.

[ZCS03] D. Zhang, R. Cleaveland, and E. Stark. The integrated CWB-NC/PIOATool for functional verification and performance analysis of concurrent systems. In H. Garavel and J. Hatcliff, editors, *Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2003)*, volume 2619 of *Lecture Notes in Computer Science*, pages 431–436, Warsaw, Poland, April 2003. Springer-Verlag.