Distributed Computation of Mixed Volume

Ioannis Z. Emiris   Thierry Giordano

INRIA Sophia Antipolis, projet SAFIR
2004 Route des Lucioles BP 93
06902 Sophia-Antipolis CEDEX, France
E-mail: Ioannis.Emiris@sophia.inria.fr
Thierry.Giordano@sophia.inria.fr

Abstract

We propose a parallel algorithm for computing the mixed volume of n convex polytopes in n-dimensional space. Mixed volume generalizes standard Euclidean volume and is the cornerstone of a, relatively recent, theory on sparse polynomials. Its importance in algebraic geometry stems from the fact that it bounds the number of common roots of a system of equations and its computation leads to a numerical computation of these roots.

Our work modifies the sequential algorithm *Lift-Prune*, which computes mixed volume in arbitrary dimension and is the fastest algorithm today for general inputs. The new algorithm is best suited to coarse-grain parallelization or distributed systems. We present a distributed implementation based on a general tool, called *Central Control*, for communication between workstations. The reported experiments on 4 workstations illustrate the robustness and speed improvements over the sequential program. New results, obtained for certain benchmark problems, are listed in order to provide evidence that the speedup is almost linear in the number of workstations. Our program is publicly available on the Internet.

Keywords: Mixed volume, convex hull, parallel algorithm, distributed implementation, experimental results

1 Introduction

The mixed volume of a set of polytopes generalizes the notion of standard Euclidean volume of a single polytope, and provides a geometric way to bound the number of common zeros of a polynomial system. This remarkable theorem [Ber75] is the cornerstone of sparse elimination theory, a modern approach to computationally efficient methods for studying sparse polynomial equations and calculating their finite roots. Hence, mixed volume measures the complexity of an algebraic problem, in particular when the polynomials have few nonzero terms. This is the most typical case in practical applications, such as in graphics, modeling, biology, robotics, vision, as well as computational geometry; for concrete examples see [EC95, VGC96] and the references found there.

The parallel algorithm proposed in section 2 is based on a sequential algorithm called *Lift-Prune* described in [EC95]. Our first contribution was to transform the sequential algorithm in a distributed algorithm. Distribution and parallelism are different: the first applies to a comparatively small number of loosely connected workstations, whereas the second refers to a very large number of processors, typically on a single machine. We have chosen to parallelize this particular algorithm because it was the fastest one for general inputs. There is another algorithm more efficient for specific inputs described in [VGC96]. Some studies on complexities of mixed volume algorithms are in [DGH94].

Our second contribution was a robust distributed implementation of the new algorithm. For this program, proposed in section 3, a software tool called *Central Control* was used [DGS94b]. *Central Control* handles communication and manages the workstations in the network in case of failure.

Our aim was a gain of robustness and of time computing the mixed volume. So, we compared, in section 4, the results given by the distribution version and the sequential version on a computer algebra benchmark, known as the cyclic n-root family of systems. Experiments have been done on 4 workstations (of which one simply manages the distribution) and yield an acceleration factor of about 3 for $n = 10$. We also get a new result computing for the first time the mixed volume for the cyclic 12-roots system.

2 Mixed Volume Theory

To give definitions and algorithm for the mixed volume we need such notions as those of liftings and subdivisions; details can be found in [EC95]. Lifting embeds "generically" n-dimensional polytopes, edges, points, etc... to $n+1$-dimension. There are 2 equivalent definitions for the mixed volume $MV(Q_1,\ldots,Q_n)$ of polytopes $Q_1,\ldots,Q_n$.

Definition 2.1 *For $\lambda_1, \ldots, \lambda_n \in \mathbb{R}_{\geq 0}$ and convex polytopes $Q_1, \ldots, Q_n \in \mathbb{R}^n$, the mixed volume $MV(Q_1, \ldots, Q_n)$ is precisely the coefficient of $\lambda_1 \ldots \lambda_n$ in $Vol(\lambda_1 Q_1 + \ldots + \lambda_n Q_n)$ expanded as a polynomial in $\lambda_1, \ldots, \lambda_n$.*

Definition 2.2 *Given a mixed subdivision Δ of the Minkowski sum $Q_1 + \ldots + Q_n$, let E be the set of mixed cells in Δ. Then, $MV(Q_1, \ldots, Q_n) = \sum_{\sigma \in E} Vol(\sigma)$.*

The mixed volume generalizes the common volume of a polytope in dimension n. We have the relation:

$$MV(Q, \ldots, Q) = n! Vol(Q).$$

It can be seen as a "volume" of n polytopes in dimension n. There are also relations between the mixed volume and the volume of the Minkowski sum of the given n polytopes; see the references in [EC95].

Lemma 2.3 *For all $J \subset \{1, \ldots, n\}$ and $L \subset J$, if $\sum_{j \in J} \widehat{e}_j$ does not lie on the lower envelope of $\sum_{j \in J} \widehat{Q}_j$ then $\sum_{l \in L} \widehat{e}_l$ does not lie on the lower envelope of $\sum_{l \in L} \widehat{Q}_l$, where $\widehat{e}_i \in \widehat{Q}_i$.*

The following algorithm is based on the *Lift-Prune* algorithm [EC95]. In this algorithm, we use definition 2.2.

Input: $Q_1, \ldots, Q_n \in \mathbb{R}^n$
Output: $MV(Q_1, \ldots, Q_n) \in \mathbb{Z}_{\geq 0}$
Parallel algorithm:

1. Enumerate the edges of all polytopes $Q_1, \ldots, Q_n$ respectively into sets $E_1, \ldots, E_n$. Compute random lifting vectors $l_1, \ldots, l_n \in \mathbb{Q}^n$. Compute lifted edge $\widehat{e}_i$ for every edge $e_i \in E_i$, $i = 1, \ldots, n$. Initialize the mixed volume to 0.

2. If $E_1 = 0$ then terminate
Otherwise, pick any edge $e_i \in E_i$, remove it from E_1, create current tuple (e_1), let sets $E'_2, \ldots, E'_n$ be copies of $E_2, \ldots, E_n$ and let $k = 1$.

3. Let i range from $k+1$ to n:
For every $e_i \in E'_i$, if $\sum_{j=1}^{k} \widehat{e}_j + \widehat{e}_i$ does not lie on the lower envelope of $\sum_{j=1}^{k} \widehat{Q}_j + \widehat{Q}_i$ then e_i is removed from E'_i. Increment k. Here we use lemma 2.3

4. If $k = disdepth$ (where $disdepth$ is the depth of distribution), then continue in parallel at step 5 and go to step 3, else continue at step 5.

5. If $k > n$, then add the volume of the Minkowski sum of $(e_1, \ldots, e_n)$ to the mixed volume; continue at step 2.
If $k \leq n$ and $E'_k = 0$ then continue at step 2. If $k \leq n$ and $E'_k \neq 0$ then for each edge $e_k \in E'_k$ add it to the current tuple $(e_1, \ldots, e_{k-1})$, remove e_k from E'_k and go to step 3.

3 Mixed Volume Software

There was a program using the algorithm described in section 2 without step 4. But a new program has been written in order to distribute the computation. The distributed implementation relies on a software tool called *Central Control* [DGS94b]. The *Central Control* is a software component that has been designed to be the kernel of environments for scientific computations which can offer a common and concurrent access to different applications. It uses the communication protocol *ASAP* [DGS94a]. The new software computing the mixed volume is based on the algorithm of section 2 and it is composed of 3 components: a main program called *mastermv*, the *Central Control*, a server program called *partiel*.

The main program launches a *Central Control* session, and this session creates several servers, called *partiel*. The *mastermv* computes step 1 to 5 until $k < disdepth$, then it communicates necessary information to *Central Control*, which communicates this information to a *partiel*, then the *partiel* continues with step 5 and in the same time, the *mastermv* continues with step 3.

We get the software more robust working on the *Central Control* part: it happens that server died for many reasons, and we didn't want to lose important information. So we use first a method of memory: during the communication between the *mastermv* and the *partiel* via *Central Control*, we keep all information sent to the *partiel* in *Central Control* , and on a second time, with *Central Control* we get the possibility to create a new server to replace the dead one and send the saved information to another *partiel*.

In the program, we cannot always compute in parallel without waiting in the main program because we had only a fixed number of servers and when they are all computing, the main program cannot send a new computation to them, so it has to wait for one of them to finish.

It is possible to find the software proposed above at the HTML page:
`http://www.inria.fr/safir/SAM/Mixvol-distrib/mixvol.html`
or, by ftp, in file `ftp://www.inria.fr/safir/dist-mixvol.tar.gz`.

4 Experimental Results

Experiments have been done on the problem of cyclic n-roots, a family of systems and a common benchmark in computer algebra.

$$x_1 + x_2 + \ldots + x_n = 0, \ldots, \sum_i \prod_{j=1}^{i+k-1} x_j = 0, (x_{n+j} = x_j), \ldots, x_1 x_2 \ldots x_n = 1.$$

The number of roots is given by the mixed volume of the Newton polytopes of the given polynomial equations. The Newton polytope is the convex hull of the exponent vectors of the nonzero monomials in each polynomial. For example, the first polynomial has Newton polytope vertices $\{(1,0,\ldots,0),\ldots,(0,\ldots,0,1)\}$.

Below are the results we obtain using 4 workstations of varying power: two Dec and two Sun-SPARC workstations. Except for one Dec, all other workstations are less powerful than the workstation used to compute the sequential version. Note that one of the workstations is used for running the master program and controlling the others, thus it is not computing any useful results. We also compare with the sequential version. In conclusion, we see that the speedup is close to 3, in other words almost linear in the actual number of workstations contributing to the final result.

cyclic	depth	Mix. Vol	distributed	sequential
$3-5$	1	$6, 16, 70$	$9s$	$0s$
6	1	156	$17s$	$1s$
7	1	924	$30s$	$18s$
8	1	2560	$1m55s$	$2m22s$
9	1	11016	$12m$	$23m$
10	2	35940	$1h20m$	$3h20m$
11	2	184756	$12h$	$31h45$

A new result, which has not been able to compute with any sequential implementation, is the mixed volume (and hence the number of isolated roots) of the cyclic 12-root system: 500352.

References

[Ber75] D.N. Bernstein. The number of roots of a system of equations. *Funct. Anal. and Appl.*, 9(2):183–185, 1975.

[DGH94] M. Dyer, P. Gritzmann, and A. Hufnagel. On the complexity of computing mixed volumes. Manuscript, December 1994.

[DGS94a] S. Dalmas, M. Gaëtano, and A. Sausse. ASAP: A protocol for symbolic computation systems. Technical Report 162, INRIA, Sophia-Antipolis, 1994. ftp://ftp.inria.fr/INRIA/publication/publi-ps-gz/RT/RT-0162.ps.gz.

[DGS94b] S. Dalmas, M. Gaëtano, and A. Sausse. Distributed computer algebra: the Central Control approach. In *PASCO'94*, pages 104–113, Linz, Austria, September 1994. World Scientific.

[EC95] I.Z. Emiris and J.F. Canny. Efficient incremental algorithms for the sparse resultant and the mixed volume. *J. Symbolic Computation*, 20(2):117–149, August 1995.

[VGC96] J. Verschelde, K. Gatermann, and R. Cools. Mixed volume computation by dynamic lifting applied to polynomial system solving. *Discr. and Comput. Geometry*, 1996. To appear. Also tech. report TW222, Katholieke Universiteit Leuven, Dept. of Comp. Science, 1995.