THE BENEFITS OF CLUSTERING IN CYLINDRICAL ALGEBRAIC DECOMPOSITION

TERESO DEL RÍO AND MATTHEW ENGLAND

Abstract. Cylindrical Algebraic Decomposition (CAD) is a very powerful algorithm with many potential applications. However, its doubly-exponential complexity limits its usability. In this document we demonstrate how the techniques of adjacency and clustering would reduce the double exponent of CAD complexity.

1. Introduction

Cylindrical Algebraic Decomposition (CAD), was introduced by Collins in [5]. Given an ordering in n variables (e.g. $x_1 \succ ... \succ x_n$) a CAD is a *decomposition* of $\mathbb{R}^n$ into cells (i.e. connected regions). These cells must be *semi-algebraic* and for any pair of cells their projections with respect to the given ordering are either equal or disjoint (*cylindricity*).

A CAD can be produced with respect to a set of polynomials in such a way that each of the polynomials is sign-invariant on each of the cells. This is very powerful, and in particular, it can be used to solve any Tarski formulae described by the given polynomials and any associated Real Quantifier Elimination problem, if an appropriate variable ordering is chosen.

CAD has been applied in a wide variety of problems, such as disproving an existent biological conjecture [9], analysing numerical schemes [11] and the "piano movers" problem [12]. However, CAD has a doubly-exponential worst case complexity in the number of variables [4], meaning that problems with many variables cannot yet be tackled. It is only thanks to 40 years of research that it is possible to apply CAD to the problems above [3].

One interesting improvement was introduced by Arnon in [1] in the early days of CAD. It consisted of clustering adjacent cells in which the sign of the given polynomials was invariant, to study them together. This idea brought savings in small examples, but at that time few algorithms to find adjacencies existed, and those that did were comparatively expensive. Over time, better projection algorithms were designed, reducing the amount of computations needed to build a CAD, and when using those new projections it was no longer possible to cluster using adjacency as presented in [1]. For these reasons, and perhaps because Arnon left academia, adjacency and clustering have not been major topics of CAD research and are not used in many CAD implementations.

However, the theory on finding adjacencies has improved since Arnon's original work, first with the local box algorithms of [7] and more recently the validated numerics approach of Strzebonski in [10], which gives a powerful and cheap algorithm to find adjacencies in CAD.

The first author is supported by Coventry University and the second by EPSRC grant EP/T015748/1: *Pushing Back the Doubly-Exponential Wall of Cylindrical Algebraic Decomposition* (DEWCAD)..

The presentation at EACA 2022 was given by the first author.

Strzebonski's results imply that the potential of clustering can be exploited if the necessary theory were extended to the modern projection algorithms such as [8]. In preparation and motivation for such developments, we present here the theoretical complexity benefits of clustering, namely a reduction in the double exponent of the worst case complexity bound.

2. Cylindrical Algebraic Decomposition

The CAD algorithm requires a variable ordering (e.g. $x_1 \succ ... \succ x_n$) and can be split into two phases: projection and lifting. In the projection phase, a projection operator is applied to a set of polynomials in n variables to obtain a set of polynomials without the largest variable in the ordering. This is done recurrently until a set of polynomials in one variable is found. The set of polynomials with only the first i variables will be denoted S_i.

This allows the next phase, the lifting phase. Here a CAD of $\mathbb{R}$ is built using S_1, then on top of it, a CAD of $\mathbb{R}^2$ is constructed using S_2, and we continue recursively until the desired CAD of $\mathbb{R}^n$ is obtained. The CAD of $\mathbb{R}^i$ will be denoted as CAD_i.

CAD_I is built from S_i and CAD_{i-1} by taking each cell of CAD_{i-1}, substituting its sample point into S_i, isolating roots, and inferring the behaviour in the whole cell from this analysis of the sample. In order for the cells generated in $\mathbb{R}^j$ to be sign-invariant we need to prove that the structure of the roots of S_i is invariant over the cell in CAD_{i-1}. This property is called delinability. The following theorem of Collins gives this assurance for his projection operator.

Theorem 2.1. *(Theorem 5 of* [5]*) Let $\mathcal{T}$ be a non-empty set of non-zero real polynomials in r real variables, $r \geq 2$. Let A be a connected subset of $\mathbb{R}^{r-1}$. Let $\mathcal{P}$ be the Collins projection of $\mathcal{T}$. If every element of $\mathcal{P}$ is sign-invariant on A, then the roots of $\mathcal{T}$ are delineable on A.*

More modern projection operators can achieve such a property with fewer polynomials. For example, the Lazard projector operator [8] satisfies the following theorem.

Theorem 2.2. *Let $\mathcal{T}$ be a non-empty set of non-zero real polynomials in r real variables, $r \geq 2$. Let A be a connected subset of $\mathbb{R}^{r-1}$. Let $\mathcal{P}$ be the Lazard projection of $\mathcal{T}$. If every element of $\mathcal{P}$ is Lazard valuation invariant on A, then the polynomials in $\mathcal{T}$ are Lazard analytic delineable on A.*

3. Adjacency and Clustering

In the theorems above, the projections satisfy neither cylindricity nor the semi-algebraic property over the region A: only connectedness is needed. In [1], Arnon took advantage of this by realizing that if two cells A and B are adjacent (i.e. their union is connected) and they share the same sign for all the relevant polynomials, then Theorem 2.1 could be applied to the union of those two cells $A \cup B$.

For example, in the CAD of $x^2 - y = 0$ depicted on in Figure 1, the blue two-dimensional cell and the purple one-dimensional cell have a connected union and share the same sign for the polynomial. This implies that in any further lifting phase we could cluster them and analyse them together rather than lifting over them separately. Moreover, the same reasoning applies for clustering the pink and the purple cell, and so on.

It is also possible to cluster together the two green one dimensional cells with the single point cell at the turning point of the curve. In total, we would be able to cluster the original 9 cells into 3 clusters, reducing by a third the effort needed to lift over this CAD.

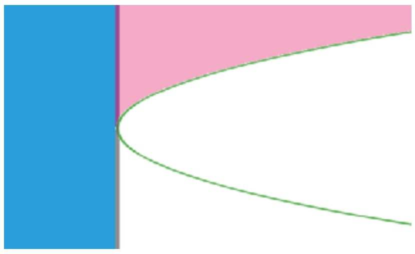

FIGURE 1. CAD of $x^2 - y = 0$ for the variable ordering $x \succ y$.

4. Complexity analysis

To give a complexity analysis of CAD we must understand how polynomial degrees grow during the projection phase. Given a set of polynomials that has a degree of most d in each of its variables, it is possible to show that the Lazard projection set of those polynomials has a degree of at most d^2 in each of its variables [6]. Likewise, it can be shown that the Collins projection set of those polynomials has a degree of at most d^6 in each.

This implies that if the original set of polynomials S_n has n variables and a degree of at most d on each of them, then the set S_i will have a degree of at most $d^{6^{n-i+1}}$ in each variable if Collins projection is used, and degree $d^{2^{n-i+1}}$ in each variable if Lazard projection is used.

The most expensive routine in the CAD algorithm is the real root isolation, and the number of root isolations needed to build a CAD is proportional to the number of cells generated. In order to study the worst case complexity of the CAD algorithm we can therefore+ study the maximum number of cells that can be generated.

It is important to note that if the number of cells of CAD_i is c_i and the sum of the total degrees of the biggest variable in S_{i+1} is d_{i+1} then the number of cells of CAD_{i+1} is $c_{i+1} \leq c_i(2d_{i+1} + 1)$ [6]. Applying this recursively, the maximum number of cells that can be generated including all the intermediate CADs is

$$\sum_{i=1}^{n} \prod_{j=1}^{i} (2d_j + 1),$$

resulting in an $\mathcal{O}(d^{\frac{6^n}{5}})$ worst case number of cells for Collins and $\mathcal{O}(d^{2^n})$ for Lazard.

However, using the upper bound of the number of sign-invariant connected components of a set of polynomials given in [2], it can be shown using Stirling's approximation that given a polynomial with a degree of at most d on each of its n variables the maximum number of sign-invariant clusters is of the order $\mathcal{O}(dn)^n$.

When clustering, the maximum number of cells that can therefore be generated including all the intermediate CADs is

$$2d_1 + 1 + \sum_{i=2}^{n} (2d_i + 1)(d_{i-1}i)^i,$$

resulting in $\mathcal{O}(d^{6^{n-1}})$ cells for Collins and $\mathcal{O}(d^{2^{n-1}})$ for Lazard.

5. Conclusion

The improvements, a reduction by one of the double exponent, might not look impressive at first glance, but are in fact significant in the context doubly-exponential growth. For example, in Lazard projection the maximum number of cells generated if clustering would be the square root of the maximum number of cells generated without using clustering. This means that for $n = 6$ we would build at most $4, 294, 967, 296$ cells when clustering compared to the $18, 446, 744, 073, 709, 551, 616$ without using clustering.

Furthermore, when clustering is not employed the number of cells in an intermediate CAD is a multiple of the number of cells in the previous CAD. However, when using clustering that is not necessarily the case, and the number of cells could decrease in some cases.

Finally, it is important to note that considering that the computations of adjacencies did not have a significant effect with respect to the cost of the CAD in [10], the usage of clustering to create a CAD would be very unlikely to slow down the process.

We hope this helps motivate the CAD community to bring Arnon's idea back to the forefront of CAD research, so that it can be coupled with the other achievements, and further push back the doubly-exponential wall of CAD!

References

1. Dennis S. Arnon, George E. Collins, and Scott McCallum, *Cylindrical Algebraic Decomposition II: An Adjacency Algorithm for the Plane*, SIAM Journal on Computing **13** (1984), no. 4, 878–889.
2. Sal Barone and Saugata Basu, *Refined Bounds on the Number of Connected Components of Sign Conditions on a Variety*, Discrete Comput Geom (2012), no. 47, 577–597.
3. Russell Bradford, James H. Davenport, Matthew England, Amir Sadeghimanesh, and Ali Uncu, *The DEWCAD Project: Pushing Back the Doubly Exponential Wall of Cylindrical Algebraic Decomposition*, ACM Communications in Computer Algebra **55** (2021), no. 3, 107–111.
4. Christopher W. Brown and James H. Davenport, *The complexity of quantifier elimination and cylindrical algebraic decomposition*, ISSAC (2007), 54–60.
5. George E. Collins, *Quantifier elimination for real closed fields by cylindrical algebraic decomposition*, Lecture Notes in Computer Science **33** (1975), 134–183.
6. Matthew England and James H. Davenport, *The complexity of cylindrical algebraic decomposition with respect to polynomial degree*, Computer Algebra in Scientific Computing (2016), 172–192.
7. Scott McCallum and George E. Collins, *Local box adjacency algorithms for cylindrical algebraic decompositions*, Journal of Symbolic Computation **33** (2002), no. 3, 321–342.
8. Scott McCallum, Adam Parusiński, and Laurentiu Paunescu, *Validity proof of Lazard's method for CAD construction*, Journal of Symbolic Computation **92** (2019), 52–69.
9. Gergely Röst and Amirhosein Sadeghimanesh, *Exotic Bifurcations in Three Connected Populations with Allee Effect*, International Journal of Bifurcation and Chaos **31** (2021), no. 13, 2150202.
10. Adam Strzeboński, *CAD adjacency computation using validated numerics*, ISSAC (2017), 413–420.
11. Stefan Takacs, *Using cylindrical algebraic decomposition and local Fourier analysis to study numerical methods: two examples*, SYNASC (2014), 42–49.
12. David Wilson, James H. Davenport, Matthew England, and Russell Bradford, *A "piano movers" problem reformulated*, SYNASC (2013), 53–60.

Coventry University, Coventry, UK.
Email address: `delriot@coventry.ac.uk, Matthew.England@coventry.ac.uk`