A Repository for CAD Examples

David J. Wilson, Russell J. Bradford, James H. Davenport
Department of Computer Science
University of Bath, Bath, England, BA2 7AY
`{D.J.Wilson,R.J.Bradford,J.H.Davenport}@bath.ac.uk`

Cylindrical Algebraic Decomposition (CAD, first introduced in [Col75]) of Euclidean space has become an important tool in mathematics and allows for practical quantifier elimination (QE) over the reals. Much research has gone into improving the projection operator (e.g. [McC85]), the use of partial CADs (e.g. [CH91]), or into alternative algorithms (e.g. [CDM+09]).

A problem of fundamentally high complexity (doubly-exponential in the number of variables [DH88]) it can be difficult to judge when a problem will be solved quickly, and easy to write down problems that are computationally infeasible. It can therefore be difficult to test new advances in this field, and even harder to 'experiment' with new ideas.

We have created a unified machine-readable repository of examples to be used when considering CADs. This allows for quick access to a host of examples, pulled from various theoretical questions and applications. As well as enabling more efficient research, the creation of the repository has prompted us to look deeper at certain questions related to CADs. The example bank currently facilitates quick input for Maple and Qepcad implementations of CAD and QE. Whilst CAD and QE procedures exist in other software, they are not directly involved in our research.

1 Structure of the Repository

The repository is freely available online, [Wil12], and consists of three parts (viewpoints), described below. The three files are structured for easy cross-referencing.

Reference file (PDF) This contains the most information about the examples. Where possible, a Tarski formula is given for the example (if not, a list of polynomials is provided) followed by a list of the free and quantified variables. If a suggested variable order was given in the original problem this is stated followed by the best achievable number of cells. Finally, any further notes are given (for example, if a problem seems infeasible) and the original source.

The following is a sample entry in the Example Bank:

Name: Parametric Parabola

$$(\exists\ x)\ [ax^2 + bx + c = 0]$$

Free Variables: a, b, c.
Quantified Variables: x.

Suggested variable order: $x > c > b > a$.

Best achievable number of cells: 27 - with MAPLE and variable ordering `[x,c,b,a]` (that is, $x > c > b > a$).

Notes: Without quantifying on x the problem can become rather simple; it is possible to construct CAD's containing only 3 cells.

Source: [CDM+09]

Maple (text) The MAPLE file should be `read` into MAPLE, after which running `Help()` provides information on subfunctions (corresponding to subsections in the reference file).

For example, the first entry in the example bank (from [CDM+09]) would be produced by calling `CMXYExamples(1);` and more information is obtained by running `CMXYExamplesInfo(1);`.

QEPCAD (text) The QEPCAD file is a basic text file from which entries can be copied and pasted into the command prompt within QEPCAD [Bro03].

The corresponding entry in the QEPCAD file is:

```
[Parametric Parabola]
(a,b,c,x)
3
(E x)[a x^2 + b x + c = 0].
```

2 Questions arising from the Repository

2.1 Formulation of the problem

A CAD or QE problem requires a formulation. There are many ways to do this and making such a choice inadvertently implies decisions about orders of variables and quantifiers.

Consequently, there are a range of 'levels' to a formulation. There is the completely abstracted problem being considered ('can this robot negotiate the corridor?', 'does this complex formula hold true?') which is not much use for calculation. There is then a question of how to translate this into a concrete problem; this produces a set of polynomials which can either be given as they are (to produce a full CAD) or with equality and logical symbols in a Tarski formula (for use in partial CAD). We can then reformulate the input to benefit performance but whilst still answering the original question. We then have a choice on the variable and quantifier ordering.

Which is the best to store in the repository? When do we consider an example to be 'distinct' from another — would the same problem with different variable orderings be considered distinct? Ideally we would store all possible formulations of the problem but this is, of course, impractical and many formulations would never be used.

The reference and QEPCAD files store the 'most' information possible by giving a Tarski formula (with an implied variable ordering). However, due to the MAPLE algorithm [CDM+09] solving only CAD problems (not QE) we simply give it a list of polynomials (without any logical or equality symbols).

2.2 Meaning of 'Smallest' possible CAD

One field we wanted to include in our repository was 'Minimal Cells in a CAD' to act as a benchmark for testing. However we soon realised that 'minimal' was too vague.

The definition of minimal fundamentally depends on what level of formulation your problem is given, and the related choices at each particular level. Once these have been decided, there is the distinction between the theoretical minimal number of cells in a CAD for that problem and an achievable minimal (through our current algorithms). Although the theoretical minimum would give us a good target to aim for, there is no easy method to calculate such a minimum, especially in high-dimensional problems. Therefore the 'achievable' minimum should be used, although this depends on the algorithms and tools available (currently we only investigate for Maple and Qepcad). The problem of actually calculating the quantity in question is difficult and time consuming.

Therefore, any quantities in this column are to be taken simply as a guideline for the general complexity of a CAD produced. Most importantly, any such value should be accompanied by the method used to obtain it.

3 Conclusion

CAD is used to address a variety of problems, in a variey of formulations. Often, the output of CAD is the input to a further process, so we may be interested in the smallest output, as well as the fastest time. Given the impact of the precise formulation on both these questions, we hope that a coheret benchmark set will assist the development of this field. Suggestions for further examples should be sent to the authors.

References

[Bro03] C. W. Brown, QEPCAD B: a program for computing with semi-algebraic sets using CADs *ACM SIGSAM Bulletin* 37(4):97–108, 2003.

[CDM+09] C. Chen, M. M. Maza, B. Xia and L. Yang, Computing Cylindrical Algebraic Decomposition via Triangular Decomposition *ISSAC '09*, 95–102, 2009.

[Col75] G. E. Collins, Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition *Automata Theory and Formal Languages 2nd GI ...*, 134–183, 1975.

[CH91] G. E. Collins and H. Hong, Partial Cylindrical Algebraic Decomposition for quantifier elimination *Journal of Symbolic Computation*, 12(3):299–328, 1991.

[DH88] J. H. Davenport and J. Heintz, Real Quantifier Elimination is Doubly Exponential *Journal of Symbolic Computation* 5(1):29–35, 1988.

[McC85] S. McCallum, An Improved Projection Operation for Cylindrical Algebraic Decomposition *EUROCAL '85* 277–278, 1985.

[Wil12] D. J. Wilson, Real Geometry and Connectedness via Triangular Descriuption: CAD Example Bank. `http://opus.bath.ac.uk/29503`, 2012.