Information Processing Letters 111 (2011) 941–944

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Efficient evaluation of specific queries in constraint databases ☆

Rafael Grimson [a,d], Joos Heintz [b,c,*], Bart Kuijpers [d]

[a] Departamento de Matemática, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires and CONICET, Ciudad Univ. Pab. I, 1428 Ciudad Autónoma de Buenos Aires, Argentina
[b] Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires and CONICET, Ciudad Univ. Pab. I, 1428 Ciudad Autónoma de Buenos Aires, Argentina
[c] Departamento de Matemáticas, Estadística y Computación, Facultad de Ciencias, Universidad de Cantabria, 39071 Santander, Spain
[d] Database and Theoretical Computer Science Research Group, Hasselt University, Agoralaan, Gebouw D, 3590 Diepenbeek, Belgium

ARTICLE INFO

Article history:
Received 23 June 2010
Received in revised form 28 April 2011
Accepted 29 June 2011
Available online 6 July 2011
Communicated by J. Chomicki

Keywords:
Constraint databases
Query evaluation
Computational complexity

ABSTRACT

Let $F_1, \ldots, F_s \in \mathbf{R}[X_1, \ldots, X_n]$ be polynomials of degree at most d, and suppose that $F_1, \ldots, F_s$ are represented by a division free arithmetic circuit of non-scalar complexity size L. Let $\mathcal{A}$ be the arrangement of $\mathbf{R}^n$ defined by $F_1, \ldots, F_s$.

For any point $x \in \mathbf{R}^n$, we consider the task of determining the signs of the values $F_1(x), \ldots, F_s(x)$ (sign condition query) and the task of determining the connected component of $\mathcal{A}$ to which x belongs (point location query). By an extremely simple reduction to the well-known case where the polynomials $F_1, \ldots, F_s$ are affine linear (i.e., polynomials of degree one), we show first that there exists a database of (possibly enormous) size $s^{\mathcal{O}(L+n)}$ which allows the evaluation of the sign condition query using only $(Ln)^{\mathcal{O}(1)} \log(s)$ arithmetic operations. The key point of this paper is the proof that this upper bound is almost optimal.

By the way, we show that the point location query can be evaluated using $d^{\mathcal{O}(n)} \log(s)$ arithmetic operations. Based on a different argument, analogous complexity upper-bounds are exhibited with respect to the bit-model in case that $F_1, \ldots, F_s$ belong to $\mathbf{Z}[X_1, \ldots, X_n]$ and satisfy a certain natural genericity condition. Mutatis mutandis our upper-bound results may be applied to the sparse and dense representations of $F_1, \ldots, F_s$.

© 2011 Elsevier B.V. All rights reserved.

1. Basic notions and notations

Throughout this paper we shall use the following notation: let n, s, d, h and L be given natural numbers, let $X_1, \ldots, X_n$ be indeterminates over $\mathbf{R}$, let $X := (X_1, \ldots, X_n)$ and let $F_1, \ldots, F_s \in \mathbf{R}[X_1, \ldots, X_n]$ be given polynomials of degree at most d.

In case that $F_1, \ldots, F_s$ belong to $\mathbf{Z}[X_1, \ldots, X_n]$, we suppose that the maximal bit-length of their coefficients (i.e. the logarithmic height of $F_1, \ldots, F_s$) is bounded by h.

Furthermore, we assume that the polynomials $F_1, \ldots, F_s$ are represented by (the output nodes of) an arithmetic circuit β in $\mathbf{R}[X]$ with inputs $X_1, \ldots, X_n$ such that β satisfies the following conditions:

- β uses only $\mathbf{R}$-linear operations and multiplications of polynomials (i.e., β is *division-free*);
- β contains at most L multiplications of polynomials which are counted at unit costs, whereas $\mathbf{R}$-linear operations are free (i.e., the non-scalar size of β is at most L).

☆ Research partially supported by the following Argentinian, Belgian and Spanish grants: CONICET PIP 2461/02, UBACYT X-098, PICT-2006-02067, MTM 2007-62799, FWO G.0344.05.

* Corresponding author at: Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires and CONICET, Ciudad Univ. Pab. I, 1428 Ciudad Autónoma de Buenos Aires, Argentina.
E-mail addresses: rgrimson@dm.uba.ar (R. Grimson), joos@dc.uba.ar (J. Heintz), bart.kuijpers@uhasselt.be (B. Kuijpers).

0020-0190/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.ipl.2011.06.015

Observe that for $k \in \mathbf{N}$ the dense and k-sparse encodings of $F_1, \ldots, F_s$ are special instances of the circuit representation of polynomials, with $L = \mathcal{O}(d^n)$ and $L = \mathcal{O}(kd)$ in each case, respectively.

A sign condition $\sigma \in \{-1, 0, 1\}^s$, with $\sigma = (\sigma_1, \ldots, \sigma_s)$ determines a polynomial inequality system of the form

$$\bigwedge_{1 \leqslant i \leqslant s} sign(F_i) = \sigma_i. \tag{1.1}$$

We call σ *consistent* if there exists a point $x \in \mathbf{R}^n$ satisfying the condition (1.1).

The consistent sign conditions on the polynomials $F_1, \ldots, F_s$ define a semi-algebraic partition $\mathcal{A} := \mathcal{A}(F_1, \ldots, F_s)$ of $\mathbf{R}^n$, called an *arrangement*. Suppose now that there is given another semi-algebraic partition $\mathcal{P} := \mathcal{P}(F_1, \ldots, F_s)$ of $\mathbf{R}^n$ which refines $\mathcal{A}$ and depends only on $F_1, \ldots, F_s$. Then $\mathcal{P}$ defines a query $Q_{\mathcal{P}}(F_1, \ldots, F_s)$ which determines, for each $x \in \mathbf{R}^n$ the (unique) element $P \in \mathcal{P}$ with $x \in P$.

In case $\mathcal{P} = \mathcal{A}$, we call $Q_{\mathcal{P}}(F_1, \ldots, F_s)$ the *sign condition query* for $F_1, \ldots, F_s$. If $\mathcal{P}$ is the partition of $\mathbf{R}^n$ into the connected components of (the elements of) $\mathcal{A}$, we call $Q_{\mathcal{P}}(F_1, \ldots, F_s)$ the *point location* query for $F_1, \ldots, F_s$.

If $F_1, \ldots, F_s$ are affine-linear and represented by their coefficients, the sign condition and the point location query may be evaluated using $\mathcal{O}(n^5 \log(s))$ arithmetic operations in $\mathbf{R}$ (see [1–3]).

The algorithm of [2] may be interpreted as follows: A first preprocessing yields a constraint database represented by an algebraic computation tree Γ of size $s^{\mathcal{O}(n)}$ which allows the evaluation of the sign condition query. While the size of the database (and the cost of its construction) are not taken into account, the evaluation of the sign condition query can be performed in time $\mathcal{O}(n^5 \log(s))$.

In this paper we analyze the complexities of the sign condition and point location queries under a constraint database point of view. In the same spirit as the arithmetic circuit representation of polynomials, we use algebraic computation trees to represent constraint databases (see [4–6] for the background on constraint databases and algebraic complexity theory).

In the next section we are going to expose our results, indicating in the most simple cases also the proofs. As part of these results, we present an algorithm that performs the evaluation of the sign condition query for $F_1, \ldots, F_s$ in time $(Ln)^{\mathcal{O}(1)} \log(s)$ and we show that this upper-bound is tight.

In the particular case that the polynomials $F_1, \ldots, F_s$ are given in dense representation, our method evaluates the sign condition query in time $d^{\mathcal{O}(n)} \log(s)$. By a mathematically different and more sophisticated method, a similar upper bound has already been derived in [7] for the coarser algorithmic model of *algebraic decision trees* (where the evaluation of any polynomial is free).

However, restricting our attention to computational models where arithmetic operations have a cost, the previous best time-upper-bound for the evaluation of this query is based on an application of general cylindrical algebraic decomposition method and results in a time bound that depends doubly-exponentially on n (see [8] and [9] for a discussion of this issue).

2. Results

2.1. Upper complexity bounds. Algorithms

Our first result relies on an extremely simple reduction argument to the case treated in [2], namely when the polynomials $F_1, \ldots, F_s$ are affine-linear.

Theorem 2.1. *Suppose that $F_1, \ldots, F_s$ can be evaluated by a division-free arithmetic circuit of non-scalar size L.*

Then there exists a constraint database represented by an algebraic computation tree of size $s^{\mathcal{O}(L+n)}$ such that in this database the sign condition query for $F_1, \ldots, F_s$ can be evaluated by this database in time $\mathcal{O}((L+n)^5 \log(s)) = (Ln)^{\mathcal{O}(1)} \log(s)$ (counting arithmetic operations and comparisons at unit costs).

Proof. Let $Y = (Y_1, \ldots, Y_L)$ be new indeterminates. From [5], Section 9 and [6], Exercise 9.18 we deduce that there exist polynomials $G_1, \ldots, G_L \in \mathbf{R}[X]$ and polynomials of degree one $H_1, \ldots, H_s \in \mathbf{R}[Y]$ having the following properties:

$G := (G_1, \ldots, G_L)$ can be evaluated using L polynomial multiplications and therefore, counting also $\mathbf{R}$-linear operations at unit costs, by means of $\mathcal{O}((L+n)^2)$ arithmetic operations. Furthermore the condition $F_1 = H_1(G), \ldots, F_s = H_s(G)$ is satisfied.

For any $x \in \mathbf{R}^n$, we compute first $G(x)$ and apply then the algorithm of [2] to $H_1, \ldots, H_s$ and the input $G(x)$ in order to determine the signs of $F_1(x), \ldots, F_s(x)$. □

If for $k \in \mathbf{N}$ the polynomials $F_1, \ldots, F_s$ are k-sparse, Theorem 2.1 remains correct when we replace the parameter L by kd.

Suppose now that the polynomials $F_1, \ldots, F_s$ are given in dense representation. Then Theorem 2.1 implies immediately that the sign condition query for $F_1, \ldots, F_s$ can be evaluated in time $d^{\mathcal{O}(n)} \log(s)$.

Taking into account that the connected components of the arrangement $\mathcal{A}(F_1, \ldots, F_s)$ are definable by $s^n d^{\mathcal{O}(n^4)}$ polynomials of $\mathbf{R}[X]$ of degree at most $d^{\mathcal{O}(n^3)}$ (see *e.g.*, [10], Theorem 16.18), we obtain finally the following result.

Corollary 2.2. *The sign condition and the point location queries for $F_1, \ldots, F_s$ can be evaluated in time $d^{\mathcal{O}(n)} \log(s)$ by a constraint database consisting of an algebraic computation tree of size $s^{\mathcal{O}(d^n)}$.*

We are now going to consider the sign condition and the point location queries under the aspect of their bit-complexity. For this purpose we assume that $F_1, \ldots, F_s$ are polynomials of $\mathbf{Z}[X]$ of logarithmic height at most h.

Furthermore we suppose that the family $F_1, \ldots, F_s$ is *generic* in the following sense: for any $1 \leqslant r \leqslant n$ and any $1 \leqslant i_1 < \cdots < i_r \leqslant s$ the polynomials $F_{i_1}, \ldots, F_{i_r}$ form a regular sequence in $\mathbf{Q}[X]$ or generate the trivial ideal.

By a considerable more elaborated method, based on arithmetic arguments from effective algebraic geometry [10], we obtain the following result in the spirit of [1].

Theorem 2.3. *There exists a constant $c \in \mathbf{N}$ such that for $\delta := 2^{-hd^{cn^2}}$ any hypercube $R \subset [0,1]^n$ of side length δ has the following property: at most n of the polynomials $F_1, \ldots, F_s$ change their signs in R.*

By means of Theorem 2.3 we are able to construct a constraint database $\mathcal{D}$ which consists of the partition of $[0,1]^n$ into hypercubes of side length δ. To each of these hypercubes we assign the indices and the signs of the at least $s-n$ polynomials among $F_1, \ldots, F_s$ which do not change their signs on it (observe that this preprocessing requires a consistency test for semi-algebraic constraint sets; see *e.g.* [11]). We obtain now the following statement.

Corollary 2.4. *For any real algebraic input point $x \in [0,1]^n$ of degree d' and logarithmic height h', the sign condition query for $F_1, \ldots, F_s$ can be evaluated in the database $\mathcal{D}$ using $(h + h')d^{\mathcal{O}(n^3)}d'^{\mathcal{O}(n)}$ bit operations. The database $\mathcal{D}$ can be represented by an algebraic computation tree of size $2^{hd^{\mathcal{O}(n^3)}}$.*

As remarked in Section 1, simple minded and unspecific applications of geometric elimination methods lead in Corollary 2.2 and Corollary 2.4 to complexity bounds which are doubly exponential in n.

2.2. Lower complexity bounds for the evaluation of sign condition queries

We may now ask whether, and in which sense, the runtime of our query evaluation algorithms may be improved.

In order to clarify this question, we are going to exhibit two examples which certify, in worst case and up to a polynomial expression in n, L and $\log(s)$, the optimality of the evaluation algorithms of [2] and Section 2.1 for the sign condition query. We limit ourselves to the algebraic complexity model. First, let us state the following technical result.

Lemma 2.5. *Let be given an algebraic computation tree Γ which realizes the evaluation of the sign condition query for $F_1, \ldots, F_s$. Then Γ satisfies the following conditions.*

(i) *The branching complexity of Γ is at least* $\log_3(\#\mathcal{A}(F_1, \ldots, F_s))$.
(ii) *Suppose that the polynomials $F_1, \ldots, F_s$ are irreducible, defining each a hypersurface (i.e., a codimension one subvariety) of $\mathbf{R}^n$. Then there exists for each $1 \leqslant i \leqslant s$ a branching node of Γ which tests the sign of a polynomial of $\mathbf{R}[X]$ which is a multiple of F_i.*

Example 1. Assume $s > n^2$ and that $F_1, \ldots, F_s$ are generic linear forms of $\mathbf{R}[X]$ represented by their coefficients. Then the evaluation of the sign condition query for $F_1, \ldots, F_s$ (by means of an algebraic decision tree) requires at least $n\frac{\log_3(s)}{2} = \Omega(n \cdot \log(s))$ arithmetic operations.

Proof. From the genericity of $F_1, \ldots, F_s$ we deduce $\#\mathcal{A}(F_1, \ldots, F_s) \geqslant \binom{s}{n}$. Let Γ be an algebraic computation tree that realizes the evaluation of the sign condition query for $F_1, \ldots, F_s$. Lemma 2.5(i) implies that the branching complexity of F is at least $\log_3(\#\mathcal{A}(F_1, \ldots, F_s)) \geqslant \log_3\left(\binom{s}{n}\right) \geqslant n(\log_3(s) - \log_3(n)) \geqslant n\frac{\log_3(s)}{2} = \Omega(n \cdot \log(s))$. □

We remark that [2] implies for Example 1 the upper complexity bound of $\mathcal{O}(n^5 \log(s)) = \mathcal{O}((n\log(s))^5)$ in the model of *linear* decision trees. We observe that this upper bound is polynomial with respect to the lower bound of $\Omega(n\log(s))$ in the more general setting of *algebraic* computation trees.

Example 2. Let $n \geqslant 2$ and $s > n^2$. We extend the family of linear polynomials $F_1, \ldots, F_s$ of Example 1 by the polynomial $F_0 := X_1^{2^L} - X_2$ and suppose that $F_0, \ldots, F_s$ are represented by a division-free arithmetic circuit in $\mathbf{R}[X]$ of non-scalar size L. Then the evaluation of the sign condition query for $F_0, \ldots, F_s$ requires $\Omega(L + n\log(s))$ arithmetic operations and comparisons.

Proof. We observe that F_0 is an irreducible polynomial of $\mathbf{R}[X]$ that defines a hypersurface of $\mathbf{R}^n$. Let Γ be an algebraic computation tree that realizes the evaluation of the sign condition query for $F_0, \ldots, F_s$. From Lemma 2.5(ii) we deduce that Γ contains a branching node which tests the sign of a polynomial $G \in \mathbf{R}[X]$ which is a multiple of F_0. Therefore the degree of G is at least 2^L and hence the evaluation of G requires at least L arithmetic operations (see [6], Section 8.1).

On the other hand, the choice of $F_1, \ldots, F_s$ implies by Example 1 that the branching complexity of Γ is at least $n\frac{\log_3(s)}{2}$. Thus Γ contains a computation path of length not less than $\max\{L, n\frac{\log_3(s)}{2}\} = \Omega(L + n\log(s))$. □

We observe that Theorem 2.1 implies for Example 2 the upper complexity bound of $\mathcal{O}((L+n)^5 \log(s)) = \mathcal{O}((L + n\log(s))^6)$. This upper bound is therefore polynomial in the lower bound $\Omega(L + n\log(s))$.

The results of this paper and their full proofs are contained in the PhD thesis of the first author [9].

References

[1] F. Meyer auf der Heide, A polynomial linear search algorithm for the n-dimensional knapsack problem, J. ACM 31 (1984) 668–676.
[2] S. Meiser, Point location in arrangements of hyperplanes, Inf. Comput. 106 (1993) 286–303.
[3] D. Liu, A note on point location in arrangements of hyperplanes, Inf. Process. Lett. 90 (2004) 93–95.
[4] G. Kuper, L. Libkin, J. Paredaens, Constraint Databases, Springer-Verlag, 2000.
[5] V. Strassen, Algebraic complexity theory, in: Handbook of Theoretical Computer Science, vol. A: Algorithms and Complexity (A), 1990, pp. 633–672.
[6] P. Bürgisser, M. Clausen, M.A. Shokrollahi, Algebraic Complexity Theory, Grundlehren der mathematischen Wissenschaften, Springer, 1997.
[7] D. Grigoriev, Topological complexity of the range searching, J. Complexity 16 (2000) 50–53.
[8] B. Chazelle, M. Sharir, An algorithm for generalized point location and its applications, J. Symb. Computation 10 (1990) 281–309.

[9] R. Grimson, Upper and lower complexity bounds for some problems in elementary geometry, Ph.D. thesis, Hasselt University, Department of Mathematics, Physics & Computer Science, 2010.
[10] S. Basu, R. Pollack, M.F. Roy, Algorithms in Real Algebraic Geometry, 2nd edn., Algorithms and Computation in Mathematics, vol. 10, Springer-Verlag, Secaucus, NJ, USA, 2006.
[11] J.F. Canny, Improved algorithms for sign determination and existential quantifier elimination, The Computer Journal 36 (1993) 409–418.