A Strong and Easily Computable Separation Bound for Arithmetic Expressions Involving Square Roots

C. Burnikel* R. Fleischer* K. Mehlhorn* S. Schirra*

Abstract

We consider arithmetic expressions over operators $+$, $-$, $*$, $/$, and $\sqrt{\ }$, with integer operands. For an expression E, a separation bound $sep(E)$ is a positive real number with the property that $E \neq 0$ implies $|E| \geq sep(E)$. We propose a new separation bound that is easy to compute and stronger than previous bounds.

1 Introduction

The evaluation of a test in a computer program frequently amounts to determining the sign of an arithmetic expression, e.g., if p, q, and r are points in the plane then

$$\sqrt{(p_x - r_x)^2 + (p_y - r_y)^2} \leq \sqrt{(q_x - r_x)^2 + (q_y - r_y)^2)}$$

determines whether p is at least as close to r as q. Section 4.2 discusses a more substantial example.

We consider expressions with operators

$$+, -, *, /, \text{ and } \sqrt{\ },$$

and integer operands. This class of expressions is particularly relevant for geometric computations [4, 5, 9, 15, 16]. For an expression E we define quantities $u(E)$ and $l(E)$ by induction on the syntactic structure of E, in Table 1.

Note that $u(E)$ and $l(E)$ are simply the numerator and denominator of an expression obtained by replacing in E all subtractions by additions.

For an expression E we use $k(E)$ to denote the number of (distinct) square root operations in E, i.e., if E is written as an acyclic directed graph where the source nodes are labeled by integers and where the internal nodes are labeled by operators from $+$, $-$, $*$, $/$, and $\sqrt{\ }$ then k is the number of nodes labeled by $\sqrt{\ }$. We slightly abuse notation and denote by E an expression given by a dag as well as the value of the expression. An expression is called *division-free* if it contains no occurrences of the operator $/$. For such expressions $l(E) = 1$.

*Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany, Email: {`burnikel, rudolf, mehlhorn, stschirr`}`@mpi-sb.mpg.de`

Theorem 1.1.
a) If E is a division-free expression and $E \neq 0$ then

$$|E| \geq u(E)^{(1-2^{k(E)})}$$

and

$$|E| \leq u(E).$$

b) If E is any expression and $E \neq 0$ then

$$|E| \geq u(E)^{(1-2^{k(E)})}/4l(E)$$

and

$$|E| \leq 4u(E) \cdot l(E)^{(2^{k(E)}-1)}.$$

There are examples where the bound given by Theorem 1.1 is nearly optimal. Consider

$$E = (2^{2^k} + 1)^{1/2^k} - 2,$$

i.e., the number 2 is squared k times, 1 is added, square roots are taken k times, and finally 2 is subtracted. We have $u(E) = (2^{2^k}+1)^{1/2^k}+2 \leq 5$ and hence, $|E| \geq 5^{1-2^k}$ by Theorem 1.1. On the other hand, $|E| \leq 2^{-2^k}$ since

$$\begin{aligned}
1 &= 2^{2^k} + 1 - 2^{2^k} \\
&= [(2^{2^k}+1)^{2^{-1}} - 2^{2^{k-1}}] \cdot [(2^{2^k}+1)^{2^{-1}} + 2^{2^{k-1}}] \\
&= [(2^{2^k}+1)^{2^{-2}} - 2^{2^{k-2}}] \cdot [(2^{2^k}+1)^{2^{-2}} + 2^{2^{k-2}}] \cdot \\
&\quad\ [(2^{2^k}+1)^{2^{-1}} + 2^{2^{k-1}}] \\
&\ \vdots \\
&= [(2^{2^k}+1)^{1/2^k} - 2] \cdot \textstyle\prod_{i=1}^{k} [(2^{2^k}+1)^{2^{-i}} + 2^{2^{k-i}}] \\
&\geq [(2^{2^k}+1)^{1/2^k} - 2] \cdot 2 \cdot \textstyle\prod_{i=0}^{k-1} 2^{2^i} \\
&= 2^{2^k} \cdot [(2^{2^k}+1)^{1/2^k} - 2].
\end{aligned}$$

We give an elementary proof of Theorem 1.1 in Section 2, compare it to previous work in Section 3, and discuss its applications to computational geometry and the number type `real` of LEDA [3, 11] in Section 4. There is a significant amount of literature on separation bounds in computer algebra [6, 8, 13], see also [14], as well as in computational geometry [4, 5, 9, 15]. Some of the work in computer algebra has concentrated on root separation bounds for systems of polynomial equations

	$u(E)$	$l(E)$
integer n	$\|n\|$	1
$E_1 \pm E_2$	$u(E_1) \cdot l(E_2) + l(E_1) \cdot u(E_2)$	$l(E_1) \cdot l(E_2)$
$E_1 \cdot E_2$	$u(E_1) \cdot u(E_2)$	$l(E_1) \cdot l(E_2)$
E_1 / E_2	$u(E_1) \cdot l(E_2)$	$l(E_1) \cdot u(E_2)$
$\sqrt{E_1}$	$\sqrt{u(E_1)}$	$\sqrt{l(E_1)}$

Table 1: Recursive rules for $u(E)$ and $l(E)$

of which our problem is a special case. We will show in Section 3 that the bound of Theorem 1.1 is never worse than the general bounds and sometimes significantly better. The work in computational geometry has concentrated on providing root separation bounds for specific geometric predicates. For example, [4] discusses the predicate which decides for a circle passing through a given point and touching two given lines whether it intersects, touches, or misses a third given line and shows a separation bound of $2^{-(32b+O(1))}$ if all endpoints of the line segments involved have integer coordinates bounded by 2^b. In [5] this bound is improved to $2^{-(24b+O(1))}$. In [9] it is shown that a certain part of the reasoning in the two aforementioned papers can be specified as rewrite rules. In this paper we go one step further and show that an even larger part of the reasoning can be fully automatized. We have to admit, however, that some of our bounds are slightly weaker than those derived in [4, 5] for the specific expressions involved in incircle tests, see Section 4.

2 The Proof

The upper bound in part a) of Theorem 1.1 is fairly obvious.

Lemma 2.1. *If E is division-free then*

$$|E| \le u(E)\,.$$

Now we turn to the proof of the lower bound.

Lemma 2.2. *Let E be a division-free expression. If $E \neq 0$ then*

$$|E| \ge u(E)^{1-2^{k(E)}}.$$

Proof. We use induction on $k(E)$. If $k(E) = 0$ then the value of E is an integer. Thus, if $E \neq 0$ then $|E| \ge 1 = u(E)^0$.

We turn to the induction step. Let $\sqrt{B}$ be an outermost root in E, i.e., a root that is not nested within any other root of E. We rewrite E as $A\sqrt{B} + C$ where A and C do not involve $\sqrt{B}$. It is easy to define A and C inductively, see Table 2.

Claim 2.1.

$$u(E) = u(A\sqrt{B} + C) \ge \max(u(A\sqrt{B}), u(C)).$$

Proof. This is easily shown by induction on the structure of E. For example, if $E = E_1 \cdot E_2$ then $u(E) = u(E_1) \cdot u(E_2) \overset{\text{I.H.}}{=} u(A_1\sqrt{B} + C_1) \cdot u(A_2\sqrt{B} + C_2) = (u(A_1) \cdot u(\sqrt{B}) + u(C_1)) \cdot (u(A_2) \cdot u(\sqrt{B}) + u(C_2)) = (u(A_1) \cdot u(C_2) + u(A_2) \cdot u(C_1)) \cdot u(\sqrt{B}) + u(A_1) \cdot u(A_2) \cdot u(\sqrt{B})^2 + u(C_1) \cdot u(C_2) = u(A\sqrt{B} + C)$.

Claim 2.2. $u(A^2 \cdot B - C^2) \le u(E)^2$.

Proof. We have

$$\begin{aligned} u(A^2 \cdot B - C^2) &= u(A)^2 \cdot u(B) + u(C)^2 \\ &\le (u(A) \cdot u(\sqrt{B}) + u(C))^2 \\ &= u(E)^2. \end{aligned}$$

We now distinguish cases according to the signs of A and C.

Case 1: A and C have opposite signs. Then $A \cdot \sqrt{B} - C \neq 0$ and hence

$$A \cdot \sqrt{B} + C = (A^2 \cdot B - C^2)/(A \cdot \sqrt{B} - C).$$

By assumption $A \cdot \sqrt{B} + C \neq 0$ and hence $A^2 \cdot B - C^2 \neq 0$. Also, $k(A^2 \cdot B - C^2) \le k(E) - 1$. Thus, by induction hypothesis and Lemma 2.1

$$\begin{aligned} |A \cdot \sqrt{B} + C| &\ge u(A^2 \cdot B - C)^{1-2^{k(E)-1}} / u(E) \\ &\ge u(E)^{2\cdot(1-2^{k(E)-1})-1} \\ &= u(E)^{1-2^{k(E)}}. \end{aligned}$$

Case 2: A and C do not have opposite signs, i.e., they either have the same sign or one of them is zero. Note that $A \cdot \sqrt{B} + C \neq 0$ implies that either $A \cdot \sqrt{B}$ or C is non-zero. Thus,

$$|E| \ge \max(|A \cdot \sqrt{B}|, |C|).$$

Further, $k(A^2 \cdot B) \le k(E) - 1$ and $k(C) \le k(E) - 1$. If C is non-zero then, by induction hypothesis,

$$\begin{aligned} |C| &\ge u(C)^{1-2^{k(C)}} \\ &\ge u(C)^{1-2^{k(E)}} \\ &\ge u(E)^{1-2^{k(E)}} \end{aligned}$$

	$A(E)$	$C(E)$
integer n	0	n
$E_1 \pm E_2$	$A(E_1) \pm A(E_2)$	$C(E_1) \pm C(E_2)$
$E_1 \cdot E_2$	$A(E_1) \cdot C(E_2) + A(E_2) \cdot C(E_1)$	$A(E_1) \cdot A(E_2) \cdot B + C(E_1) \cdot C(E_2)$
$\sqrt{B}$	1	0
$\sqrt{E_1}$ with E_1 different from B	0	$\sqrt{E_1}$

Table 2: Inductive definition of $A(E)$ and $C(E)$

and if $A \cdot \sqrt{B}$ is non-zero then, by induction hypothesis applied to $A^2 \cdot B$,

$$\begin{aligned} |A \cdot \sqrt{B}| &= \sqrt{A^2 \cdot B} \\ &\geq (u(A^2 \cdot B)^{1-2^{k(A^2 B)}})^{1/2} \\ &\geq u(A^2 \cdot B)^{1/2 \cdot (1-2^{k(E)})} \\ &\geq u(E)^{(1-2^{k(E)})}. \end{aligned}$$

This completes the proof of part a) of Theorem 1.1. The proof of part b) of Theorem 1.1 has roughly the same outline as the proof of part a). It involves, however, a somewhat stronger refined definition of $u(E)$ and $l(E)$. Note that this refined definition is used in the proof only. For lack of space this part of the proof is omitted here.

The following alternative proof of part a) of Theorem 1.1 is based on suggestions by Johannes Blömer [2].

Let $A(X) = \sum_{i=0}^{n} a_i X^i = a_n \prod_{i=1}^{n}(X - \alpha_i)$ and $B(X) = \sum_{i=0}^{m} b_j X^j = b_m \prod_{i=1}^{m}(X - \beta_j)$. Let $res_X(A, B)$ be the resultants of A and B considered as polynomials in X. Resultants can be used to construct polynomials having roots $\alpha_i + \beta_j$, $\alpha_i - \beta_j$, and $\alpha_i \beta_j$ [10]:

$$(-1)^{nm} res_Y(A(X - Y), B(Y)) = a_n^m b_m^n \prod_{i=1}^{n} \prod_{j=1}^{m} (X - (\alpha_i + \beta_j))$$

$$(-1)^{nm} res_Y(A(X + Y), B(Y)) = a_n^m b_m^n \prod_{i=1}^{n} \prod_{j=1}^{m} (X - (\alpha_i - \beta_j))$$

$$(-1)^{nm} res_Y(Y^n A(X/Y), B(Y)) = a_n^m b_m^n \prod_{i=1}^{n} \prod_{j=1}^{m} (X - (\alpha_i \beta_j))$$

Furthermore, the polynomial $A(X^2)$ has roots $\sqrt{\alpha_1}, \ldots, \sqrt{\alpha_i}, \ldots, \sqrt{\alpha_n}, -\sqrt{\alpha_1}, \ldots, -\sqrt{\alpha_i}, \ldots, -\sqrt{\alpha_n}$. Let E be an expression over operators $+, -, *$ and $\sqrt{\ }$ and integral operands. Using resultants for operations $+, -, *$ and the replacement of X by X^2 for $\sqrt{\ }$ operations we can construct a polynomial

$$P_E(X) = \prod (X - \gamma_i)$$

having root E, starting with a simple polynomial of the from $X - z$ where z is an integer. An integral polynomial is called monic, if its leading coefficient is 1. If $A(X)$ and $B(X)$ are monic, then the polynomials $res_Y(\ldots A(\ldots), B(\ldots))$ used in the construction are monic as well. Obviously, $A(X^2)$ is monic, if $A(X)$ is monic. Since we start with monic polynomials $X - z$, the polynomial $P_E(X)$ is a monic integral polynomial.

By the discussion above we can get any root of $P_E(X)$ by replacing some of the radicals in E by their conjugates, i.e., replacing $\sqrt{\ldots}$ by $-\sqrt{\ldots}$. Since $u(E)$ is simply E where all subtractions are replaced by additions and integers by their absolute values, the absolute value of all roots of $P_E(X)$ is bounded by $u(E)$.

It is not hard to see that the minimal polynomial $M_E(X)$ of E has degree at most $2^{k(E)}$. Since $P_E(X)$ has the root E too, $M_E(X)$ divides $P_E(X)$, i.e.

$$M_E(X) = \prod_{j=1}^{\deg(M_E(X))} (X - \gamma_{i_j}).$$

We use the *measure* of a polynomial to get a lower bound on the absolute value of the roots of $M_E(X)$. The measure $mes(A(X))$ of a polynomial $A(X) = \sum_{i=0}^{n} a_i X^i = a_n \prod_{i=0}^{n}(X - \alpha_i)$ is defined by

$$mes(A(X)) = |a_n| \prod_{i=1}^{n} \max(1, |\alpha_i|)$$

By definition

$$|\alpha_i| \leq mes(A(X)).$$

Since

$$\begin{aligned} & mes(X^n A(1/X)) \\ = \;& mes(\sum_{i=0}^{n} a_{n-i} X^i) \end{aligned}$$

$$
\begin{aligned}
&= mes((a_n \prod_{i=1}^{n} \alpha_i) \prod_{i=1}^{n}(X - 1/\alpha_i)) \\
&= (|a_n| \prod_{i=1}^{n} |\alpha_i|) \cdot \prod_{i=1}^{n} \max(1, 1/|\alpha_i|) \\
&= |a_n| \prod_{i=1}^{n} \max(1, |\alpha_i|) \\
&= mes(A(X)),
\end{aligned}
$$

the measure gives a lower bound on the roots, too, namely

$$|\alpha_i| \geq mes(A(X))^{-1}.$$

Considering the measure $mes(M_E(X))$ of the minimal polynomial of E we get

$$|E| \geq \left(\prod_{i=1}^{\deg(M_E(X))} \max(1, |\gamma_{i_j}|) \right)^{-1}.$$

Remember that $|\gamma_{i_j}| \leq u(E)$ for all j. Either $|\gamma_{i_j}| > 1$ for all j and hence

$$|E| > 1 \geq u(E)^{1-2^{k(E)}},$$

because $u(E) \geq 1$, or at least one of the absolute values of the roots of $M_E(X)$ is at most 1 and hence

$$\prod_{i=1}^{\deg(M_E(X))} \max(1, |\gamma_{i_j}|) \leq u(E)^{2^{k(E)}-1}.$$

In either case we have shown

$$|E| \geq \left(u(E)^{2^{k(E)}-1}\right)^{-1} = u(E)^{1-2^{k(E)}}.$$

An extension of the proof to expressions involving divisions is not obvious.

However, the alternative proof can easily be extended to expressions involving r-th roots. Let E be an expression over the operators $+, -, *$ involving $k(E)$ root operations $\sqrt[r_1]{\ldots}$, $\ldots$, $\sqrt[r_{k(E)}]{\ldots}$. If we extend the definition of u by $u(\sqrt[r]{E_1}) = \sqrt[r]{u(E_1)}$ we get

$$|E| \geq u(E)^{1-d(E)}$$

where $d(E) = \prod_{i=1}^{k(E)} r_i$.

3 Comparison to Previous Work

There is a rich body of literature on root separation bounds [6, 8, 12, 13], the work by Mignotte [13, 12] and Canny [6] being particularly relevant for the present paper. Mignotte [12] considers the identification of algebraic numbers, say α and β, given as algebraic expressions. Looking for a "numerical proof" of the identity of α and β he discusses bounds on $|\alpha - \beta|$. In our case $\beta = 0$. Mignotte defines functions measuring the *size* of algebraic numbers. The definition of a size-function includes inequalities that relate the size of the sum and the product of algebraic numbers to the sizes of the operands. However, inequalities for square root operation and quotients are not given. The recursive rules given in Table 3 of quantities $M(E)$ and $deg(E)$ for an expression E are based on Section 5 of [12] and the measure of an algebraic number, cf. Section 4.3 in [13]. We have $|E| \geq M(E)^{-1}$ for all expressions E. We call this the *degree-measure bound*. For the expression $E = (2^{2^k} + 1)^{2^{-k}} - 2$ considered in the introduction the degree-measure bound is $8^{-2^k - 1/3}$ which is slightly worse than ours. The next lemma shows that this is not a coincidence.

Lemma 3.1. *For all expressions E we have*

$$
\begin{aligned}
M(E) &\geq u(E)^{\deg(E)}, \\
M(E) &\geq l(E)^{\deg(E)}, \text{ and} \\
\deg(E) &\geq 2^{k(E)}.
\end{aligned}
$$

Proof. We use structural induction.

If E is an integer then $M(E) = u(E) \geq 1 = l(E)$ and $\deg(E) = 1 = 2^0 = 2^{k(E)}$.

If $E = E_1 \pm E_2$ we have

$$
\begin{aligned}
& M(E_1 \pm E_2) \\
&= 2^{\deg(E_1)\cdot\deg(E_2)} \cdot M(E_1)^{\deg(E_2)} \cdot M(E_2)^{\deg(E_1)} \\
&\overset{\text{I.H.}}{\geq} 2^{\deg(E_1)\cdot\deg(E_2)} \cdot \max(u(E_1), l(E_1))^{\deg(E_1)\cdot\deg(E_2)} \cdot \\
&\quad \max(u(E_2), l(E_2))^{\deg(E_1)\cdot\deg(E_2)} \\
&= (2\max(u(E_1), l(E_1)) \max(u(E_2), l(E_2)))^{\deg(E_1)\deg(E_2)} \\
&\geq (u(E_1)\cdot l(E_2) + u(E_2)\cdot l(E_1))^{\deg(E_1)\cdot\deg(E_2)} \\
&= u(E)^{\deg(E)}
\end{aligned}
$$

and $M(E) \overset{\text{I.H.}}{\geq} (l(E_1)\cdot l(E_2))^{\deg(E_1)\cdot\deg(E_2)} = l(E)^{\deg(E)}$. Furthermore, we have $\deg(E) = \deg(E_1)\cdot\deg(E_2) \overset{\text{I.H.}}{\geq} 2^{k(E_1)}\cdot 2^{k(E_2)} \geq 2^{k(E)}$.

For $E = E_1 * E_2$ or $E = E_1/E_2$ the proof is analogous.

For $E = \sqrt{E_1}$ we have

$$
\begin{aligned}
M(E) &= M(E_1) \\
&\overset{\text{I.H.}}{\geq} u(E_1)^{\deg(E_1)} \\
&= \sqrt{u(E_1)}^{2\cdot\deg(E_1)} \\
&= u(E)^{\deg(E)}
\end{aligned}
$$

and analogously $M(E) \geq l(E)^{\deg(E)}$. For the degree bound note that $\deg(E) = 2\cdot\deg(E_1) \geq 2\cdot 2^{k(E_1)} = 2^{k(E)}$.

	$M(E)$	$\deg(E)$
integer n	$\lvert n\rvert$	1
$E_1 \pm E_2$	$2^{\deg(E_1)\cdot\deg(E_2)} \cdot M(E_1)^{\deg(E_2)} \cdot M(E_2)^{\deg(E_1)}$	$\deg(E_1)\cdot\deg(E_2)$
$E_1 * / E_2$	$M(E_1)^{\deg(E_2)} \cdot M(E_2)^{\deg(E_1)}$	$\deg(E_1)\cdot\deg(E_2)$
$\sqrt{E_1}$	$M(E_1)$	$2\cdot\deg(E_1)$

Table 3: Recursive rules for degree-measure bound

By Lemma 3.1

$$\begin{aligned} M(E) &\geq \max(u(E), l(E))^{\deg(E)} \\ &\geq \max(u(E), l(E))^{2^{k(E)}} \\ &\geq u(E)^{2^{k(E)}-1} \cdot l(E) \end{aligned}$$

and hence $M(E)^{-1} \leq u(E)^{1-2^{k(E)}}/l(E)$. Thus, our bound is never worse than the degree-measure bound for division-free expressions, and it is within a factor of 4 of the degree-measure bound for the general case.

Canny [6] considers systems of n polynomial equations in n unknowns. He shows that if $a_1,\ldots,a_n$ is any solution to such a system then

$$a_i = 0 \quad \text{or} \quad |a_i| \geq (3dc)^{-n\cdot d^n}$$

for all i, where d is the maximal degree of any polynomial in the system and c is the maximal coefficient appearing in any of the polynomials. Hong [8] had previously proven a similar bound for tridiagonal systems.

There are many ways to transform expression dags into polynomial systems, e.g., one may either introduce a variable for each node of the dag and then relate the input and output variables of each node by an equation, or one may introduce a variable for each square root and each division. For the expression $E = (2^{2^k}+1)^{2^{-k}} - 2$ considered in the introduction the two methods yield

$$\begin{array}{rcl} x_1 &=& 2 \\ x_2 &=& x_1^2 \\ &\vdots& \\ x_k &=& x_{k-1}^2+1 \\ y_k^2 &=& x_k \\ &\vdots& \\ y_1^2 &=& y_2 \\ y &=& y_1 - 2 \end{array} \qquad \text{and} \qquad \begin{array}{rcl} x_k &=& 2^{2^k}+1 \\ y_k^2 &=& x_k \\ &\vdots& \\ y_1^2 &=& y_2 \\ y &=& y_1 - 2 \end{array}$$

respectively, i.e., we have $n = 2k+1$, $d = 2$ and $c = 2$ in the former case, and $n = k+2$, $d = 2$, and $c = 2^{2^k}+1$ in the latter case. The bounds are therefore $12^{-(2k+1)\cdot 2^{2k+1}}$ and $(6\cdot(2^{2^k}+1))^{-(k+2)\cdot 2^{k+2}}$, respectively. Try $k = 3$ to see the difference between these bounds and our bound.

We show next that our bound is never worse than the bound derived from Canny's result for division-free expressions. Let $d \geq 2$ and c be integers and consider a system

$$x_i^2 = p_i(x_1,\ldots,x_{i-1}), 1 \leq i \leq k$$

where each p_i is a polynomial of degree at most d whose maximal coefficient is bounded by c. By writing

$$x_i = \sqrt{p_i(x_1,\ldots,x_{i-1})}$$

the system turns into a dag with k square root nodes. Let u_i be $u(x_i)$ as computed by the rules given in the introduction. Note that $k(x_i) = i$ for all i. If $(a_1,\ldots,a_k)$ is any solution of this system then

$$a_i = 0 \text{ or } |a_i| \geq (3dc)^{-i\cdot d^i}$$

according to Canny and

$$a_i = 0 \text{ or } |a_i| \geq u_i^{1-2^i} \geq u_i^{-2^i}$$

according to Theorem 1.1.

Lemma 3.2. *$u_i^{-2^i} \geq (3dc)^{-i\cdot d^i}$ for all i.*

Proof. It suffices to prove that

$$u_i \leq (3dc)^{i\cdot(d/2)^i}$$

for all i. We use induction on i. The first equation is $x_1^2 = c'$ for some $c' \leq c$. Thus, $u_1 = \sqrt{c'}$ and the inequality clearly holds for $i = 1$.

For the induction step we have

$$u_{i+1} \leq (c \cdot \sum_{l_1+\ldots+l_i \leq d} u_1^{l_1}\ldots u_i^{l_i})^{1/2}.$$

We now distinguish cases. For $d = 2$ we have $u_l \geq (6c)^l$ for $l \leq i$ by induction hypothesis and hence

$$\begin{aligned} u_{i+1} &\leq c^{1/2}\cdot(1+\textstyle\sum_{l=1}^{i} u_l\cdot(1+\sum_{j=1}^{l-1}u_j))^{1/2} \\ &\leq c^{1/2}\cdot(1+\textstyle\sum_{l=1}^{i}(6c)^l\cdot\sum_{j=0}^{l-1}(6c)^j)^{1/2} \\ &\leq c^{1/2}\cdot(1+\textstyle\sum_{l=1}^{i}(6c)^l\cdot(6c)^l/(6c-1))^{1/2} \\ &\leq (c\cdot(36c^2)^{i+1}/(6c-1))^{1/2} \\ &\leq (6c)^{i+1}. \end{aligned}$$

For $d \geq 3$ we have $u_l \leq (3dc)^{l\cdot(d/2)^l}$ for $l \leq i$ and hence

$$\begin{aligned} u_{i+1} &\leq (c \cdot (d+1)^i \cdot u_i^d)^{1/2} \\ &\leq (c \cdot (d+1)^i \cdot (3dc)^{i\cdot(d/2)^i \cdot d})^{1/2} \\ &\leq [c^{1/2} \cdot (d+1)^{i/2}/(3dc)^{(d/2)^{i+1}}] \cdot \\ &\quad (3dc)^{(i+1)\cdot(d/2)^{i+1}} \\ &\leq (3dc)^{(i+1)\cdot(d/2)^{i+1}} \end{aligned}$$

since $i/2 \leq (d/2)^{i+1}$ for $d \geq 3$ and $i \geq 1$.

4 Applications

Our motivation for considering separation bounds is their use for testing the sign of real-valued expressions. Let E be a real-valued expression and $\mathrm{sep}(E)$ a separation bound for E. We can then compute the sign of E, as follows. We compute an interval I_ϵ of length ϵ such that $0 < \epsilon < \mathrm{sep}(E)$ and I_ϵ contains E. If 0 is inside I_ϵ, the value of E is smaller than $\mathrm{sep}(E)$ and hence E is zero. Otherwise, the sign of E is equal to the sign of the numbers in I_ϵ.

LEDA [11] provides a number type `real` that uses the concept of separation bounds for testing the sign of an arithmetic expression over the operations $+$, $-$, $*$, $/$, $\sqrt{\ }$. Dubé and Yap [7] provide a similar number type called `bigExpression`. The procedure used in `bigExpression` for computing a separation bound is based on a bound, called *degree-length bound* in [7], which is almost identical to the degree-measure bound described in Section 3. Since the recursive rules for product and quotient involve an additional factor of $2^{\min(\deg(E_1),\deg(E_2))}$, the resulting bounds for expressions over operators $+, -, *, /$ and $\sqrt{\ }$ and integral operands are even weaker than the degree-measure bounds.

We implemented three procedures for computing separation bounds that use the bound of Theorem 1.1, the degree-measure bound, and a bound based on Canny's result, respectively. We call the bound based on Canny's result *polynomial system bound.*

4.1 Actual computation of separation bounds

Theorem 1.1 cannot be applied directly since it evolves quantities $u(E)$ and $l(E)$ that are in general non-rational algebraic numbers. We propose two methods to compute slightly weaker separation bounds, firstly computing with floating point numbers with rounding towards larger values and secondly maintaining integral logarithms of $u(E)$ and $l(E)$.

For the first method we assume the IEEE standard 754 for binary floating point arithmetic [1]. The IEEE standard 754 requires that the result of the floating point operations $+$, $-$, $*$, $/$, and $\sqrt{\ }$ are exactly rounded according to the chosen rounding mode. Moreover, the standard requires a rounding mode towards infinity, i.e. to larger values. Using this rounding mode we can compute upper bounds for $u(E)$ and $l(E)$ with floating point arithmetic.

Changing the rounding mode is expensive on some systems. As an alternative one can simply evaluate $u(E)$ and $l(E)$ in double precision floating point arithmetic and multiply each intermediate result by $(1 + 2^{-52})$. Note that underflow does not occur here since we always have $u(E) \geq 1$ whenever $u(E)$ is nonzero. In case of overflow, the resulting value of $u(E)$ will be infinity, which is a correct (but meaningless) bound.

In the second method we recursively compute integers $u'(E) \geq \log_2(u(E))$ and $l'(E) \geq \log_2(l(E))$ as shown in Table 4. The recursive rules for $u'(E)$ and $l'(E)$ do not evolve square root operations. All numbers arising in the computation of $u'(E)$ and $l'(E)$ are integers, and fairly small in practice. Hence rounding errors are not an issue for computing $u'(E)$ and $l'(E)$.

We implemented the second method and compared the results with the polynomial system bound and the degree-measure bound. In our implementation the degree-measure bound is maintained logarithmically as well, as suggested in [12]. Furthermore, instead of recursively computing $\deg(E)$ we use the estimate $2^{k(E)}$ to bound the algebraic degree of expression E. In the polynomial system bound we maintain bounds on the degree and coefficient size of the implicitly generated system of polynomial equations.

4.2 Application to Voronoi diagram computation

The following incircle tests arise in the computation of Voronoi diagrams of line segments and points. Let v be the Voronoi vertex having minimal distance to sites s_1, s_2 and s_3 and let s be another site. Is s inside the Voronoi circle of v, i.e., is it nearer to v than the sites s_1, s_2, s_3? In the implementation of the incircle test we compare the squared distance of the sites s_1, s_2 and s_3 to v with the squared distance of s to v.

For example let v be given by two segments s_1, s_2 and a point p and let s be a segment s_3. Let $l_i : a_i x + b_i y + c_i = 0$ be the supporting line of segment s_i. In [5] it is shown that for $p = (0,0)$ the incircle test includes a sign test of the form $E < 0$ for

$$E = (x_v^2 + y_v^2)(a_3^2 + b_3^2) - (a_3 x_v + b_3 y_v + c_3 z_v)^2$$

where the coordinates $(x_v/z_v, y_v/z_v)$ of vertex v are

$$\begin{aligned} x_v &= a_1c_2 + a_2c_1 \pm \sqrt{2c_1c_2(\sqrt{N} + (a_1a_2 - b_1b_2))} \\ y_v &= b_1c_2 + b_2c_1 \pm \sqrt{2c_1c_2(\sqrt{N} - (a_1a_2 - b_1b_2))} \\ z_v &= \sqrt{N} - (a_1a_2 + b_1b_2) \end{aligned}$$

	$u'(E)$	$l'(E)$
integer n	$\lceil \log_2(n) \rceil$	0
$E_1 \pm E_2$	$1 + \max(u'(E_1) + l'(E_2), l'(E_1) + u'(E_2))$	$l'(E_1) + l'(E_2)$
$E_1 \cdot E_2$	$u'(E_1) + u'(E_2)$	$l'(E_1) + l'(E_2)$
E_1 / E_2	$u'(E_1) + l'(E_2)$	$l'(E_1) + u'(E_2)$
$\sqrt{E_1}$	$\lceil u'(E_1)/2 \rceil$	$\lceil l'(E_1)/2 \rceil$

Table 4: Recursive rules for easily computable $u'(E)$ and $l'(E)$

incircle test	our bound	polynomial system bound	degree-measure bound
`ppp-p`	$0 \cdot b+0$	$4 \cdot b+17$	$92 \cdot b+39$
`ppp-l`	$0 \cdot b+0$	$8 \cdot b+42$	$202 \cdot b+91$
`ppl-p`	$6 \cdot b+14$	$64 \cdot b+312$	$2032 \cdot b+1034$
`ppl-l`	$12 \cdot b+27$	$96 \cdot b+680$	$8864 \cdot b+4586$
`llp-p`	$28 \cdot b+84$	$384 \cdot b+1728$	$156416 \cdot b+82584$
`llp-l`	$56 \cdot b+161$	$512 \cdot b+3840$	$1307136 \cdot b+697736$
`lll-p`	$70 \cdot b+147$	$512 \cdot b+3456$	$14188928 \cdot b+6165768$
`lll-l`	$84 \cdot b+168$	$640 \cdot b+4352$	$11404160 \cdot b+4762120$

Table 5: Comparison of separation bounds for incircle tests

and $N = (a_1^2 + b_1^2)(a_2^2 + b_2^2)$.

There are different types of incircle tests that correspond to the different types of sites defining v and s. s can have type p (point) or l (line segment) and v can have type ppp, ppl, llp, or lll where a symbol p stands for a point among the sites s_i and a symbol l stands for a line segment among the sites s_i. For example, the incircle test above has type llp-l.

Let us assume that input points have b bit integral Cartesian coordinates and line segments are given by two input points. Our main goal is to find the exact dependence of the separation bound of a test expression on the maximal bit size b of the integer coordinates. Note that for any of the methods discussed here the logarithm of the separation bound for expression E is a linear function $s(b) = s_E(b)$ of b. We split $s(b)$ into $s(b) = s_b b + s_c$ and compute the values s_b and s_c separately. s_b and s_c can be computed recursively by formulas that are analogous to the formulas given before. For the various incircle tests, we get the results shown in Table 5.

Compared with the bounds based on the results of Canny and Mignotte, our new separation bounds lead to a considerable improvement. As proven in Section 3, the polynomial system bounds are always weaker than ours but, at least in these examples, still usable for the sign computation. For the last four tests the degree-measure bounds are much too weak to be of any practical use. The reader should realize that computing just one single square root operation up to a million binary digits is not feasible in reasonable time on most computers.

Comparing Table 5 with the results of [5] which were computed by hand, we see that we get the same asymptotic bounds for the first four tests and slightly weaker bounds for the last four tests. Note that the test expressions for ppp-p and ppp-l are integral and hence 1 is an obvious separation bound in these cases. Only our method achieves this bound. On the other hand, there are several reasons why our results are not always optimal. In the test expression E for llp-l (and also for llp-p), we have three different square root operations. However the two nested roots $\sqrt{F}$ and $\sqrt{G}$ that appear in E are algebraically dependent over the rationals and, in particular, $\sqrt{F}\sqrt{G}$ is an integer. Hence E has in fact algebraic degree 2^2 at most and not $2^{k(E)} = 2^3$ as is implicitly assumed in the computation of our bound.

References

[1] *IEEE standard 754-1985 for binary floating-point arithmetic.* reprinted in SIGPLAN 22, 1987. 2:9-25.

[2] J. Blömer. Private communication, September 1996.

[3] C. Burnikel, K. Mehlhorn, and S. Schirra. The LEDA class real number. Technical Report MPI-I-96-1-001, Max-Planck-Institut für Informatik, Saarbrücken, January 1996.

[4] C. Burnikel, K. Mehlhorn, and S. Schirra. How to Compute the Voronoi Diagram of Line Segments: Theoretical and Experimental Results. In *Proceedings of ESA '94*, LNCS 855, Springer Verlag, pages 227–239, 1994.

[5] C. Burnikel. Exact Computation of Voronoi Diagrams and Line Segment Intersections. PhD thesis, Max-Planck-Institut für Informatik, Saarbrücken, April 1996.

[6] J.F. Canny. *The Complexity of Robot Motion Planning*. The MIT Press, 1987.

[7] T. Dubé, C.K. Yap, *A Basis for Implementing Exact Geometric Algorithms*. October, 1993.

[8] J.W. Hong. Proving by Example and Gap Theorem. In *Symposium on Foundations of Computer Science*, 1987.

[9] G. Liotta, F.P. Preparata, and R. Tamassia. Robust Proximity Queries in Implicit Voronoi Diagrams. Technical Report RI 02912-1910, Center for Geometric Computing, Department of Computer Science, Brown University, May 1996.

[10] R. Loos. Computing in algebraic extensions. In B. Buchberger, G.E. Collins, and R. Loos, editors, *Computer Algebra*, pages 173–187. Springer Verlag, 1982.

[11] K. Mehlhorn and S. Näher. LEDA: A platform for combinatorial and geometric computing. *Communications of the ACM*, 38(1):96–102, 1995. cf. `http://www.mpi-sb.mpg.de/LEDA/leda.html`.

[12] M. Mignotte. Identification of Algebraic Numbers. *Journal of Algorithms*, 3(3), September 1982.

[13] M. Mignotte. *Mathematics for Computer Algebra*. Springer Verlag, 1992.

[14] C.K. Yap. *Fundamental Problems in Algorithmic Algebra*. Princeton University Press, 1996 (to appear).

[15] C.K. Yap. Towards Exact Geometric Computation. In *Proceedings of CCCG'93*, pages 405–419, 1993.

[16] C. K. Yap and T. Dubé. The exact computation paradigm. In D.Z. Du and F. Hwang, editors, *Computing in Euclidean Geometry*, pages 452–492. World Scientific Press, 1995. 2nd edition.