Robust Geometric Computation

Kurt Mehlhorn

MPI für Informatik
Saarbrücken
Germany

Approaches to Robust Geometric Computation

- robust (correct) = our implementations conform to their specifications
- see http://cs.nyu.edu/exact/resource/ for many pointers
- approaches
  - exact computation paradigm: make sure that geometric primitives are evaluated correctly
    - exact number types for correctness
    - floating point filters for speed
  - controlled perturbation: as above, but run algorithm on a slightly perturbed input to reduce arithmetic demand
  - learn to live with inexact primitives and modify algs so that they work nevertheless
    - topological consistency
    - new algorithms
  - I have personally pursued the first two approaches, but not the third. Mistake?

Exact Computation Paradigm

- many geometric predicates are defined as the sign of an arithmetic expression
- recall: orientation of three points = sign of a 3×3 determinant
- approach: compute with exact arithmetic, e.g., bigints, rationals, algebraic numbers, ...
- this works, but is slow
- floating point filter:
  - evaluate with floating point arithmetic and check whether it gives the right sign.
  - if not: resort to exact arithmetic

Exact Number Types

MAX-PLANCK-GESELLSCHAFT

- *bigints:* arbitrary precision integers, there are many packages available, e.g., GMP (Gnu multi-precision), LEDA, ... Most of them use Karatsuba-Offman multiplication, running time $O(L^{\log 3})$ for L-bit numbers
- *bigrationals:* quotients of bigints,
  - arithmetic as taught in high-school
  - it is important to reduce fractions regularly, i.e., to cancel out common factors
  - common factors can be found by gcd-computation, Euclid's algorithm
  - sometimes common factors are known beforehand, e.g.,

$$\frac{a}{2^n} + \frac{b}{2^m} = \frac{a2^m + b2^n}{2^{n+m}}$$

  - pragmatic suggestion: use gcd-computation to cancel out common factors; if in a certain expression, cancelation is always possible start to think

An Example of Cancelation: Gaussian Elimination

$$\begin{pmatrix} a & b & e \\ c & d & f \\ g & h & i \end{pmatrix} \to \begin{pmatrix} a & b & e \\ 0 & d-b(c/a) & f-e(c/a) \\ 0 & h-b(g/a) & i-e(g/a) \end{pmatrix}$$

$$\to \begin{pmatrix} a & b & e \\ 0 & (ad-bc)/a & (af-ec)/a) \\ 0 & (ah-bg)/a & (ai-eg)/a \end{pmatrix}$$

$$\to \begin{pmatrix} a & b & e \\ 0 & (ad-bc)/a & (af-ec)/a) \\ 0 & 0 & (ai-eg)/a - \frac{(ah-bg)/a}{(ad-bc)/a}(af-ec)/a \end{pmatrix}$$

$$(ai-eg)/a - \frac{(ah-bg)/a}{(ad-bc)/a}(af-ec)/a = \frac{(ai-eg)(ad-bc)-(ah-bg)(af-ec)}{a(ad-bc)}$$

$$= \frac{a(\ldots)+(egbc-bgec)}{a(ad-bc)} = \frac{\ldots}{ad-bc}$$

Homogeneous Coordinates

MAX-PLANCK-GESELLSCHAFT

- use of rational numbers can often be avoided by the use of homogeneous coordinates
- homogeneous coordinates = same denominator for all coordinates
- in $\mathbb{R}^2$: the homogeneous point (X,Y,W) with $X,Y,W \in \mathbb{Z}$, $W > 0$, represents the Cartesian point $(X/W, Y/W)$
- let $p = (p_x, p_y) = (p_X, p_Y, p_W)$ and similarly for q and r. Then

$$orient(p,q,r) = sign \begin{vmatrix} p_x & p_y & 1 \\ q_x & q_y & 1 \\ r_x & r_y & 1 \end{vmatrix} = sign \begin{vmatrix} p_X/p_W & p_Y/p_W & 1 \\ q_x & q_y & 1 \\ r_x & r_y & 1 \end{vmatrix}$$

$$= sign \begin{vmatrix} p_X & p_Y & p_W \\ q_x & q_y & 1 \\ r_x & r_y & 1 \end{vmatrix} = sign \begin{vmatrix} p_X & p_Y & p_W \\ q_X & q_Y & q_W \\ r_X & r_Y & r_W \end{vmatrix},$$

i.e., predicates can be expressed directly in homogenous coordinates

Exact Number Types Beyond Rationals

- algrebraic expressions: expressions with integer operands and operators $+$, $-$, $\times$, $/$, and $\sqrt[k]{}$.
- algebraic numbers = roots of polynomials with integer coefficients
- $\sin$, $\cos$, $\ln$, $\exp$, ...
- complexity of deciding equality grows with richness of set of operators
- undecidability for very rich sets
- more on this subject
  - C. Burnikel and R. Fleischer and K. Mehlhorn and S. Schirra: A Strong and Easily Computable Separation Bound for Arithmetic Expressions Involving Radicals, Algorithmica, 2000, volume = 27, pages = 87–99
  - books by Yap: Fundamental Problems in Algorithmic Algebra and Basu, Pollack, Roy: ???
  - Yap's home page

Floating Point Filters

MAX-PLANCK-GESELLSCHAFT

- to compute the sign of an expression E:
- compute $\tilde{E}$ with floating point arithmetic and a bound B such that

$$|E - \tilde{E}| \leq B$$

  - if $|\tilde{E}| > B$ return the sign of $\tilde{E}$
  - otherwise, evaluate E with exact arithmetic

- there is a trade-off between cost (overhead with respect to a pure floating point evaluation) and precision (fraction of expressions for which sign can be determined)
- see LEDAbook, Section 9.7, for a discussion of different approaches
- I will discuss one particular filter: the one used in LEDA and developed by Stefan Näher and myself.

Properties of Floating Point Arithmetic

- $\varepsilon = 2^{-53}$, precision of double precision flaoting point arithmetic
- floating point number: sign bit s, mantisses $m = m_1 \ldots m_{52}$, exponent $e = e_1 \ldots e_{11}$
- e is interpreted as a number in $[0..2047]$
- $e = 0$ and $m_1 = \ldots = m_{52}$, number is $s \cdot 0$.
- represents $s \cdot (1 + \sum_{1 \le i \le 52} m_i 2^{-i}) 2^{e-1023}$, if $1 \le e \le 2046$
- .....
- for any real x, let $\mathrm{fl}(x)$ be the floating point number closest to x
- for an integer a: $\quad |a - \mathrm{fl}(a)| \le \varepsilon |\mathrm{fl}(a)|$
- for $o \in \{+, -, *\}$: $\quad |f_1 \; o \; f_2 - f_1 \; \tilde{o} \; f_2| \le \varepsilon |f_1 \; \tilde{o} \; f_2|$
- floating point arithmetic is monotone

A Floating Point Filter

Let E be an arithmetic expression. The operands are floating point numbers and we have the operators $+$, $-$, and $\times$. Let $\tilde{E}$ be the value of E when evaluated with floating point arithmetic and let E be the value of the expression when evaluated with exact real arithmetic.
We derive a bound of the form

$$|E - \tilde{E}| \leq B \quad \text{or} \quad E \in [\tilde{E} - B, \tilde{E} + B]$$

where

$$B = \varepsilon \cdot ind_E \cdot mes_E,$$

i.e., we enclose the true value of E in an interval with center $\tilde{E}$ and radius B.
In our main theorem (validity of the filter) we prove the equation above and (as an auxiliary claim) $|\tilde{E}| \leq mes_E$.
Our expression for the radius consists of two parts: the part ind_E which is easily precomputed and a part mes_E which is easily computed on-line.
We define ind_E as $((1+\varepsilon)^{d_E} - 1)$ and use the simple recursive definition given in Table below. We write δ instead of $1+\varepsilon$.

The Definitions of mes_E and ind_E

MAX-PLANCK-GESELLSCHAFT

E	$\tilde{E}$	mes_E	d_E
a, integer	$fl(a)$	$\lvert fl(a) \rvert$	1
a, float integer	$fl(a)$	$\lvert fl(a) \rvert$	0
$A+B$	$\tilde{A} \oplus \tilde{B}$	$mes_A \oplus mes_B$	$1+\max(d_A, d_B)$
$A-B$	$\tilde{A} \ominus \tilde{B}$	$mes_A \oplus mes_B$	$1+\max(d_A, d_B)$
$A \cdot B$	$\tilde{A} \odot \tilde{B}$	$mes_A \odot mes_B$	$1+d_A+d_B$

- The recursive definition of mes_E and ind_E.
- column 1 = case distinction according to the syntactic structure of E,
- column 2 = rule for computing $\tilde{E}$ and
- columns 3 and 4 = rules for computing mes_E and ind_E;
- $\oplus$ and $\odot$: floating point implementations of addition and multiplication.
- $fl(a)$ = the floating point number closest to a; a is a float integer if it can be represented as a floating point number

The Theorem

MAX-PLANCK-GESELLSCHAFT

Theorem 1. *If mes_E and d_E are computed according to the Table above then $|\tilde{E}| \leq mes_E$ and $|\tilde{E} - E| \leq ((1+\varepsilon)^{d_E} - 1) \cdot mes_E$*

Intuition and discussion

- mes_E is an upper bound for the E and $\tilde{E}$
- d_E measures the levels of rounding in E.
- each operator introduces an additional level.
- In an addition, the arguments contribute the maximum of their levels,
- in a multiplication, the arguments contribute their sum.
- each level of rounding increases the range of uncertainty by a multiplicative factor of $1+\varepsilon$.
- Subtraction of -1 reflects the fact that we are interested in the error.

The Proof

We use induction on the structure of the expression E.

The claim $|\tilde{E}| \leq mes_E$ follows immediately from the monotonicity of floating point arithmetic. For the second claim we have to work slightly harder.

Assume first that E is an integer a. Then

$$|a - fl(a)| \leq \varepsilon \cdot |fl(a)|$$

and the claim is certainly true. If a is a floating point integer then $fl(a) = a$ and hence the degree can be set to zero for floating point integers.

For the induction step we make a case distinction according to the operation combining A and B.

Induction Step: Addition, $E = A + B$

$$|\tilde{E} - E| = |\tilde{A} \oplus \tilde{B} - (A + B)| \leq |\tilde{A} \oplus \tilde{B} - (\tilde{A} + \tilde{B})| + |\tilde{A} - A| + |\tilde{B} - B|.$$

Item (d) with $f_1 = \tilde{A}$ and $f_2 = \tilde{B}$ implies that the first term is bounded by $\varepsilon|\tilde{A} \oplus \tilde{B}|$ and monotonicity of floating point arithmetic implies that

$$|\tilde{A} \oplus \tilde{B}| \leq mes_A \oplus mes_B = mes_E.$$

For the other two terms we use the induction hypothesis to conclude

$$\begin{aligned} |\tilde{A} - A| + |\tilde{B} - B| &\leq ((1+\varepsilon)^{d_A} - 1) \cdot mes_A + ((1+\varepsilon)^{d_B} - 1) \cdot mes_B \\ &\leq ((1+\varepsilon)^{\max(d_A, d_B)} - 1) \cdot (mes_A + mes_B) \\ &\leq ((1+\varepsilon)^{\max(d_A, d_B)} - 1) \cdot (1+\varepsilon) \cdot mes_E. \end{aligned}$$

Putting the two bounds together we obtain:

$$\begin{aligned} |\tilde{E} - E| &\leq [\varepsilon + ((1+\varepsilon)^{\max(d_A, d_B)} - 1) \cdot (1+\varepsilon)] \cdot mes_E \\ &= [(1+\varepsilon)^{1+\max(d_A, d_B)} - 1] \cdot mes_E. \end{aligned}$$

Induction Step: Multiplication, $E = A \cdot B$

$$|\tilde{E} - E| = |\tilde{A} \odot \tilde{B} - A \cdot B| \leq |\tilde{A} \odot \tilde{B} - \tilde{A} \cdot \tilde{B}| + |\tilde{A} \cdot \tilde{B} - A \cdot \tilde{B}| + |A \cdot \tilde{B} - A \cdot B|.$$

Item (d) with $f_1 = \tilde{A}$ and $f_2 = \tilde{B}$ implies that the first term is bounded by $\varepsilon|\tilde{A} \odot \tilde{B}|$ and monotonicity of floating point arithmetic implies that

$$|\tilde{A} \odot \tilde{B}| \leq mes_A \odot mes_B = mes_E.$$

For the second term we use the induction hypothesis to conclude

$$\begin{aligned}
|\tilde{A} \cdot \tilde{B} - A \cdot \tilde{B}| &= |\tilde{A} - A| \cdot |\tilde{B}| \\
&\leq ((1+\varepsilon)^{d_A} - 1) \cdot mes_A \cdot mes_B \\
&\leq ((1+\varepsilon)^{d_A} - 1) \cdot (1+\varepsilon) \cdot (mes_A \odot mes_B) \\
&= ((1+\varepsilon)^{d_A} - 1) \cdot (1+\varepsilon) \cdot mes_E,
\end{aligned}$$

and ...

and for the third term we conclude similarly

$$\begin{aligned}
|A\cdot\tilde{B}-A\cdot B| &= |A|\cdot|\tilde{B}-B| \\
&\leq (|\tilde{A}|+|\tilde{A}-A|)\cdot|\tilde{B}-B| \\
&\leq (1+\varepsilon)^{d_A}\cdot mes_A\cdot((1+\varepsilon)^{d_B}-1)\cdot mes_B \\
&\leq (1+\varepsilon)^{1+d_A}\cdot((1+\varepsilon)^{d_B}-1)\cdot mes_A\odot mes_B \\
&= (1+\varepsilon)^{1+d_A}\cdot((1+\varepsilon)^{d_B}-1)\cdot mes_E
\end{aligned}$$

Putting the three bounds together we obtain

$$\begin{aligned}
((1+\varepsilon)^{d_E}-1) &\leq \varepsilon+(1+\varepsilon)\cdot((1+\varepsilon)^{d_A}-1)+(1+\varepsilon)^{1+d_A}\cdot((1+\varepsilon)^{d_B}-1) \\
&= \varepsilon+\delta^{d_A+1}-1-\varepsilon+\delta^{1+d_A+d_B}-\delta^{1+d_A} \\
&= \delta^{1+d_A+d_B}-1
\end{aligned}$$

and the induction step is completed.

Corrollary

The bound involves $((1+\varepsilon)^{d_E}-1)$. This number is somewhat hard to compute. We upper bound it by a number that is simpler to compute.

Lemma 1. *If* $d \le \sqrt{1/\varepsilon}-1$ *then* $((1+\varepsilon)^d-1) \le (d+1)\varepsilon$

Proof. We have

$$(1+\varepsilon)^d - 1 = \sum_{1\le i\le d} \binom{d}{i} \varepsilon^i \le \sum_{i\ge 1} (d\cdot\varepsilon)^i = d\varepsilon/(1-d\varepsilon).$$

Next observe that $d\varepsilon/(1-d\varepsilon) \le (1+d)\varepsilon$ iff $d/(1-d\varepsilon) \le (1+d)$ iff $d \le d+1-d^2\varepsilon-d\varepsilon$ iff $d(d+1) \le 1/\varepsilon$. This is certainly the case when $(d+1)^2 \le \varepsilon$ or $d \le \sqrt{1/\varepsilon}-1$. ∎

Theorem 2. *If* $d_E \le \sqrt{1/\varepsilon}-1$ *then* $|E-\tilde{E}| \le (d_E+1)\cdot\varepsilon\cdot mes_E$.

Observe that the condition $d_E \le \sqrt{1/\varepsilon}-1$ is hardly constraining. For $\varepsilon = 2^{-53}$ it amounts to $d_E < 2^{26.5}$.

Application

Let us apply our filter to the orientation predicate for three points in homogeneous coordinates.

$$(ax*bw-bx*aw)*(ay*cw-cy*aw)-(ay*bw-by*aw)*(ax*cw-cx*aw)$$

Let us compute d_E. We have:
The degree of an integer a is $s_1 = 1$;
The degree of an expression of the form $a \cdot a$ is $s_2 = 1+1+1 = 3$.
The degree of an expression of the form $a \cdot a + a \cdot a$ is $s_3 = 1+s_2 = 4$.
The degree of an expression of the form $(a \cdot a + a \cdot a) \cdot (a \cdot a + a \cdot a)$ is $s_4 = 1+s_3+s_3 = 9$.
The degree of the orientation predicate is $s_5 = 1+s_4 = 10$.
We can therefore take 11 as ind_E

In his master's thesis Stefan Funke discussed floating point filters for a larger variety of operations, in particular, divisions and roots.

Static Filters:

In the semi-dynamic filter, ind_E is precomputed and mes_E is computed on-line. Can we also precompute mes_E and in this way obtain a filter where all quantities are precomputed (= static filter)

On page 620 of the LEDAbook, we argue that one can upper bound mes_E in certain situations. If an a-priori bound on the size of all inputs is known, say $mes_a \leq 2^L$ for all inputs a to a computation, then the maximal possible value of mes_E can be precomputed. For example, $mes_E = 2^{4L+3}$ for the orientation predicate.

We concluded that static filters cannot be used for on-line algorithms (=algs which receive their inputs one by one). Also it is a nuisance to precompute L at the beginning of an algorithm. It requires to iterate over all inputs and hence a change of the algorithm.

For this reason, Stefan and I decided against static filters and implemented

semi-dynamic filters.

A Working Static Filter

MAX-PLANCK-GESELLSCHAFT

Stefan Näher, Mathias Bäsken, and Oliver Zlotowski (personal communication, November 10, 2001) observed that our arguments against static filters are not convincing. They suggest to do the following.

We keep a global variable L. Whenever, a new input arrives, we update L. If the value of L changes, we recompute mes_E and B for all geometric predicates used by the algorithm (or for all geometric predicates in the LEDA kernel). The hope is that L and hence B change only rarely. In this way, the filter becomes very fast. We simply compute $\tilde{E}$ and compare it against B.

This has hardly any overhead, since B will be in cache (as it is accessed very frequently). Mathias and Oliver have implemented the new scheme. It will become part of the next release of LEDA. There will be two filter stages in the next release.

1. stage 1: static filter as described in the preceding paragraph
2. stage 2: old semi-dynamic filter
3. stage 3: exact integer computation

Topological Consistency

MAX-PLANCK-GESELLSCHAFT

- pioneered by Sugihara K. Sugihara and M. Iri: A robust Topology-Oriented Incremental algorithm for Voronoi diagrams, Internat. J. Comput. Geom. Appl., volume = 4, year = 1994, pages = 179–228
- idea: make sure that the data structure stays in a consistent state
- in this way avoid that the program crashes
- convex hull algorithm in $\mathbb{R}^2$
- make sure that a new point sees a consecutive set of edges and does not see all edges
  - let e be any visible edge
  - determine all visible edges
  - if all edges of current hull are visible, select one and declare it invisible
- this seems strange

Topological Consistency II

MAX-PLANCK-GESELLSCHAFT

- pioneered by Sugihara
- also used by Held: Voronoi diagrams of line segments. His program VRONI is the currently best program for computing Voronoi diagrams of line segments
- also used by Julian Smith: Boolean Operations on Polyhedra and Polygons.
-
- a good master project: Voronoi diagrams of line segments and Controlled Perturbation