Lower Bounds for Algebraic Computation Trees

As part of the graduate course Algorithms and Complexity

Petros Triantafyllos

09 November 2023

Table of contents

Intro

Background-Motivation

- Algebraic Complexity Theory was a very hot research topic during late 70s with a focus on arithmetic algorithms

Background-Motivation

- Algebraic Complexity Theory was a very hot research topic during late 70s with a focus on arithmetic algorithms
- However previous to this paper no general lower bound method had been provided for algorithms that involve arithmetical operations and comparisons

Background-Motivation

- Algebraic Complexity Theory was a very hot research topic during late 70s with a focus on arithmetic algorithms
- However previous to this paper no general lower bound method had been provided for algorithms that involve arithmetical operations and comparisons
- Nowadays among various general methods for obtaining lower complexity bounds for this model, one of the most efficient uses homotopy invariants, Euler characteristic and Betti numbers, as arguments for the bounding functions.

Background-Motivation

- Algebraic Complexity Theory was a very hot research topic during late 70s with a focus on arithmetic algorithms
- However previous to this paper no general lower bound method had been provided for algorithms that involve arithmetical operations and comparisons
- Nowadays among various general methods for obtaining lower complexity bounds for this model, one of the most efficient uses homotopy invariants, Euler characteristic and Betti numbers, as arguments for the bounding functions.
- The history of this approach started probably in mid 70s with the work of Dobkin and Lipton, and features prominent results such as Ben-Or's and later Yang's.

Main Achievements

- This paper introduces a new topological method for obtaining lower bounds for this general type of algorithms, formally described as algebraic computation trees
- With this method it became possible to generalize, and present in a uniform and easy way, almost all the known nonlinear lower bounds for algebraic computations at that time
- Able to tackle a wide berth of previously untouchable problems such as lower bounds for the complexity of constructions with a ruler and compass in plane Euclidean geometry.

Introduction

Example. Element Distinctness

Given $x_1, \ldots, x_n \in \mathbb{R}$, is there a pair of indexes i, j such that

$$i \neq j \text{ and } x_i = x_j$$

Introduction

Example. Element Distinctness

Given $x_1, \ldots, x_n \in \mathbb{R}$, is there a pair of indexes i, j such that

$$i \neq j \text{ and } x_i = x_j$$

Theorem 1.

Any algebraic computation tree that solves the n-element distinctness problem must have complexity of at least $\Omega\left(nlogn\right)$.

The Algebraic Computation Tree Model

The Membership Problem

Let $W \subset \mathbb{R}^n$. Given $x = (x_1, \ldots, x_n) \in \mathbb{R}^{\ltimes}$ determine if $x \in W$

The Algebraic Computation Tree Model

The Membership Problem

Let $W \subset \mathbb{R}^n$. Given $x = (x_1, \ldots, x_n) \in \mathbb{R}^\ltimes$ determine if $x \in W$

Definition

the Algebraic Computation Tree is a binary tree T with a function that assigns:

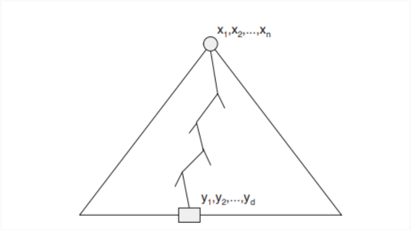
- To any leaf an output Accept or Reject
- To any vertex v with exactly one child an operational instruction of the form $f_v := f_{v_1} \circ f_{v_2}$ or $f_v := c \circ f_{v_1}$ or $f_v := \sqrt{f_{v_1}}$ where v_i is a predecessor of v in T,or $f_{v_i} \in x_1, \ldots, x_n$, $\circ \in \{-, +, \times, \backslash\}$ and $c \in \mathbb{R}$
- To any vertex v with two children a test instruction of the form $f_{v_1} > 0$ or $f_{v_1} \geq 0$ or $f_{v_1} = 0$ where v_1 is a predecessor of v, or $f_{v_i} \in x_1, \ldots, x_n$

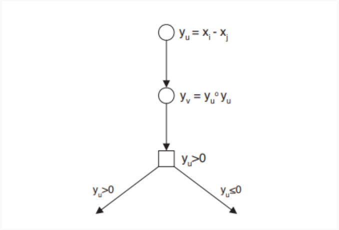
The Algebraic Computation Tree Model

Definitions

- We say that the computation tree T solves the membership problem for W if the answer returned is correct for every input $x \in W$
- Let $cost\,(x, T)$ denote the number of vertices that an input x passes through
- The complexity of $T, C(T) = \underset{x \in W}{max} cost(x, T)$
- Let $C(W) = \underset{T \text{ solves } W}{min} C(T)$

The Algebraic Computation Tree Model





Counting Connected Components

Definitions

Let $V \subset \mathbb{R}^n$ be a set defined by the following polynomial equations

$$q_1(x_1, \ldots, x_n) = 0, \ldots, q_m(x_1, \ldots, x_n) = 0$$

$$p_1(x_1, \ldots, x_n) > 0, \ldots, p_s(x_1, \ldots, x_n) > 0 \tag{1}$$

$$p_{s+1}(x_1, \ldots, x_n) \geq 0, \ldots, p_h(x_1, \ldots, x_n) \geq 0$$

where $\forall i, j : p_i, q_j \in \mathbb{R}[x_1, \ldots, x_n]$ and $d = max\{2, deg(p_i), deg(q_j)\}$

Counting Connected Components

Denote by $\#V$ the number of connected components of V has.
For any $n, h, d \in \mathbb{Z}$ we define

$$\beta_d\,(n,h) = max\{\#V | V \subset \mathbb{R}^n \text{ as defined by } \ (1)\}$$

Counting Connected Components

Denote by $\#V$ the number of connected components of V has.
For any $n, h, d \in \mathbb{Z}$ we define

$$\beta_d\,(n, h) = max\{\#V | V \subset \mathbb{R}^n \text{ as defined by } (1)\}$$

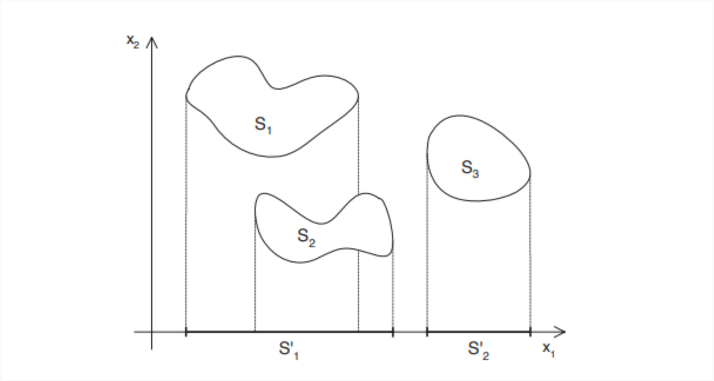
Note that we do not bound the number of equations describing V

Main Results

Counting Connected Components

Theorem

For $d \geq 2$, $\beta_d(n, h) \leq d(2d-1)^{n+h-1}$



Main Theorem

Theorem 3.

Let $W \subset \mathbb{R}^n$ by any set, and let T be a computation tree that solves the membership problem for W. If N is the number of disjoint connected components of W and $h = C(T)$, then

$$2^h 3^{n+h} \geq N$$

Main Theorem

Theorem 3.

Let $W \subset \mathbb{R}^n$ by any set, and let T be a computation tree that solves the membership problem for W. If N is the number of disjoint connected components of W and $h = C(T)$, then

$$2^h 3^{n+h} \geq N$$

Theorem 4.

For any $W \subset \mathbb{R}^n$

$$C(W) \geq \frac{logN}{1 + log3} - \frac{log3}{1 + log3} n \approx 0.38 logN - 0.61n$$

where $N = max\{\#W, \#(\mathbb{R}^n - W)\}$

Main Theorem

Definitions

Let T be a computation tree and let $M(x, T)$ denote the sum of the costs of the operations along the path $P(x)$. The multiplicative complexity of T, $M(T)$, is the maximum of $M(x, T)$ for any $x \in \mathbb{R}^n$, and the (multiplicative) complexity of W, $M(W)$, is the minimum $M(T)$ for any algebraic computation tree

Theorem 5.

For any $W \subset \mathbb{R}^n$,

$$M(W) = \Omega(logN - n)$$

where $N = max\{\#W, \#(\mathbb{R}^n - W)\}$

Main Theorem

Definitions

A d-th order decision tree T for testing if $x \in W \subset \mathbb{R}^n$, is a decision tree where the functions allowed are polynomials of degree at most d, each leaf of T contains the answer Accept or Reject, and for any $x \in \mathbb{R}^n$, T decides correctly if $x \in W$. Denote by $C_d(W)$ the minimum height for any d-th order decision tree for the set W.

Theorem 6.

Let $W \subset \mathbb{R}^n$ be any set, and let T be a d-th order algebraic decision tree that solves the membership problem for W. If N is the number of disjoint connected components of W, and h is the height of T, then

$$2^h \beta_d(n, h) \geq N$$

Thus for fixed d, $C_d(W) = \Omega(logN - n)$

Applications