

Computing Rational Radical Sums in Uniform TC^0

Paul Hunter², Patricia Bouyer-Decitre¹, Nicolas Markey¹,
Joël Ouaknine², James Worrell²

¹ LSV, CNRS & ENS Cachan, France

² OUCL, Oxford, UK

December 13, 2010

Computing Arithmetic Expressions

Problem

How to *efficiently* compute arithmetic expressions?

- Decision problem or function problem?
 - Is the result less than a given value?
 - Does the result equal a given value?
 - Is the result zero?
- What does *efficiently* mean?
 - Obviously, elementary operations (in floating-point arithmetic or over the rationals) can be computed in polynomial time;
 - The problem becomes harder when e.g. radicals come into play;
 - On the theoretical point of view, what is the exact complexity of those problems?

Example: addition of two integers

Example

Addition of two n -bit integers can be computed by circuits:

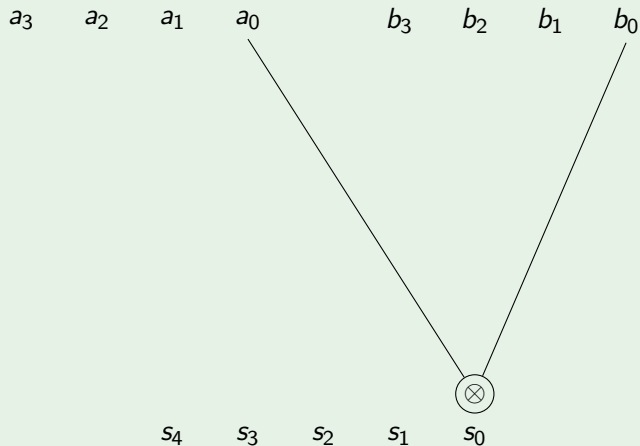
a_3 a_2 a_1 a_0 b_3 b_2 b_1 b_0

s_4 s_3 s_2 s_1 s_0

Example: addition of two integers

Example

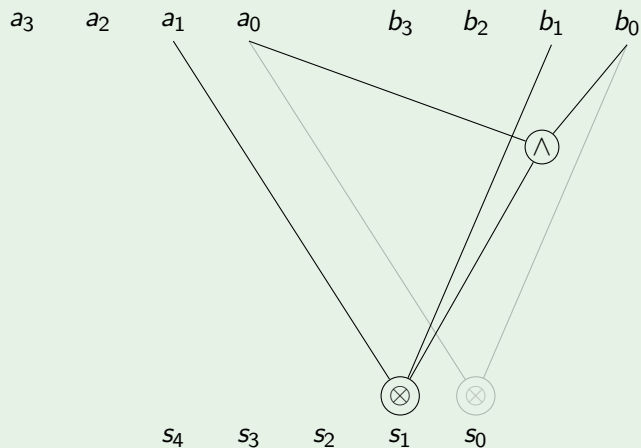
Addition of two n -bit integers can be computed by circuits:



Example: addition of two integers

Example

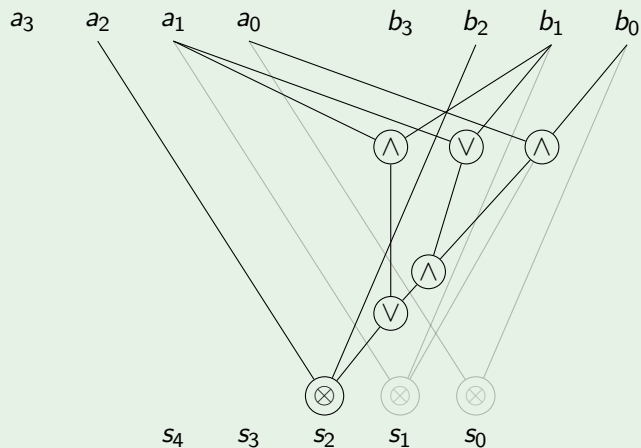
Addition of two n -bit integers can be computed by circuits:



Example: addition of two integers

Example

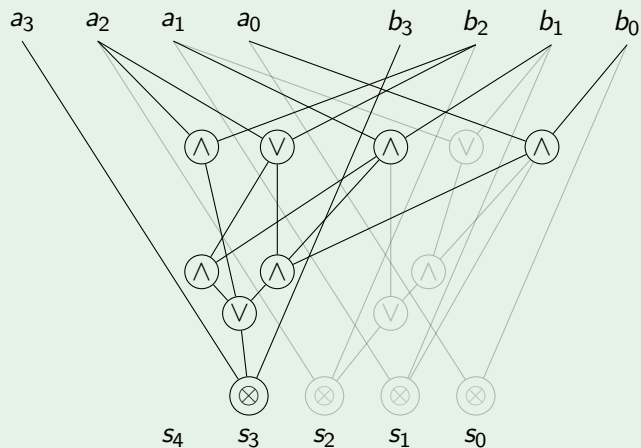
Addition of two n -bit integers can be computed by circuits:



Example: addition of two integers

Example

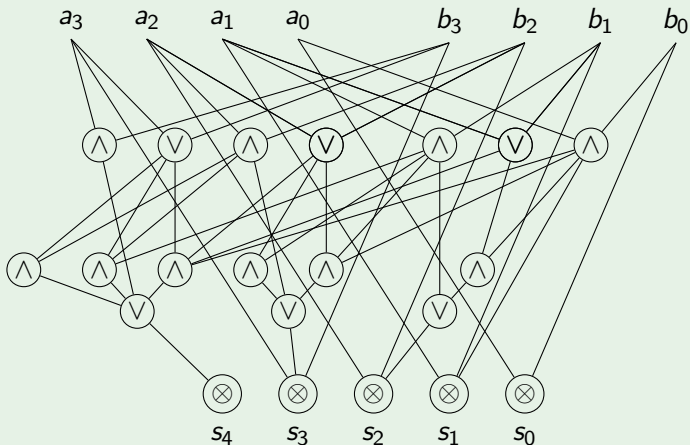
Addition of two n -bit integers can be computed by circuits:



Example: addition of two integers

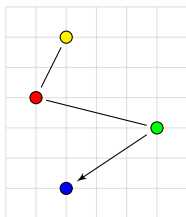
Example

Addition of two n -bit integers can be computed by circuits:



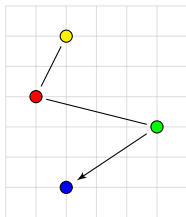
Square-root-sum and related problems

- Geometric problems:
 - Euclidean Traveling Salesman Problem:
compare $\sqrt{5} + \sqrt{18}$ with $\sqrt{10} + \sqrt{13}$.



Square-root-sum and related problems

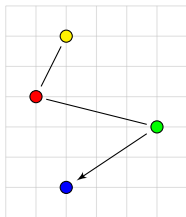
- Geometric problems:
 - Euclidean Traveling Salesman Problem:
compare $\sqrt{5} + \sqrt{18}$ with $\sqrt{10} + \sqrt{13}$.
 - Euclidean Minimum Spanning Tree Problem.



Square-root-sum and related problems

- Geometric problems:

- Euclidean Traveling Salesman Problem:
compare $\sqrt{5} + \sqrt{18}$ with $\sqrt{10} + \sqrt{13}$.
- Euclidean Minimum Spanning Tree Problem.



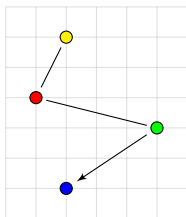
Quoting David Eppstein:

It is not known on Turing machines how to quickly compare a sum of distances (square roots of integers) with an integer or other similar sums, so even (decision versions of) easy problems such as the minimum spanning tree are not known to be in NP.

Square-root-sum and related problems

- Geometric problems:

- Euclidean Traveling Salesman Problem:
compare $\sqrt{5} + \sqrt{18}$ with $\sqrt{10} + \sqrt{13}$.
- Euclidean Minimum Spanning Tree Problem.



- Recently, the “square-root-sum” problem has been reduced to **problems in probabilistic systems and games** [EY07,HMS10]:
 - probability of reachability in Recursive Markov Chains;
 - approximation of Nash equilibria in Shapley’s games.

Square-root-sum problem

Definition (Square-root-sum problem)

Given naturals A_1, \dots, A_n and A , decide whether

$$\sum_{1 \leq i \leq n} \sqrt{A_i} \leq A$$

For instance,

$$\sqrt{518} + \sqrt{855} = 51.9999963 \dots$$

$$\sqrt{457} + \sqrt{763} = 49.0000129 \dots$$

Theorem ([ABKM06])

The square-root-sum problem is in $P^{PP^{PP^{PP}}} \subseteq CH \subseteq PSPACE$.

Radical-sum-eq problem

Definition (Radical-sum-eq problem)

Given rationals $(A_i)_{i \in I}$, $(B_i)_{i \in I}$ and $(C_i)_{i \in I}$ with $0 \leq B_i$ and $0 \leq A_i \leq 1$, decide whether

$$\sum_{i \in I} C_i \cdot B_i^{A_i} = 0.$$

Radical-sum-eq problem

Definition (Radical-sum-eq problem)

Given rationals $(A_i)_{i \in I}$, $(B_i)_{i \in I}$ and $(C_i)_{i \in I}$ with $0 \leq B_i$ and $0 \leq A_i \leq 1$, decide whether

$$\sum_{i \in I} C_i \cdot B_i^{A_i} = 0.$$

Quoting Chee Yap

Whether or not we can decide zero determines whether or not we can compute correctly.

Radical-sum-eq problem

Definition (Radical-sum-eq problem)

Given rationals $(A_i)_{i \in I}$, $(B_i)_{i \in I}$ and $(C_i)_{i \in I}$ with $0 \leq B_i$ and $0 \leq A_i \leq 1$, decide whether

$$\sum_{i \in I} C_i \cdot B_i^{A_i} = 0.$$

Theorem ([Blö91])

Radical-sum-eq is in PTIME.

Radical-sum-eq problem

Definition (Radical-sum-eq problem)

Given rationals $(A_i)_{i \in I}$, $(B_i)_{i \in I}$ and $(C_i)_{i \in I}$ with $0 \leq B_i$ and $0 \leq A_i \leq 1$, decide whether

$$\sum_{i \in I} C_i \cdot B_i^{A_i} = 0.$$

Theorem ([Blö91])

Radical-sum-eq is in PTIME.

Our result

Radical-sum-eq is in uniform-TC⁰.

Outline of the talk

- 1 Introduction
- 2 Circuit Complexity
- 3 RADICALSUMEQ is in uniform TC^0
- 4 Conclusions

Outline of the talk

- 1 Introduction
- 2 Circuit Complexity
- 3 `RADICALSUMEQ` is in uniform TC^0
- 4 Conclusions

Computing with circuits

Circuit complexity

Complexity classes within PTIME:

- build a boolean circuit depending only on the size of the input;
- compute the values of the output gates (in parallel).

Computing with circuits

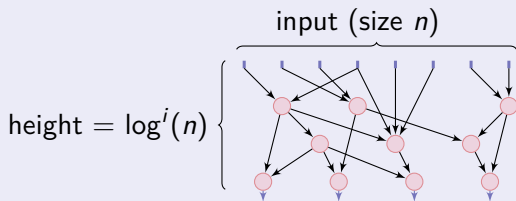
Circuit complexity

Complexity classes within PTIME:

- build a boolean circuit depending only on the size of the input;
- compute the values of the output gates (in parallel).

Several parameters

- “height” of the circuit:



Computing with circuits

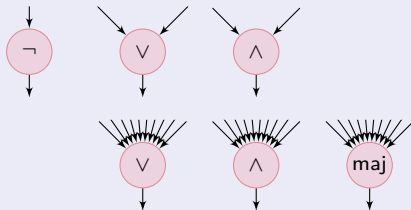
Circuit complexity

Complexity classes within PTIME:

- build a boolean circuit depending only on the size of the input;
- compute the values of the output gates (in parallel).

Several parameters

- “height” of the circuit:
- boolean gates:



Computing with circuits

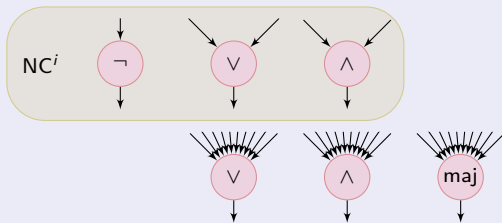
Circuit complexity

Complexity classes within PTIME:

- build a boolean circuit depending only on the size of the input;
- compute the values of the output gates (in parallel).

Several parameters

- “height” of the circuit:
- boolean gates:



Computing with circuits

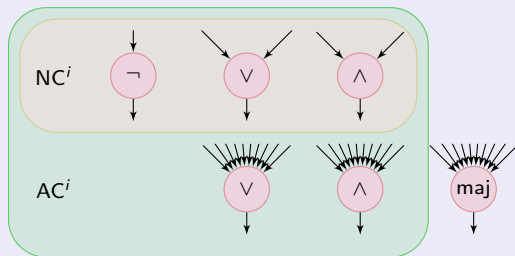
Circuit complexity

Complexity classes within PTIME:

- build a boolean circuit depending only on the size of the input;
- compute the values of the output gates (in parallel).

Several parameters

- “height” of the circuit:
- boolean gates:



Computing with circuits

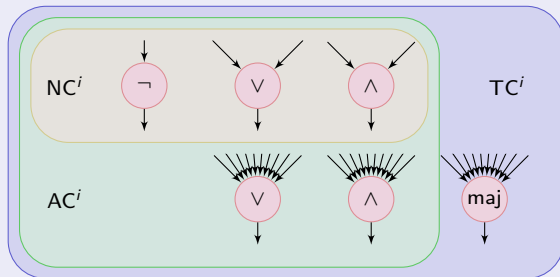
Circuit complexity

Complexity classes within PTIME:

- build a boolean circuit depending only on the size of the input;
- compute the values of the output gates (in parallel).

Several parameters

- “height” of the circuit:
- boolean gates:



Computing with circuits

Circuit complexity

Complexity classes within PTIME:

- build a boolean circuit depending only on the size of the input;
- compute the values of the output gates (in parallel).

Several parameters

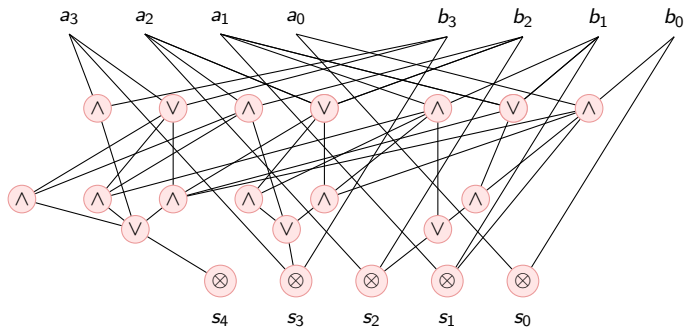
- “height” of the circuit:
- boolean gates:
- “computational power” to build the circuit:

Turing machine for computing circuit \mathcal{C}_k :

- polynomial time can be relevant;
- logarithmic space is often used;
- logarithmic time is especially interesting for smaller classes (AC^0 , TC^0 , ...).

Circuit complexity classes

- Addition is in (DLOGTIME-uniform) AC^0 .



Circuit complexity classes

- Addition is in (DLOGTIME-uniform) AC^0 .
- The following problems are in (DLOGTIME-uniform) TC^0 :
 - Iterated addition (adding n n -bit integers);
 - Multiplication (of two n -bit integers);
 - Iterated multiplication (multiplying n n -bit numbers);
 - Division (integer division of two n -bit numbers).

Circuit complexity classes

- Addition is in (DLOGTIME-uniform) AC^0 .
- The following problems are in (DLOGTIME-uniform) TC^0 :
 - Iterated addition (adding n n -bit integers);
 - Multiplication (of two n -bit integers);
 - Iterated multiplication (multiplying n n -bit numbers);
 - Division (integer division of two n -bit numbers).
- The following problems are not known to be in TC^0 :
 - Greatest common divisor
 - Iterative methods (e.g. Newton's method for computing $\sqrt[n]{A}$)

Circuit complexity classes

- Addition is in (DLOGTIME-uniform) AC^0 .
- The following problems are in (DLOGTIME-uniform) TC^0 :
 - Iterated addition (adding n n -bit integers);
 - Multiplication (of two n -bit integers);
 - Iterated multiplication (multiplying n n -bit numbers);
 - Division (integer division of two n -bit numbers).
- The following problems are not known to be in TC^0 :
 - Greatest common divisor
 - Iterative methods (e.g. Newton's method for computing $\sqrt[n]{A}$)

Theorem

$NC^i \subseteq AC^i \subseteq TC^i \subseteq NC^{i+1} \subseteq PTIME$ for all i .

Theorem

$NC^1 \subseteq LOGSPACE \subseteq NLOGSPACE \subseteq AC^1$.

Outline of the talk

- 1 Introduction
- 2 Circuit Complexity
- 3 RADICALSUMEQ is in uniform TC^0**
- 4 Conclusions

Blömer's algorithm

Lemma ([Blö91])

Let $(A_i)_{i \in I}$ and $(B_i)_{i \in I}$ be two finite sequence of positive rational numbers. The radicals $B_1^{A_1}, \dots, B_n^{A_n}$ are linearly independent over \mathbb{Q} if they are pairwise linearly independant.

Blömer's algorithm

Lemma ([Blö91])

Let $(A_i)_{i \in I}$ and $(B_i)_{i \in I}$ be two finite sequence of positive rational numbers. The radicals $B_1^{A_1}, \dots, B_n^{A_n}$ are linearly independent over \mathbb{Q} if they are pairwise linearly independent.

Algorithm

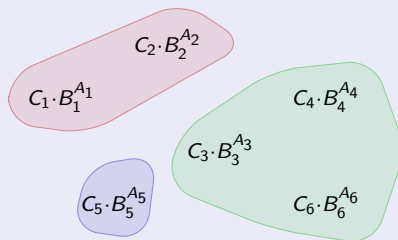
$$\begin{array}{ccc} & C_2 \cdot B_2^{A_2} & \\ C_1 \cdot B_1^{A_1} & & C_4 \cdot B_4^{A_4} \\ & C_3 \cdot B_3^{A_3} & \\ C_5 \cdot B_5^{A_5} & & C_6 \cdot B_6^{A_6} \end{array}$$

Blömer's algorithm

Lemma ([Blö91])

Let $(A_i)_{i \in I}$ and $(B_i)_{i \in I}$ be two finite sequence of positive rational numbers. The radicals $B_1^{A_1}, \dots, B_n^{A_n}$ are linearly independent over \mathbb{Q} if they are pairwise linearly independent.

Algorithm



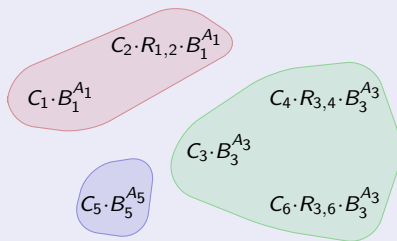
- Partition input terms into linearly dependent groups;

Blömer's algorithm

Lemma ([Blö91])

Let $(A_i)_{i \in I}$ and $(B_i)_{i \in I}$ be two finite sequence of positive rational numbers. The radicals $B_1^{A_1}, \dots, B_n^{A_n}$ are linearly independent over \mathbb{Q} if they are pairwise linearly independant.

Algorithm



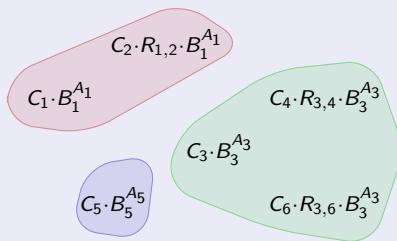
- In each group, rewrite terms with a common radical;

Blömer's algorithm

Lemma ([Blö91])

Let $(A_i)_{i \in I}$ and $(B_i)_{i \in I}$ be two finite sequence of positive rational numbers. The radicals $B_1^{A_1}, \dots, B_n^{A_n}$ are linearly independent over \mathbb{Q} if they are pairwise linearly independant.

Algorithm



- Check for zero in each group.

How to do it in uniform TC^0

Lemma

The following problems are in DLOGTIME-uniform TC^0 :

How to do it in uniform TC^0

Lemma

The following problems are in DLOGTIME-uniform TC^0 :

- *for integers $a < n$ and $B < 2^n$, compute $n^{O(1)}$ bits of $B^{1/a}$;*
 - *approximation by power series [MT99,HAB02].*

How to do it in uniform TC^0

Lemma

The following problems are in DLOGTIME-uniform TC^0 :

- *for integers $a < n$ and $B < 2^n$, compute $n^{O(1)}$ bits of $B^{1/a}$;*
- *for integers $a < n$ and $B < 2^n$, compute $\sqrt[a]{B}$ if in \mathbb{N} :*
 - *compute an integer approximation R of $B^{1/a}$;*
 - *check whether $(R - 1)^a$, R^a or $(R + 1)^a$ equals B .*

How to do it in uniform TC^0

Lemma

The following problems are in DLOGTIME-uniform TC^0 :

- for integers $a < n$ and $B < 2^n$, compute $n^{O(1)}$ bits of $B^{1/a}$;
- for integers $a < n$ and $B < 2^n$, compute $\sqrt[a]{B}$ if in \mathbb{N} ;
- for integer A and rational $B = M/N$, compute $\sqrt[A]{B}$ if in \mathbb{Q} :

Lemma

If $\sqrt[A]{B} \in \mathbb{Q}$, then A is “small” (less than $1 + \log(M \cdot N)$).

- compute $C = \sqrt[A]{M \cdot N^{A-1}}$;
- if it is in \mathbb{N} , return C/N .

How to do it in uniform TC^0

Lemma

The following problems are in DLOGTIME-uniform TC^0 :

- for integers $a < n$ and $B < 2^n$, compute $n^{O(1)}$ bits of $B^{1/a}$;
- for integers $a < n$ and $B < 2^n$, compute $\sqrt[a]{B}$ if in \mathbb{N} ;
- for integer A and rational $B = M/N$, compute $\sqrt[A]{B}$ if in \mathbb{Q} ;
- for $A, A' \in \mathbb{Z}$, $B, B' \in \mathbb{Q}_{>0}$, compute $S = \frac{\sqrt[A]{B}}{\sqrt[A']{B'}}$ if in \mathbb{Q} :

Lemma

If $\frac{\sqrt[A]{B}}{\sqrt[A']{B'}}$ is in \mathbb{Q} , then

- either B and B' are powers of the same rational,
- or A and A' are “small”.

How to do it in uniform TC^0

Lemma

The following problems are in DLOGTIME-uniform TC^0 :

- for integers $a < n$ and $B < 2^n$, compute $n^{O(1)}$ bits of $B^{1/a}$;
- for integers $a < n$ and $B < 2^n$, compute $\sqrt[a]{B}$ if in \mathbb{N} ;
- for integer A and rational $B = M/N$, compute $\sqrt[A]{B}$ if in \mathbb{Q} ;
- for $A, A' \in \mathbb{Z}$, $B, B' \in \mathbb{Q}_{>0}$, compute $S = \frac{\sqrt[A]{B}}{\sqrt[A']{B'}}$ if in \mathbb{Q} :

Lemma

$S \in \mathbb{Q}$ iff, writing $D = \gcd(A, A')$, it holds

- $R = \sqrt[A]{B^D}$ is in \mathbb{Q} ;
- $R' = \sqrt[A']{B'^D}$ is in \mathbb{Q} ;
- $S = \sqrt[D]{R/R'}$ is in \mathbb{Q} ;

Outline of the talk

- 1 Introduction
- 2 Circuit Complexity
- 3 RADICALSUMEQ is in uniform TC^0
- 4 Conclusions

Conclusions

- Radical-Sum-Eq is in DLOGTIME-uniform TC^0 :
 - careful implementation of Blömer's algorithm;
 - very low complexity class, while the problem looks difficult;
- Unfortunately, this does not give much insight on the square-root-sum problem or other geometrical problems.