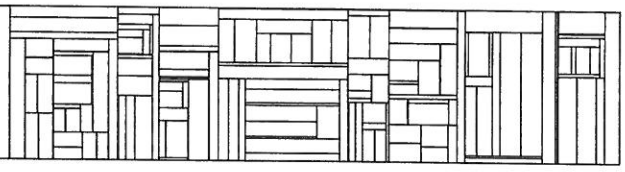
ISSN 0105-8517

The Depth Efficacy of Unbounded Characteristic Finite Field Arithmetic

Gudmund Frandsen
Carl Sturtivant

DAIMI PB – 240
December 1988

AARHUS UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
Ny Munkegade 116 – DK 8000 Aarhus C – DENMARK
Telephone: + 45 6 12 71 88 *Telex: 64767 aausci dk*



Abstract

We introduce an arithmetic model of parallel computation. The basic operations are $\prod$ and $\sum$ gates over finite fields. Functions computed are unary and increasing input size is modelled by shifting the arithmetic base to a larger field.

When only finite fields of bounded characteristic are used, then the above model is fully general for parallel computations; in that size and depth of optimal arithmetic solutions are polynomially related to size and depth of general (boolean) solutions.

In the case of finite fields of unbounded characteristic, we prove that the existence of a fast parallel (boolean) solution to the problem of powering an integer modulo a prime (and powering a polynomial modulo an irreducible polynomial) in combination with the existence of a fast parallel (arithmetic) solution for the problem of computing a single canonical function, $f(x)$, in the prime fields, guarantees the full generality of the finite field model of computation.

We prove that the function $f(x)$ has a fast parallel arithmetic solution for any "shallow" class of primes, i.e. primes p such that any prime power divisor q of $p-1$ is bounded in value by a polynomial in $\log p$.

Introduction

This paper generalises and extends two previous papers [Sturtivant 87, Boyar 88] in which arithmetic models of computation have been investigated.

The first of these papers proposed a model of computation based on fan-in two arithmetic such that every general problem (i.e. a family of functions $\{f_n : \{0,1\}^n \to \{0,1\}^n\}$) has an arithmetic counterpart (ie. a family of functions $\{f_q : \mathbf{F}_q \to \mathbf{F}_q\}$). This counterpart is defined by interpreting a boolean vector of length n as the binary representation of an element in $\mathbf{F}_{2^n}$. One may now compare the complexity of a boolean solution for the original problem with the complexity of an arithmetic solution for the corresponding field problem.

In this first paper, a boolean solution is defined to be a family of circuits $\{c_n : \{0,1\}^n \to \{0,1\}^n\}$ using $\{\vee, \wedge, \neg\}$-gates with fan-in at most two. Similarly, an arithmetic solution is a family of arithmetic circuits $\{c_q : \mathbf{F}_q \to \mathbf{F}_q\}$ using $\{+, -, \cdot, /\}$-gates with fan-in two. The arithmetic or general complexity of a problem is defined as the minimal size (as function of $\log q$ or n) of a correct arithmetic or general solution, respectively.

The arithmetic and general complexity measures were proven to be polynomially related, if and only if a certain family of functions, $\{f_p \mid \mathbf{F}_q$ is a finite prime field$\}$, defined by

$$f_p(x) = \frac{x - x^p}{p} \bmod p$$

has a polynomial size arithmetic solution.

If the finite field model is restricted to fields of bounded characteristic, only a finite number of prime fields are involved and the arithmetic model is then polynomially related to the general model.

All these results are inherently sequential, since fan-in two arithmetic requires depth k circuits to compute the function $x \to x^{2^k}$ over $\mathbf{F}_{2^n}$ for $1 \leq k < n$; whereas the same function in a standard representation of $\mathbf{F}_{2^n}$

may be computed in depth $O(\log\ n)$ with fan-in two boolean circuits.

To overcome this problem, an arithmetic model identical to the one described above, except using unbounded fan-in $\prod$ and $\sum$ gates, was introduced in the second paper, [Boyar 88]. This arithmetic model was compared with a boolean model using threshold gates. For characteristic two fields, the two models have the same depth complexity measure for polynomial size circuits within a constant factor.

In the present paper, we prove that all the sequential results about the fan-in two arithmetic model in [Sturtivant 87] translate naturally into sequential results about a model based on unbounded fan-in arithmetic. Secondly, we discuss the extent to which the parallel model of computation based on characteristic 2 arithmetic [Boyar 88] can be generalised to arbitrary characteristic finite field arithmetic.

Several problems turn up. By having $\prod$-gates, one may obtain high powers very fast (arithmetic depth one), but in the prime fields $\{\mathbf{F}_p\}$ the computation of arbitrary powers fast in parallel (i.e. in depth $(\log\ \log\ p)^{O(1)}$) with boolean operations and hence with threshold gates, is a well known open problem.

Another problem carries over from the sequential case. A boolean solution can exploit the direct access to the bits in the representation; whereas these bits can be accessed fast by arithmetic if and only if the family of functions $\{f_p\}$ mentioned above has polynomial size and polylog depth circuits. We prove that the bits of the standard representation can indeed be accessed fast in parallel by arithmetic for a family of prime fields provided the characteristics form a shallow set of primes, i.e. primes p such that $p-1$ has only small prime power divisors.

Definition d1/1

Arithmetic problems

An arithmetic problem is a family $\{f_q\}$ containing one function $f_q : \mathbf{F}_q \to \mathbf{F}_q$ for every finite field $\mathbf{F}_q$. The input size is defined to be $n = \log q$.

Definition d2/2

Arithmetic circuits

An $\mathbf{F}_q$-circuit is an arithmetic circuit in the field $\mathbf{F}_q$ using unbounded fan-in sum (Σ) and product (Π) gates.

In an $\mathbf{F}_q$-2-circuit, the fan-in to any gate is restricted to be at most 2.

In both cases the size is defined to be the number of gates and the depth is defined in the usual manner.

Remark: The number of edges in a $\mathbf{F}_q$-circuit may be exponentially larger than the size of the circuit, since there may be many parallel wires connecting the output of one gate with the input of another in order to obtain exponentiation (Π) or multiplication by a constant (Σ).

Definition d3/3

Arithmetic Solution

An arithmetic solution to a finite field problem $\{f_q\}$ consists of a family of $\mathbf{F}_q$-circuits ($\mathbf{F}_q$-2-circuits) $\{a_q\}$ such that a_q and f_q are functionally equivalent.

The size and depth of a solution is measured as a function of the input size n.

Remark: We always use unbounded fan-in arithmetic unless explicitly stated otherwise.

Lemma 11/4

Given a family of $\mathbf{F}_q$-circuits of size $S(n)$ and depth $D(n)$, there exists a functionally equivalent family of $\mathbf{F}_q$-circuits of size $O(nS(n)^2)$ and depth $O(D(n)\log S(n))$ satisfying that the fan-in to each Π, Σ gate is at most n, except possibly for some Π-gates (powering-gates), where all incoming wires originate from the same source and their number (the fan-in) is $2^j < q$.

Proof

The difficulty lies in restricting the possibly exponentially high fan-in to the special powering gates.

In the case of a Σ-gate, inputs from the same place are replaced by a single input that via a Π-gate are multiplied by an appropriate constant (namely the number of identical inputs modulo the characteristic of the field). This process reduces the fan-in to about $S(n)$ and a Σ-gate can thus be replaced by a tree of size $O(S(n))$ and depth $O(\log S(n))$ consisting of Σ-gates with fan-in at most n.

Concerning Π-gates, inputs from the same place are bundled together in a like manner: Since k inputs of a value u to a Π-gate introduces a factor of u^k into the output of the Π-gate, we can replace such a bundle of $k = \sum_{i=0}^{n-1} b_i 2^i$ (k can be assumed to be at most $q-1$, the order of the multiplicative group) inputs from the same place by a single input from a small circuit that computes the k th power of u using at most n powering gates and an ordinary Π-gate of fan-in at most n. All Π-gates (except the special powering-gates) have now fan-in $O(S(n))$. As in the case of the Σ-gates those Π-gates may be replaced by trees of size $O(S(n))$ and depth $O(\log S(n))$ using Π-gates of fan-in at most n.

□

Theorem th1/5

Unbounded fan-in versus fan-in 2

If an arithmetic problem $f_q : \mathbf{F}_q \to \mathbf{F}_q$ has optimal solution of size $S(n)$ and $S'(n)$ with unbounded fan-in and fan-in 2 arithmetic respectively. Then $S(n)$ and $S'(n)$ are polynomially related.

Proof

Trivially, $S(n) \leq S'(n)$ since an $\mathbf{F}_q$-2-circuit is also an $\mathbf{F}_q$-circuit. Conversely, an $\mathbf{F}_q$-circuit of size $S(n)$ may be transformed into another $\mathbf{F}_q$-circuit of size $O(n \cdot (S(n))^2)$ using Σ, Π-gates of fan-in at most n except for special powering gates, by l 1/4. Powering gates may be simulated by at most n fan-in 2 gates using 'repeated squaring'. A Π or Σ-gate of fan-in n may be simulated by $n-1$ fan-in 2 gates arranged in a binary tree.

Remark: This shows that all results in [Sturtivant 87] are also valid for unbounded fan-in arithmetic. A corresponding result does not hold for circuit depth since raising to the power $q-2$ requires depth $O(n)$ with fan-in 2 gates, but can be accomplished by a single Π-gate.

Definition d4/6

Threshold gates and Trigger circuits

A threshold gate is a function $Th_k^{\underline{\alpha}} : \{0,1\}^n \to \{0,1\}$, where $\underline{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ is an integer vector of length n and $k \in \mathbf{Z}$ is the threshold. The effect of the gate is defined by

$$Th_k^{\underline{\alpha}}(x_1, x_2, \ldots, x_n) = 1 \text{ iff } \sum_{k=1}^{n} \alpha_i x_i \geq k.$$

A trigger circuit is a circuit over the Booleans using unbounded fan-in threshold gates.

Remark: The usual boolean gates, i.e. *and, or, exclusive-or* and unary negation gates with unbounded fan-in, may all be simulated by constant depth polymonial size trigger circuits. Nevertheless, threshold gates are a physically reasonable primitive gate [Frandsen 88].

Definition d 5/7

Shallow family of circuits

A shallow family of circuits (trigger or arithmetic) satisfy that the size and depth are polynomially bounded in respectively n and $\log n$.

Definition d6/8

Representation

A representation of the finite fields is a tuple $R = (\ell, \{D_q\}, \{\varphi_q\}, \{Z_q\})$, where

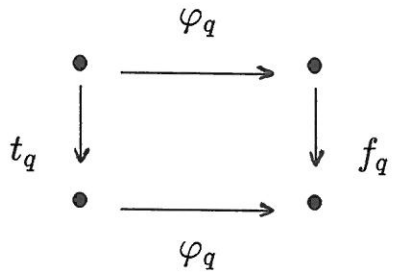
ℓ		is a polynomial
D_q	$\subseteq$	$\{0,1\}^{\lceil \ell(n) \rceil}$ is the set of bit string representations of $\mathbf{F}_q$ elements.
φ_q	:	$D_q \rightarrow \mathbf{F}_q$ is the semantic function.
Z_q	:	$D_q \rightarrow \{0,1\}$ is a trigger circuit satisfying $Z_q(d) = 1$ iff $\varphi_q(d) = 0$. The whole family $\{Z_q\}$ is shallow.

Remark: This definition is deliberately made as broad as possible, allowing multiple bit strings to denote the same field element within a single representation. Hence the zero recognizer $\{Z_q\}$ is necessary in order to interpret the output of a computation. The only other restriction is that reasonably short bit strings must be used.

Definition d7/9

General Solution

A general solution to a finite field problem $\{f_q\}$ with respect to a representation $R = (\ell, \{D_q\}, \{\varphi_q\}, \{Z_q\})$ consists of a family of trigger circuits $\{t_q\}$ such that the following diagram commutes.



Definition d8/10

Efficiency of Arithmetic

Finite field arithmetic is said to be efficient for general parallel computations iff there exists a representation R and a polynomial p such that

(i) Every finite field problem that has an arithmetic solution of size $S(n)$ and depth $D(n)$ has a general solution with respect to R of size no larger than $p\,(n \cdot S(n))$ and depth no larger than $p\,(D(n) \cdot \log(n \cdot S(n)))$.

(ii) Every finite field problem that has a general solution with respect to R of size $S(n)$ and depth $D(n)$ has an arithmetic solution of size no larger than $p\,(n \cdot S(n))$ and depth no larger than $p\,(\log n \cdot D(n))$.

Remark: The definition is in reality a weaker restriction than the natural one, because the result of a finite field computation can not be interpreted unless it is in the standard representation. (Of course a clever new "standard" may be invented, but then the argument would apply

to that.) However, the definition has been made in this way to include anything that might be remotely useful.

Definition d9/11

Good Representation

A good representation $R = (\ell, \{D_q\}, \{\varphi_q\}, \{Z_q\})$ has associated $n+2$ shallow families of trigger circuits

$$\begin{array}{lcl} \{S_q & : & D_q^n \rightarrow D_q\} \\ \{P_q & : & D_q^n \rightarrow D_q\} \\ \{Pow_q^j & : & D_q \rightarrow D_q\}, \quad 1 \leq j < n \end{array}$$

satisfying

$$\begin{array}{lcl} \varphi_q(S_q(s_1, s_2, \ldots, s_n)) & = & \sum_{i=1}^n \varphi_q(s_i) \\ \varphi_q(P_q(s_1, s_2, \ldots, s_n)) & = & \prod_{i=1}^n \varphi_q(s_i) \\ \varphi_q(Pow_q^j(s)) & = & \varphi_q(s)^{2^j} \end{array}$$

Remark: A good representation allows a very efficient parallel implementation of arithmetic.

Definition d10/12

Strong Representation

A strong representation $R = (\ell, \{D_q\}, \{\varphi_q\}, \{Z_q\})$ has associated two shallow families of arithmetic circuits.

$$\begin{array}{lcl} \{i_q & : & D_q \rightarrow \mathbf{F}_q\} \\ \{o_q & : & \mathbf{F}_q \rightarrow D_q \quad (\subseteq \{0,1\}^{l(n)} \subseteq \mathbf{F}_q^{l(n)})\} \end{array}$$

satisfying

$$\begin{aligned} i_q(x_1, x_2, \ldots, x_{l(n)}) &= \varphi_q(\text{“}x_1 x_2 \ldots x_{l(n)}\text{”}) \\ i_q(o_q(x)) &= x \end{aligned}$$

Remark: Note that the Boolean values $0, 1$ is identified with field values $0, 1$ in the above definition. Intuitively, a strong representation is one in which entry to or exit from a representation can be accomplished very efficiently by arithmetic.

Definition d11/13

Equivalent Representations

A representation $R = (\ell, \{D_q\}, \{\varphi_q\}, \{Z_q\})$ translates into a representation $R' = (\ell', \{D'_q\}, \{\varphi'_q\}, \{Z'_q\})$ (written $R \leq R'$) iff there exists a shallow familiy of trigger circuits $\{T_q : D_q \to D'_q\}$, satisfying $\varphi_q = \varphi'_q \circ T_q$.

R and R' are equivalent representations (written $R \equiv R'$) iff $R \leq R'$ and $R' \leq R$.

Remark: In seeking to determine the parallel efficiency of finite field arithmetic as defined in d8/10, we need only distinguish representations up to the equivalence just defined.

Lemma l2/14

There exists a polynomial p such that every general solution of size $S(n)$ and depth $D(n)$ to a finite field problem (with respect to a representation R) have a functionally equivalent family of arithmetic circuits of size $O(p(n \cdot S(n)))$ and depth $O(p(D(n)))$, provided $p(n \cdot S(n)) < q - 1$ ($0, 1$ Boolean values are identified with $0, 1$ field values in the functional equivalence).

Proof

In [Boyar 88], we show that there exists a polynomial p', such that any family of trigger circuits (of size $S(n)$ and depth $D(n)$) can be transformed into a family of circuits using majority and negation gates (of size and depth bounded by $p'(n \cdot S(n))$ and $p'(D(n))$ respectively) satisfying that each majority gate has fan-in at most $p'(n \cdot S(n))$. The term majority gate denotes the special threshold gate $Th_k^{(1,1,\ldots,1)}(x_1, \ldots, x_{2k})$, i.e. a majority gate tests whether at least half of the inputs are one.

The negation gate is easily simulated arithmetically: $\neg(x) = 1 - x$. The simulation of an N-input majority gate is a bit more complicated. Let $x_1, x_2, \ldots, x_N$ be the inputs that will be $\{0, 1\}$ field values. Each input, x_i, enters a small circuit that on input 0, outputs 1; and on input 1, outputs g, where g is a fixed primitive element [Lidl 83] in the field $\mathbf{F}_q$. All of these computations occur in depth 2 and all their results are the input to a single Π-gate. Thus, the function so far computed is given by the expression:

$$\prod_{i=1}^{N} g^{x_i} = g^{\sum_{i=1}^{N} x_i}$$

Since g is a generator for the multiplicative group of $\mathbf{F}_q$, the $N+1$ possible powers of g that can arise from this expression are all distinct, provided $N + 1 \leq p'(n \cdot S(n)) + 1 < q$. As the majority function only depends on the number of ones in the input it may be computed from this result by table lookup. This is achieved in $\mathbf{F}_q$ by the fact that $b(1 - (x - a)^{q-1})$ computes b when x is a and zero otherwise [Lidl 83]. The resulting family of arithmetic circuits satisfies the condition of the lemma for $p(x) = p'(x)^2$.

□

Lemma 13/15

For every strong representation R, there exists a polynomial p such that whenever an arithmetic problem $\{f_q\}$ has a general solution of size $S(n)$ and depth $D(n)$ with respect to R then $\{f_q\}$ has an arithmetic solution of size $O(p(n \cdot S(n)))$ and depth $O(p(\log n \cdot D(n)))$.

Proof

Since every finite field problem has an arithmetic solution of size $O(q) = O(2^n)$ and depth 2 viz $f(x) = \sum_{j=0}^{q-1} a_j x^j$, we may assume that $p(nS(n)) < q - 1$ for p being the polynomial, whose existence is guaranteed by 12/14. In which case, 12/14 provides a family of arithmetic circuits $\{a_q\}$ functionally equivalent to the general solution. Combining this with the shallow families $\{i_q\}$, $\{o_q\}$ that exist due to R being strong, yields an arithmetic solution $\{i_q \circ a_q \circ o_q\}$, satisfying the statement of the lemma.

□

Lemma 14/16

For every good representation R, there exists a polynomial p such that whenever an arithmetic problem $\{f_q\}$ has an arithmetic solution of size $S(n)$ and depth $D(n)$, then $\{f_q\}$ has a general solution with respect to R of size $O(p(n \cdot S(n)))$ and depth $O(p(\log(n \cdot S(n)) \cdot D(n))$.

Proof

First, by the use of 11/4, the arithmetic solution is transformed into a family of $\mathbf{F}_q$-circuits satisfying that the fan-in to each Π, Σ gate is at most n, except possibly for some special powering gates that raise the input value to a power of two. All these gates can be simulated by the

shallow families of trigger circuits $\{P_q\}$, $\{S_q\}$ and $\{Pow_q^i\}$ respectively; all of which are guaranteed to exist by R being good.

□

Lemma 15/17

Given a strong representation R and a good representation R' it is the case that $R \equiv R'$.

Proof

$\underline{R \leq R'}$: If R is strong, then a shallow family of $\mathbf{F}_q$-circuits $\{i_q\}$ exists for R. By 14/16, $\{i_q\}$ may be implemented using representation R'. This gives a shallow family of trigger circuits that takes the R' representation of the R bit string representing a field element, and produces the R' representation of that element. All that remains is to choose two bit strings a_q, b_q satisfying that $\varphi'_q(a_q) = 0$ and $\varphi'_q(b_q) = 1$, and to prefix each input to the new circuit, where an R' representation of zero or one is required, with a small circuit that takes a boolean zero or one and switches into the old input either a_q or b_q as appropriate.

$\underline{R' \leq R}$: If R is strong, then a shallow family of $\mathbf{F}_q$-circuits $\{o_q\}$ exists for R. By 14/16, $\{o_q\}$ may be implemented as a shallow family of trigger circuits using representation R'. The input is now the R' representation of a field element and the output is the R' representation of some bit string that is an R representation of the same field element. Now, we construct a small circuit that will recognize the R' representations of zero and one and output the corresponding boolean value. This may be done using the circuits $\{Z'_q\}$ that exist, due to R' being a representation. Appending one of the constructed circuits to each output where a zero or one in representation R' will appear achieves the desired result.

□

Theorem th2/18

(i) Finite field arithmetic is efficient for general parallel computation *iff* there exists a representation R that is both good and strong.

(ii) If two representations R and R' are both good and strong then $R \equiv R'$.

(iii) If there exists a representation R that is both good and strong, then any representation R' that is either good or strong is both good and strong.

Proof

(i) only if part: We shall prove that the representation R, whose existence is implied by the efficiency of arithmetic is both good and strong.

First, we consider goodness. The arithmetic problems $\sum_{i=1}^{n} x_i$, $\prod_{i=1}^{n} x_i$, and $\prod_{i=1}^{2^j} x$ have all constant size and depth arithmetic solutions. Hence by d8/10 (i), they also have shallow general solutions with respect to R, which implies the goodness of R.

Second, we consider strongness. The arithmetic problems $\{i_q\}$ and $\{o_q\}$ mentioned in d10/12 have shallow general solutions with respect to R. By d8/10 (ii), this implies the existence of shallow arithmetic solutions. Hence, R is strong.

if part: The result follows by l3/15 and l4/16, when taking the polynomial that has to exist according to d8/10 to be the sum of the two polynomials mentioned in l3/15 and l4/16.

(ii) Follows immediately from l5/17.

(iii) We assume R' is strong. By l5/17, $R \equiv R'$. Hence there exist shallow families of trigger circuits $\{T_q : D_q \rightarrow D'_q\}$ and $\{T'_q : D'_q \rightarrow D_q\}$ such that $\{T_q \circ S_q \circ T'_q\}$, $\{T_q \circ P_q \circ T'_q\}$ and $\{T_q \circ Pow_q^j \circ T'_q\}$ certifies the goodness of R', provided S_q, P_q, Pow_q^j certifies the goodness of R.
The case of R' being good is similar, but slightly more involved. It

is here necessary to construct arithmetic analogues of T_q and T'_q by following the ideas used in the proofs of l2/14 and l3/15.

□

Remark: Part (i) of th2/18 shows that the very general question of whether all problems have "same" arithmetic and general complexity reduces to the question of whether there exists a representation such that five specific problems have identical arithmetic and general complexity (the problems $\{P_q\}$, $\{S_q\}$, $\{Pow_q^j\}$ mentioned in d9/11 and the problems $\{i_q\}$, $\{o_q\}$ mentioned in d10/12).

Part (ii) shows that if arithmetic is efficient then there is essentially only one interesting representation.

Part (iii) tells that it is enough to look for a representation that is either good or strong.

In what follows we investigate whether the standard representation is applicable.

Definition d12/19

Standard representation

(i) For prime fields $\mathbf{F}_p$ (p is a prime) the standard representation is $R = (\ell, \{D_p\}, \{\varphi_p\}, \{Z_p\})$, where

$\ell = \lceil \log_2 p \rceil$ (we abbreviate $\ell(n)$ to ℓ)

$D_p = \{b \in \{0,1\}^\ell \mid b$ is the usual binary representation of an integer in the range $[0, p-1]\}$

$\varphi_p(b) = \Sigma_{i=1}^{b} b_i 2^i$

Z_p – obviously a constant depth polynomial size trigger circuit for zero recognition exists.
$Z_p(b_0, b_1, \ldots, b_{\ell-1}) = \wedge_{i=0}^{\ell-1}(\neg b_i)$

(ii) For extension fields $\mathbf{F}_q (q = p^k$, p is a prime, k is a positive integer) a standard representation $R = (\ell, \{D_q\}, \{\varphi_q\}, \{Z_q\})$ is

$\ell = k \cdot \lceil \log_2 p \rceil$ (we abbreviate $\ell(n)$ to ℓ)
$D_q = D_p \times D_p \times \ldots \times D_p$ (k-fold product)
$\varphi_q(d_0, d_1, \ldots, d_{k-1}) = \sum_{i=0}^{k-1} \varphi_p(d_i)\rho_q^i$
where ρ_q is some generating element of the field $\mathbf{F}_q$
Z_q obviously is easy, since $Z_q(d_0, d_1, \ldots, d_k - 1) = \wedge_{i=0}^{k-1} Z_p(d_i)$

Remark: The standard representation of prime fields is unique, whereas the representation of extension fields depends on the choice of generating element ρ_q. For more background see [Lidl 83]. In the following all references to representations allude to a standard representation unless explicitly stated to the contrary.

Lemma 16/20

Given a generating element ρ_q for each $\mathbf{F}_q (q = p^k)$, there exist constant depth polynomial size arithmetic circuits

$$\begin{aligned} \{a_q &: \mathbf{F}_p^k \rightarrow \mathbf{F}_q\} \text{ and} \\ \{b_q &: \mathbf{F}_q \rightarrow \mathbf{F}_p^k\} \end{aligned}$$

such that $x = \sum_{i=0}^{k-1} b_q(x)_i \, \rho_q^i$ and $a_q(b_q(x)) = x$

Proof

Since ρ_q is a generating element of $\mathbf{F}_q$, $\{1, \rho_q, \rho_q^2, \ldots, \rho_q^{k-1}\}$ forms a basis for $\mathbf{F}_q$ regarded as a vector space over $\mathbf{F}_p$. This basis has a dual basis $\{h_0, h_1, \ldots, h_{k-1}\}$ such that if $x = \sum_{i=0}^{k-1} b_i \rho_q^i$ then $b_i = \sum_{j=0}^{k-1} (h_i x)^{p^j}$. For details see [Lidl 83]. Hence $\{b_q\}$ has an arithmetic solution of depth 2 and size $O(k) \leq O(n)$. The same trivially holds for $\{a_q\}$, since $a_q(b_0, b_1, \ldots, b_{k-1}) = \sum_{i=0}^{k-1} b_i \rho_q^i$.

□

Lemma 17/21

Given a standard representation $R = (\ell, \{D_q\}, \{\varphi_q\}, \{Z_q\})$ there exists for each $q = p^k$ a unique degree k polynomial $g_q(x) \in \mathbf{F}_p[x]$ of which $\rho_q = \varphi_q(0,1,0,0,\ldots,0)$ is a root.

If a bit string $d = (d_0, d_1, \ldots, d_{k-1}) \in D_q$ is identified with the polynomial $\chi(d)(x) = \sum_{i=0}^{k-1} \varphi(d_i)x^i$ in $\mathbf{F}_p[x]$ then

$$\chi(d)(x) = [\chi(d')(x) * \chi(d'')(x)] \; mod \; g_q(x)$$

iff

$$\varphi_q(d) = \varphi_q(d') * \varphi_q(d'')$$

for * denoting plus or times.

Proof

see [Lidl 83].

□

Lemma 18/22

The following problems all have Constant Depth Polynomial size Trigger Circuit solutions:

(i) Iterated integer addition

(ii) Iterated integer multiplication

(iii) Computing an integer remainder with respect to an integer modulus

(iv) Iterated polynomial addition

(v) Iterated polynomial multiplication

(vi) Computing a polynomial remainder with respect to a polynomial modulus

Proof

(i) Follows from [Chandra 84].

(ii) Follows from [Beame 86] combined with [Chandra 84].

(iii) Follows from [Chandra 84].

(iv) Follows from (i).

(v) This result is nontrivial, and we merely indicate the line of a proof. The coefficients of the product polynomial are computed modulo p_i for a lot of small primes, $p_1, p_2, \ldots, p_r$ in parallel. By the use of parallel Chinese Remaindering the full coefficients are then reconstructed [Beame 86, Chandra 84].
In order to compute the product polynomial modulo a small prime p, one may use a generalisation of the method used in [Boyar 88] for the special case of $\mathbf{F}_2[x]$.

(vi) Reduces to (iv).

Remark: If in the statement of the lemma only shallow solutions are required, then the proof of (i)-(vi) are all trivial.

Definition d13/23

Modular Powering (MP)

(i) The integer MP-problem is a family of functions

$\{f_{(p,r)} : D_p \to D_p,$ where p is a prime, $0 \leq r \leq p-1$ and D_p is the binary integers in the standard representation of the prime fields.$\}$

defined by

$$f_{(p,r)}(x) = (x^r \bmod p)$$

naturally, the input size $n = \log p$.

(ii) The polynomial MP-problem is a family of functions

$\{f_{(q,r,\rho_q)} : D_q \to D_q,$ where $q = p^k$ for a prime p, $0 \leq r \leq p-1$, D_q is the bit strings of a standard representation determined by a generating element ρ_q of $\mathbf{F}_q$.$\}$

defined by

$$\chi(f_{(q,r,\rho_q)}(d))(x) = (\chi(d)^r \bmod g_q(x))$$

where $\chi, g_q(x)$ is chosen according to 17/21.

Remark: The integer MP-problem reduces to the polynomial one since powering of a constant polynomial in $\mathbf{F}_p[x]$ is equivalent to powering in $\mathbf{F}_p$.

Definition d14/24

Bit Extraction (BE)

The BE problem is the finite field problem (for prime fields only)

$$\{b_p : \mathbf{F}_p \to \{0,1\}^n \in \mathbf{F}_p^n\},$$

defined by $\varphi_p(b_p(x)) = x$, where φ_p is the semantic function of the standard representation and field values $0, 1$ are identified with Boolean values $0, 1$.

Theorem th3/25

Standard and strong/good

(i) A standard representation is strong iff the BE-problem has a shallow arithmetic solution.

(ii) A standard representation is good iff the MP-problems have shallow general solutions.

Proof

(i) if-part: A shallow arithmetic solution for the $\{o_q : \mathbf{F}_q \to \{0,1\}^{\ell(n)}\}$ problem may be constructed by combining the circuits $\{b_q\}$ (by 16/20) with a shallow arithmetic solution for the BE-problem.

In case of the $\{i_q : \{0,1\}^{\ell(n)} \to \mathbf{F}_q\}$ problem, a shallow arithmetic solution comes from the circuits $\{a_q\}$ (16/20) combined with a constant depth circuit for $(b_0, b_1, \ldots, b_{n-1}) \to \sum_{i=0}^{n-1} b_i 2^i$.

only if-part: We assume a shallow solution for the $\{o_q\}$ problem exists. This solution is specifically valid in the prime fields and is thus also a solution for the BE-problem.

(ii) if-part: By 17/21 and 18/22 existence of shallow families of trigger circuits for $\{P_q\}$ and $\{S_q\}$ follows. In the case of $\{Pow_q^j\}(q = p^k)$, 2^j is written in base p : $2^j = \sum_{i=0}^{k-1} d_i p^i$. Consequently, $x^{2^j} = \Pi_{i=0}^{k-1}(x^{d_i})^{p^i}$.

A solution to the MP-problem allows us to raise to the power d_i and the Π-gate is implemented using 18/22. The only remaining problem is conjugation, $x \to x^{p^i}$, which is an automorphism on the field $\mathbf{F}_q$. Hence if $x = \sum_{j=0}^{k-1} x_j \rho^j$ then $x^{p^i} = \sum_{j=0}^{k-1} x_j \rho^{j \cdot p^i}$ and since there must exist numbers $M_{ij\ell} \in \mathbf{F}_p$ such that $\rho^{jp^i} = \sum_{\ell=0}^{k-1} M_{ij\ell}\rho^\ell$, we obtain $x^{p^i} = \sum_{j=0}^{k-1} x_j \sum_{\ell=0}^{k-1} M_{ij\ell}\rho^\ell = \sum_{\ell=0}^{k-1}(\sum_{j=0}^{k-1} x^j M_{ij\ell})\rho^\ell$. Consequently, conjugation may be implemented in constant depth by 17/21 and 18/22.

only if-part: Assume we have shallow families of trigger circuits for powering $\{Pow_q^j\}$ and product $\{P_q\}$. By writing $r \in \{0, 1, \ldots, p-1\}$ in binary form, $r = \sum_{i=0}^{\log p} b_i 2^i$, it is easy to construct a solution for the polynomial (and thus the integer) MP-problem $\{f_{(q,r,\rho_q)}\}$ from $\{Pow_q^i\}$ and $\{P_q\}$ using 17/21.

Remark: For bounded characteristic both BE and MP have shallow solutions and hence by Th3/25 and Th2/18(i), finite field arithmetic is efficient for parallel computations in this case.

Th2 and 18(iii) in connection with Th3/25 imply that if one of the problems BE or MP are known to have a shallow solution, then the other of the two problems have a shallow solution iff finite field arithmetic is efficient for parallel computations.

Below we shall construct a shallow solution for the BE problem in a special case.

Definition d15/26

Shallow Primes

Given a polynomial ℓ, a set of primes S is shallow (with respect to ℓ) if any prime power r that divides $p-1$ for $p \in S$, satisfies that $r \leq \ell(n)$, $n = \log p$.

Remark: Given a prime p, one may decide in time polynomial in n whether p is shallow with respect to ℓ by trial division.

Lemma 19/27

The BE-problem has a shallow arithmetic solution when restricted to an arbitrary set of shallow primes.

Proof

Assume $p-1 = q_1 \cdot q_2 \cdot \ldots \cdot q_k$ for relatively prime prime powers q_j satisfying that $q_j \leq \ell(n)$ for a polynomial ℓ.

$\mathbf{F}_p^*$, the multiplicative group of $\mathbf{F}_p$ has unique subgroups $U_1, U_2, \ldots, U_k$ of order $q_1, q_2, \ldots, q_k$ respectively and is isomorphic to $U_1 \times U_2 \times \ldots \times U_k$ by the Chinese Remainder Theorem.

Actually, any $x \in \mathbf{F}_q^*$ may be written uniquely as $x = x_1 \cdot x_2 \cdot \ldots \cdot x_k$,

where $x_i \in U_i$. Each x_i may be computed by a single Π gate:

$$x_i = x^{\frac{p-1}{q_i} \cdot c_i} \text{ (where } c_i = (\frac{p-1}{q_i})^{-1} \; mod \; q_i \text{ is a constant.)}$$

Since an x_i takes at most $q_i \leq \ell(n)$ different values, the bits of the standard representation of x_i can be found by a constant depth size $O(\ell(n))$ table lookup circuit. The multiplication $x = \Pi_{i=1}^{k} x_i$ may now be simulated on the bits of x_i to give the bits of x. This can be done by shallow arithmetic circuits using l8/22 and l2/14.

□

Theorem th4/28

Efficiency and shallow primes

When restricting attention to a class of finite fields for which the characteristics form a shallow set of primes the following is true:

The restricted finite field arithmetic is efficient for parallel computations iff the MP-problem has a shallow general solution for these finite fields.

Proof

Follows by l9/27, th3/25 and th2/18.

□

Remark: By th2/18(ii) for a shallow set of primes the standard representation is the only candidate (up to equivalence) for a representation satisfying d8/10.

When considering the efficiency of finite field arithmetic for sequential computations only, a stronger statement can be made. The MP-problems obviously have a polynomial size solution (using repeated squaring) and the definition of shallow set of primes (with respect to ℓ) may be relaxed to include all classes of primes p for which every prime divisor r in $p-1$ satisfies that $r \leq \ell(n)$.

In fact the following statement is valid: finite field arithmetic is efficient for sequential computations, when restricting attention to a class of fields for which the characteristics form a shallow set of primes, with respect to the relaxed (sequential) definition of a shallow set of primes.

Definition d16/29

f, g, π, σ, m

The finite field problem $\{f_p\colon \mathbf{F}_p \rightarrow \mathbf{F}_p\}$ is defined by:

$f_p(x) = (\frac{x-x^p}{p} \ mod\ p)$, where a field element x is identified with its standard (integer) representation, and the expression is evaluated over the integers, before taking the remainder to get a field element

Similarly $\{g_p : \mathbf{F}_p^2 \rightarrow \mathbf{F}_p\}$ is defined by:

$$g_p(x,y) = (\frac{x^p + y^p - (x+y)^p}{p} \ mod\ p)$$

The functions $\{\pi_p, \sigma_p : \mathbf{F}_p^2 \rightarrow \mathbf{F}_p\}$ are the multiplicative and additive carry respectively from the first position into the second position in base p arithmetic, i.e.:

$$\pi_p(x,y)p + [(x \cdot y)\ mod\ p] = x \cdot y$$
$$\sigma_p(x,y)p + [(x+y)\ mod\ p] = x + y$$

The function $m_p : \mathbf{Z}_{p^2} \rightarrow \mathbf{Z}_{p^2}$ is the "*mod p*" function defined by:

$$m_p(xp + y) = y \text{ for } x, y \in \mathbf{Z}_p (\cong \mathbf{F}_p)$$

Remark: A more thorough introduction to the functions f and g may be found in [Sturtivant 87], where their importance for the BE-problem with respect to sequential efficiency of finite field arithmetic is discussed. Here we shall find that a similar result holds in the parallel case. Below we present some selected identities from [Sturtivant 87].

Lemma l10/30

π, σ, g, f satisfy the following identities

(i) $\pi_p(x, y) = x \cdot f_p(y) + f_p(x) \cdot y - f_p(x \cdot y)$

(ii) $\sigma_p(x, y) = f_p(x) + f_p(y) + g_p(x, y) - f_p(x + y)$

(iii) $g_p(x, y) = \begin{cases} 0 \text{ for } x = 0, y = 0, x + y = 0 \\ x[f_p(1 + \frac{y}{x}) - f_p(\frac{y}{x})], \text{ otherwise} \end{cases}$

Proof

See [Sturtivant 87].

□

Lemma l11/31

Given a modulus r, $2 \leq r \leq p - 1$. If $p = s \cdot r + t$, $0 \leq t \leq r - 1$ then $x \; mod \; r = x - r[\pi_p(s, x) + \sigma_p(x - \pi_p(s, x)r, p - r)]$ for $x \in \mathbf{Z}_p$, $(0 \leq x \leq p - 1)$.

Proof

Assume $x = ar + b$ for $0 \leq b \leq r - 1$. Since $x \; mod \; r = b = x - ra$, it suffices to find an expression for a.

We may compute $s \cdot x = s(ar + b) = ap + (sb - at)$, which combined with the observation that $\mid sb - at \mid \leq p - 1$ leads to two different cases.

(i) If $0 \leq sb - at \leq p - 1$ then $a = \pi_p(s, x)$. This implies that $\sigma_p(x - \pi_p(s, x)r, p - r) = \sigma_p(b, p - r) = 0$.

(ii) If $-(p - 1) \leq sb - at < 0$ then $a = \pi_p(s, x) + 1$. This implies that $\sigma_p(x - \pi_p(s, x)r, p - r) = \sigma_p(b + r, p - r) = 1$

Thus in both cases, we find $a = \pi_p(s, x) + \sigma_p(x - \pi_p(s, x)r, p - r)$.

□

Lemma l12/32

If the binary expansion of $p - 1$ is $p - 1 = \sum_{j=0}^{n} b_j 2^j$ then the following formula is valid

$$f_p(x) = -x \sum_{k=0}^{n} [\frac{\pi_p(x^{\sum_{j=0}^{k-1} b_j 2^j}, x^{b_k 2^k})}{x^{\sum_{j=0}^{k} b_j 2^j}} + b_k 2^k \sum_{\ell=0}^{k-1} \frac{1}{2^{\ell+1}} \frac{\pi_p(x^{2^\ell}, x^{2^\ell})}{x^{2^{\ell+1}}}]$$

for $\mathbf{F}_p$-arithmetic.

Proof

The following formula comes from a simple transcription of l10/30(i)

$$\frac{f_p(xy)}{xy} = \frac{f_p(y)}{y} + \frac{f_p(x)}{x} - \frac{\pi_p(x, y)}{xy}$$

and it gives by repeated application

$$\frac{f_p(x^{2^k})}{x^{2^k}} = 2^k \cdot \frac{f_p(x)}{x} - \sum_{\ell=0}^{k-1} \frac{2^k}{2^{\ell+1}} \frac{\pi_p(x^{2^\ell}, x^{2^\ell})}{x^{2^{\ell+1}}}$$

Similarly, we find

$$f_p(x^{p-1}) = f_p(x^{\sum_{j=0}^{n} b_j 2^j}) = \sum_{k=0}^{n} \frac{f_p(x^{b_k 2^k})}{x^{b_k 2^k}} - \sum_{k=0}^{n} \frac{\pi_p(x^{\sum_{j=0}^{k-1} b_j 2^j}, x^{b_k 2^k})}{x^{\sum_{j=0}^{k} b_j 2^j}}$$

Now, $x^{p-1} \in \{0,1\}$ and $f_p(0) = f_p(1) = 0$ by 110/30. Hence, $f_p(x^{b_k 2^k}) = 0$ for $b_k = 0$.

By using these fact and combining the above equations, we obtain:

$$0 = \sum_{k=0}^{n} b_k 2^k \cdot \frac{f_p(x)}{x} - \sum_{k=0}^{n} \sum_{\ell=0}^{k-1} b_k \frac{2^k}{2^{\ell+1}} \frac{\pi_p(x^{2^\ell}, x^{2^\ell})}{x^{2^{\ell+1}}} - \sum_{k=0}^{n} \frac{\pi_p(x^{\sum_{j=0}^{k-1} b_j 2^j}, x^{b_k 2^k})}{x^{\sum_{j=0}^{k} b_j 2^j}}$$

from which the lemma follows by rearrangement of terms.

□

Theorem th5/33

f versus the BE-problem

The BE-problem has a shallow arithmetic solution, iff $\{f_p\}$ has a shallow arithmetic solution.

Proof

only if-part: We assume the BE-problem has a shallow arithmetic solution. In this case, $\{\pi_p\}$ may be computed fast in

parallel by arithmetic, using th3/25(i), l3/15 and l8/22. A shallow arithmetic solution for $\{f_p\}$ may now be constructed using l12/32.

if-part: We assume $\{f_p\}$ has a shallow arithmetic solution. By l10/30, so have $\{\pi_p\}$ and $\{\sigma_p\}$. Hence by l11/31, we may compute $x_k = (x \ mod \ 2^k)$ for $1 < 2^k < p$ by shallow arithmetic circuits. If $x = \Sigma_{i=0}^{n} b_i 2^i$ $(2^n < p \leq 2^{n+1})$ we find

$$\begin{aligned} b_k &= (x_{k+1} - x_k)/2^k \text{ for } 0 < k < n \\ b_0 &= x_1 \\ b_n &= \sigma_p(x, p - 2^n) \end{aligned}$$

Thus the BE-problem has a shallow arithmetic solution.

Remark: When resolving the status of the standard representation with respect to being strong, Th3/25 and th5/33 combined tell us that we need only consider the $\{f_p\}$-function.

Theorem th6/34

The $\{f_p\}$-problem has a shallow arithmetic solution iff the $\{g_p\}$-problem and the $\{m_p\}$-problem both have shallow arithmetic solutions with $\mathbf{Z}_p$ and $\mathbf{Z}_{p^2}$-arithmetic respectively.

Proof

We omit details of this proof. The proof given in [Sturtivant 87] for a sequential version of the above theorem carries over to the parallel case, when using the fact that Π, Σ-gates over $\mathbf{Z}_{p^2}$ can be simulated by shallow circuits employing Π, Σ, g-gates over $\mathbf{Z}_p$ in the Witt-representation of $\mathbf{Z}_{p^2}$. (For definition and background of the Witt representation, see [Sturtivant 87]). The simulation uses the two formulas:

$$\begin{aligned}
\Pi_{i=1}^{k}(x_0^{(i)}, x_1^{(i)}) &= (x_0, x_0 \cdot \Sigma_{i=1}^{k} \tfrac{x_1^{(i)}}{x_0^{(i)}}) \ , \ \text{where} \ x_0 = \Pi_{i=1}^{k} x_0^{(i)} \\
\Sigma_{i=1}^{k}(x_0^{(i)}, x_1^{(i)}) &= (x_0, x_1 + \Sigma_{i=1}^{k} g(\Sigma_{j=1}^{i-1} x_0^{(j)}, x_0^{(i)})), \\
&\quad \text{where} \ x_\ell = \Sigma_{i=1}^{k} x_\ell^{(i)} \ \text{for} \ \ell = 0, 1
\end{aligned}$$

□

Conclusion

In the case of sequential computations, the following can be added to the points argued in [Sturtivant 87].

- fan-in two gates and gates of unbounded fan-in give rise to optimal arithmetic circuits of polynomially related sizes.
- finite field arithmetic is fully general for sequential computations, when characteristics are constrained to a sequentially shallow set of primes, i.e. primes p for which all prime divisors of $p-1$ are small.

Those findings suggest:

- When reasoning about the complexity of computations, we should use Π and Σ gates to avoid the combinatorics of fan-in two gates, even in the case of sequential computations.

In the case of parallel computations, we have shown

- When restricted to bounded characteristic, finite field arithmetic is fully general for parallel computations.
- The efficiency of unbounded characteristic finite field arithmetic is implied by the existence of a polylog depth, poly size solution for the modular powering problem in combination with polylog depth, poly size arithmetic circuits to extract the bits of a prime field element in the standard representation.

- This problem of bit extraction has a polylog depth, poly size arithmetic solution if and only if the prime field function $f(x) \stackrel{p}{\equiv} (\frac{x-x^p}{p})$ has a polylog depth polysize arithmetic solution.

- The function $f(x)$ has a polylog depth, poly size arithmetic solution iff the function $g(x,y) \stackrel{p}{\equiv} \frac{x^p+y^p-(x+y)^p}{p}$ and $m(x) \stackrel{p^2}{\equiv} (x \ mod \ p)$ both have polylog depth poly size arithmetic circuits over $\mathbf{Z}_p$ and $\mathbf{Z}_{p^2}$ respectively.

- The problem of bit extraction has a constant depth, poly size solution, when the characteristic is restricted to a parallel shallow class of primes, i.e. primes p for which all prime power divisors of $p-1$ are small.

- If unbounded characteristic finite field arithmetic is efficient for parallel computations, despite the bit extraction and the modulo powering problem not having shallow solutions, then there exists an efficient nonstandard representation, whose relation to the standard representation is inherently sequential.

- If finite field arithmetic is efficient for parallel computations, then there exists only one efficient representation of the fields (up to fast parallel conversion).

References

[BEAME 86] BEAME, P.W., COOK, S.A. and HOOVER, H.J., Log Depth Circuits for Division and Related Problems. *SIAM J. Computing 15 (1986), pp. 994-1003.*

[BOYAR 88] BOYAR, J., FRANDSEN, G.S. and STURTIVANT, C. An Algebraic Model for Bounding Threshold Circuit Depth. *Technical Report DAIMI PB-239, Computer Science Department, Aarhus University, Denmark, 1988.*

[CHANDRA 84] CHANDRA, A.K., STOCKMEYER, L. and VISHKIN, U., Constant Depth Reducibility. *SIAM Journal on Computing 13 (1984), pp. 423-439.*

[FRANDSEN 88] FRANDSEN, G.S. and STURTIVANT, C., An exact and efficient implementation of Threshold Gates with arbitrary Real Weights. *Technical Report DAIMI PB-241, Computer Science Department, Aarhus University, Denmark, 1988.*

[LIDL 83] LIDL, R. and NIEDERREITER, H., *Finite Fields.* Encyclopedia of Mathematics and its Applications, 20. Addison-Wesley, Reading, Mass., 1983.

[STURTIVANT 87] STURTIVANT, C. and FRANDSEN, G.S., The Computational Efficacy of Finite Field Arithmetic. *Technical Report DAIMI PB-227, Computer Science Department, Aarhus University, Denmark, 1987.*