Bounds On Parallel Computation of Multivariate Polynomials

Ilan Sade* Amir Averbuch

Deptartment of Computer Science
School of Mathematical Sciences, Tel Aviv University
Tel Aviv 69978, Israel
E-mail: sade@math.tau.ac.il

Abstract. We consider the problem of fast parallel evaluation of multivariate polynomials over a field **F**. We define "maximal-degree" ($max_{\deg}$) of a multivariate polynomial f as $\max_i deg_{x_i}(f(x_1, \ldots, x_n))$ $i = 1, \ldots, n$.
The first lower bound result states that if a circut **G** evaluates a multivariate polynomial f, where its nodes are capable of performing $(+, *)$, then the depth(**G**) is not less than $\log_2\lceil max_{\deg}(f)\rceil$. This result is a generalization of Kung's[K] results for a univariate polynomial which is $\log_2\lceil deg\, f\rceil$.
In the second part, we consider the circuit **G** which evaluates an arbitrary polynomial f in n variables with $max_{\deg}(f) \triangleq d_p > 1$.
We present two algorithms that achieve better performance than the classical results of Hyafil[H] and Valiant et al. [VSBR] for most classes of multivariate polynomials. For the class of " dense " polynomials the results are closed to the theorethical bound $\log C$, where C is the sequential complexity.
The algorithms generalize Munro-Paterson[MP] method for the univariate case. It should be noticed that the bound obtained by Hyafil[H] and Valiant, Skyum, Berkowitz and Rackoff[VSBR] is not sufficiently tight for the worst-sequential case (dense multivariate polynomials) and their bound can be reduced by the factor of $\log d$ while the number of required processors is only $O(C)$. The best improvement is achieved in a case of a "dense" multivariate polynomial. A polynomial is dense if the computation necessitates $\Omega(d_p^n)$ sequential steps.
The simple algorithm requires only $n \log d_p + O(n\sqrt{\log d_p})$ parallel steps .The second algorithm has parallel complexity , measured by the depth of the circuit , $depth(\mathbf{G}) \leq n(\log d_p + \beta) + \log n + \sqrt{\log d_p}$ where $\beta \leq \sqrt{\log d_p}$. If $C = \Omega(d_p^n)$ then it is less than $\log C + \sqrt{n \cdot \log C}$ where C is the number of sequential steps, and it requires only $O(C)$ processors.
The second algorithm is slightly better than the " simple" one. The improvement is achieved when β is small. The improvement of both algorithms in the parallel complexity and the number of processors with respect to Valiant is significant for most classes of multivariate polynomials.

Key words :Complexity of parallel computation, Multivariate polynomials, Design and analysis of parallel algorithms , Maximal-degree, Dense polynomial.

* This author was supported by Eshkol Fellowship No. 0375 administered by the Israeli Ministry of Science.

1 Introduction

Hyafil[H] proved that any multivariate polynomial $f(x_1, \ldots, x_n)$ of degree d that can be computed with C sequential steps, can be computed in parallel in $O(\log d)(\log C + \log d)$ steps. His method requires, in general, $C^{\log d}$ processors. Therefore, even if C and d are both polynomially bounded in terms of n (the number of indeterminates) the number of processors required is not polynomially bounded.

Valiant et. al.[VSBR] improved Hyafil's result by reducing the number of required processors to $O(Cd)^{\beta}$ processors for an appropriate β.

In the first part of the paper we prove a lower bound for the parallel time and at the second part of the paper we consider the worst sequential case, which we define as "dense" polynomial. For this case, we prove that the parallel time is only $O(\log C)$ and the number of required processors is only $O(C)$ which are both better than the previous mentioned general results. It stressed the necessity of parallelization for the computation of "dense" multivariate polynomials.

Definition 1. Let $\mathbf{F}$ be a field and let $\mathbf{F}[x_1, \ldots, x_n]$ be the ring of polynomials over indeterminates $x_1, \ldots, x_n$ with coefficients from $\mathbf{F}$.

A program f over $\mathbf{F}$ is a sequence of instructions:

$$v_i \longleftarrow v_i' \circ v_i'' \quad i = 1, \ldots, C$$

where for each value i :

I. $v_i', v_i'' \in \mathbf{F} \cup \{x_1, \ldots, x_n\} \cup \{v_1, \ldots, v_{i-1}\}$
II. $\circ$ is one of the ring operations $\{+, *\}$.

Subtraction can be simulated by using $(-1) \in \mathbf{F}$. The formal polynomial evaluation at v_i can be defined as $f(v_i)$. The degree of $f(v_i)$ is denoted by $d(v_i)$.

The computational model is a circuit with nodes capable of performing "add" and "multiply" operations.

We assign to each node v_i a vector of n elements $\mathbf{deg}(v_i) = \{deg_1(v_i, \ldots, deg_n(v_i)\}$ where for $l = 1, \ldots, n$, we define $deg_l(v_i) = deg_{x_l} f(v_i)$ and

$$d(v_i) \triangleq \sum_{l=1}^{n} deg_l(v_i) \qquad max_{\deg}(v_i) \triangleq \max_l deg_l(v_i)$$

The degree of a scalar in the field is 0, the degree of an indetereminate x_l is the unit vector $\mathbf{e_l}$ $\; l = 1, \ldots, n$. In case of a multiplication node v_i then $deg_l(v_i) = deg_l(v_i') + deg_l(v_i'')$ $\; l = 1, \ldots, n$, and in case of an addition node then $deg_l(v_i) = \max(deg_l(v_i'), deg_l(v_i''))$ $\; l = 1, \ldots, n$.

2 Lower Bound

Kung[K] derived a lower bound for the parallel processing time of rational expressions in the case of a univariate function ($n = 1$) [Th. 6.1.3 in [BM]].

We generalize the result to a multivariate polynomial case.

Theorem 1 *The lower bound on the paraellel computation time of any multivariate polynomia $f(x_1,\dots,x_n)$ is $\lceil \log_2(max_{\deg} f(x_1,\dots,x_n))\rceil$.*

This theorem can be stated using the arithmetic circuit model as:

Theorem 2 *If a circuit* **G** *evaluates f then depth(**G**) is not less than $\log_2(max_{\deg} f(x_1,\dots,x_n))$.*

Proof: The proof is similar to Kung[K], where the degree of the univariate polynomial should be replaced by the $max_{\deg}$ of a multivariate polynomial. □

Corollary 1 *The same theorem can be generalized to any multivariate rational function by defining $max_{\deg}$ of a function $\frac{p(x_1,\dots,x_n)}{q(x_1,\dots,x_n)}$ as $\max\{max_{\deg}(p), max_{\deg}(q)\}$.*

3 The complexity of the problem

Hyafil[H] showed that any polynomial P of degree d that can be computed with $C(P)$ $\{+,-,*\}$ operations serially, can be computed in parallel in time proportional to $\log d(\log C + \log d)$. Unfortunately, his method, in general, requires $C^{\log d}$ processors.

Valiant-Skyum-Berkowitz-Rackoff[VSBR] showed that any multivariate polynomial of degree d that can be computed serially in C steps can be computed in parallel in $O(\log d)(\log C + \log d))$ steps using $(Cd)^{O(1)}$ processors.

We define a "dense polynomial" as a polynomial of n indeterminates that has $max_{\deg} = d_p$, where its evaluation requires $\Omega(d_p^n)$ serial steps. We present an algorithm, that in a case where the multivariate polynomial is "dense", then its parallel time is almost $\log C$ and the number of processors is $O(C)$.

Dense polynomials are actually representing the worst-case analysis for serial evaluation and the aim of this paper is to show that its parallelization will speed-up the computation.

Our main result indicates that Valiant et. al.[VSBR] bound is not sufficiently tight in the case of dense multivariate polynomials. Their result for the parallel complexity is quite good for sparse systems where $C = O(nd)$. But for denser systems, their bound can be improved. Our result for the parallel complexity is quite closed to $\log C$, which is the theoretical but not achievable optimum for any PRAM parallel system.

Definition 2 *Let $T(d_p, n)$ be the time that parallel computation of polynomials in n varaibles and $max_{\deg} = d_p$ necessiated on array of sufficient number of processors.*

We, first, consider the simple algorithm, which eventually improves Valiant et. al. bound for dense multivariate polynomials.

The simple algorithm. Let f be an arbitrary polynomial in n variables with $max_{\deg}(f) = d_p$.

$$f(x_1 \dots x_n) = \sum_{i=0}^{d_p} c_i(x_2 \dots x_n) x_1^i. \qquad (1)$$

By definition, we have:

$$T(d_p, n) = T(d_p, n-1) + T(d_p, 1) \tag{2}$$

Since the best bound for the single-variable polynomial is [MP]:

$$T(d_p, 1) = \log d_p + O(\sqrt{\log d_p}) \tag{3}$$

Hence,

$$T(d_p, n) = n \cdot \log d_p + n \cdot O(\sqrt{\log d_p}) \tag{4}$$

This result is $O(\log C)$. If the polynomial is "dense" , the result is $\log C + O(\sqrt{n \log C})$.

We shall try to decrease the bound closer to the theoretical limit which is $\log C$. Although the simple algorithm is quite closed to the limit.

Definition 3 *Let $D(r) = \frac{1}{2}r(r+1)+1$ where r is a natural number. Hence, $D(r) = D(r-1) + r$.*

Assumptions: Let f be an arbitrary polynomial in n variables with $max_{\deg} = d_p$ where $2^{D(r-2)} \leq d_p < 2^{D(r-1)}$.

The exact location of d_p in $[2^{D(r-2)},\ 2^{D(r-1)})$ is determined by β, $\beta = D(r-1) - \log d_p$. Hence, $0 < \beta \leq r-1$.

Let $Q(r,n)$ be $\max(T(d_p(r), n))$ where the maximum is taken over all the relevant polynomials.

Lemma 1 $Q(r,n) \geq D(r) + \log n$.

Proof: The proof is done by induction on n.

$Q(r,1) = D(r)$ See [MP].

$Q(r,n+1) \geq Q(r,n)+1$ since at least one more operation is needed to evaluate a polynomial in $n+1$ variables.

Substitutting the induction hypothesis into $Q(r,n+1)$ and using the fact that $1 + \log n \geq \log(n+1)$ and we have:

$$Q(r,n+1) \geq Q(r,n)+1 \geq D(r) + \log n + 1 \geq D(r) + \log(n+1) \tag{5}$$

The hypothesis holds for $n+1$. $\Box$

Lemma 2 *Let f be an arbitrary polynomial in n variables with $max_{\deg}(f) = d_p$, $2^{D(r-2)} \leq d_p < 2^{D(r-1)}$, then there exists a circuit which evalu f with depth $= T(d_p,n) \leq nD(r-1) + \log n + r$.*

Proof: The proof is done by introducing an algorithm based on Munro-Paterson[MP] splitting technique.

The proof is proceed by induction on r. The induction hypothesis is done on the depth: $T(d_p,n) \leq Q(r,n) = nD(r-1) + \log n + r$. We assume that there exists a circuit which evaluates any arbitrary multivariate polynomial in *depth* $= Q(r,n)$. To verify for $r = 1$:

$$P_1(x_1,\ldots,x_n) = \sum_{i_1=0}^{1} \cdots \sum_{i_n=0}^{1} a_{i_1 \ldots i_n} x_1^{i_1} \ldots x_n^{i_n} \tag{6}$$

Obviously, there are at most 2^n possible operands in the summation due to all possible combinations. Therefore, at most n parallel additions after $\log n$ parallel time for multiplications in each monomial and one multiplication by the coefficient. In total: $n + \log n + 1 = nD(0) + \log n + 1$. (Munro and Paterson proved the lemma for a single variable case).

So for $n = 1$ we have that $Q(r, 1) = D(r)$.

To prove the lemma we construct the difference equation:

$$T(d_p(r+1), n) \leq Q(r+1, n) = rn + 1 + \max(Q(r,n), D(r-1) + r + \log n)$$
$$= rn + 1 + \max(Q(r,n), D(r) + \log n) \qquad (7)$$

This equation emerged from the nature of the circuit which is typical to the splitting method.

Any arbitrary polynomial f with high parallel degree can be composed from lower parallel degree polynomials.

Assume that $max_{\deg}(f) = d_p = 2^{D(r)} - 1$ then f can be decomposed as:

$$f(x_1, \ldots, x_n) = \sum_{i_1=0}^{2^r-1} \cdots \sum_{i_n=0}^{2^r-1} q_{i_1 \ldots i_n}(x_1 \ldots x_n) x_1^{i_1 2^{D(r-1)}} \ldots x_n^{i_n 2^{D(r-1)}} \qquad (8)$$

Each of the polynomials $q_{i_1 \ldots i_n}(x_1 \ldots x_n)$ has $max_{\deg}$ less than $2^{D(r-1)}$.

According to the induction hypothesis each $q_{i_1 \ldots i_n}(x_1 \ldots x_n)$ takes at most $Q(r, n)$ time units.

In parallel each monomial $x_1^{i_1 2^{D(r-1)}} \ldots x_n^{i_n 2^{D(r-1)}}$ can be computed in at most $D(r-1) + r + \log n$ time units.

Then the stage of the summation and multiplication takes at most $nr + 1$ parallel steps.

Now we use the result of Lemma 1 and have $Q(r+1, n) = rn + 1 + Q(r, n)$ and w know that $Q(1, n) = n + \log n + 1$. By the summation:

$$Q(r+1, n) = Q(1, n) + n \sum_{j=1}^{r} j + r = n + \log n + r + 1 + n\frac{r(r+1)}{2}$$

$$= n[1 + \frac{r(r+1)}{2}] + \log n + r + 1 = nD(r) + \log n + r + 1 \qquad (9)$$

Hence $Q(r+1, n) = nD(r) + \log n + r + 1$ or $Q(r, n) = nD(r-1) + \log n + r$. □

Lemma 3 *The depth of the circuit of f is bounded by: $n(\log d_p + \beta) + \log n + O(\sqrt{\log d_p})$ for $d_p > 1$. β is determined by the location of d_p on the strip $2^{D(r-2)} \leq d_p < 2^{D(r-1)}$. $0 < \beta < \sqrt{\log d_p}$.*

proof: Since $\log d_p = D(r-1) - \beta$ and $r \approx \sqrt{\log d_p}$ then

$$depth = T(d_p, n) \leq n(\log d_p + \beta) + \log n + \sqrt{\log d_p} \quad \square$$

Theorem 3 *An arbitrary dense polynomial in n variables can be computed in parallel in less than $\log C + \sqrt{n \cdot \log C}$ steps using only $O(C)$ processors. In fact, the bound is quite closed to the theoretical time limit which is $\log C$.*

Proof: By defintion, C the serial complexity of the computation of a dense polynomial, is $\Omega({d_p}^n)$. By applying the circuit described in Lemma 2 and Lemma 3, we have the required circuit. The number of processors required for the fast parallel evaluation is computed as follows:

Let $N(n,r)$ denote the maximum number of processors required at the r-th stage in evaluating an arbitrary polynomial in n variables. According to the architecture of the circuit, we have:

$$N(n,r+1) = 2^{rn}N(n,r) + \{generators \quad of \quad all \quad high \quad powers\}+$$

$$\{monomials \quad generators \quad , multipliers \quad and \quad summation\} \tag{10}$$

where we know that $N(n,1) < 3 \cdot 2^n$.

In Eq. (10) the first term $2^{nr} \cdot N(n,r)$ dominates the others. Generation of all high powers requires $O(n \cdot 2^{D(r)})$ processors. The multiplications and the summation require $O(2^{nr})$ processors. Thus, we can say:

$$N(n,r+1) = 2^{rn}N(n,r) + O(n \cdot 2^{D(r)}) + O(2^{nr}) \tag{11a}$$

$$N(n,r+1) = 2^{rn}N(n,r) + \alpha_1 n 2^{D(r)} + \alpha_2 2^{nr} \tag{11b}$$

$$N(n,1) < 3 \cdot 2^n \tag{12}$$

The leading term is $2^{rn} \cdot N(n,r)$. Therefore, after $r-1$ iteratio we can have that

$$N(n,r) = O(2^{n(\frac{r(r-1)}{2}+1)}) = O(2^{nD(r-1)}) = O(2^{n \log d_p}) \tag{13}$$

The final result is $O(2^{n \log d_p}) = O(d_p^n)$ which is $O(C)$ in the case of dense multivariate polynomial. $\square$

The results are better than Hayfil[H] and Valiant et. al.[VSBR] for this case. Generally speaking, we may say that when the system becomes denser, the bound is $O(\log C)$, which is a better bound than Hyafil and Valiant et. al. bound.

4 Conclusions

1. The obtained circuit is almost optimal in time, since $\log C$ is the theoretical optimum for any PRAM parallelization.
2. The circuit can be implemented in a "modular" VLSI-architecture using only $O(C)$ processors. The structure is dictated either by the simple algorithm or by the later one.
3. The criterion for parallel complexity of multivariate polynomials evaluation in n varaiables should be expressed by n and the "maximal-degree" ($max_{\deg}$).

References

[BM] Borodin, A., Munro, I.: The Computational Complexity of Algebraic and Numeric problems, American Elsevier Publishing Company, 1975.

[K] Kung, H.: New Algorithm and Lower Bounds for the Parallel Evaluation of Certain Rational Expressions and Recurrences, J. ACM, vol. **23**, no. 2 (1976) 252-261.

[H] Hyafil, K.: On the Parallel Evaluation of Multivariate Polynomials, Proc., 10th ACM Symposium on Theory of Computing, (1978) pp. 193-195.

[VSBR] Valiant, L.G., Skyum, S., Berkowitz, S., Rackoff, C.: Fast Parallel Computation of Polynomials using Few Processors, SIAM J. Comput., **12**, No. 4, (1983) 641-644.

[MP] Munro, I., Paterson, M.: Optimal Algorithms for Parallel Polynomial Evaluation, J. of Computer and System Science **7**(1973) 189-198.

This article was processed using the LaTeX macro package with LLNCS style