© 2012 г. Н.А. ЛУКИН, канд. техн. наук
(ИМАШ УрО РАН, НПО автоматики им. ак. Н.А. Семихатова)

ДЕКОМПОЗИЦИОННЫЕ И КЛЕТОЧНЫЕ АЛГОРИТМЫ ПАРАЛЛЕЛЬНОГО ВЫЧИСЛЕНИЯ ПОЛИНОМОВ В БАЗИСЕ СХЕМ ИЗ ФУНКЦИОНАЛЬНЫХ ЭЛЕМЕНТОВ СБИС[1]

Степенные многочлены (полиномы) являются одними из основных вычислительных процедур для систем реальном времени. В работе рассматриваются параллельные алгоритмы вычисления полиномов с помощью специализированных частично-однородных параллельных архитектур, состоящих из множества процессорных элементов. Предполагается, что такие архитектуры реализуются в виде либо отдельной СБИС, либо их части, т.е. кремниевого кластера. Исследование полиномов проведено на основе анализа двух основных метрических характеристик их алгоритмических графов – ширины D и высоты H и взаимосвязей между ними для конкретных типов полиномов. В работе рассмотрены две группы параллельных алгоритмов вычисления полиномов. Алгоритмы первой группы базируются на представлении исходного полинома в виде параллельной и последовательной частей. Алгоритмы второй группы базируются на иерархическом представлении полинома степени m (макро-полином) в виде клеточной структуры, состоящей из полиномов степени l, где $l < m$. (микро-полиномов). Для каждого из рассматриваемых типов полиномов получены аналитические соотношения между D и H, которые используются для оптимизации вычислений по параметрам сложности вычислений в базисе функциональных элементов СБИС. В ряде случаев получены условия существования оптимальных полиномов.

DECOMPOSITIONAL AND CELLULAR PARALLEL ALGORITHMS FOR POLYNOMIALS EVALUATION ON THE LEVEL OF FUNCTIONAL ELEMENTS OF VLSI /N.A. Lookin (Institute of Engineering Sciences, Komsomolskaya, 34, Yekaterinburg 620219, Russia, E-mail: nicklookin@mail.ru). Polynomials are the base computation procedures for real-time systems algorithms. The paper is devoted to parallel algorithms of polynomial evaluation by means of special-purpose partially homogeneous 2D arrays of processing elements. Such architectures may be realized as a VLSI or its part. Our investigation is based on the analysis of two main parameters of algorithmic graphs – width D and height H. The two groups of parallel algorithms for polynomial are considered. The first group of algorithms is based on representation of polynomial as a combination of sequence and parallel parts. The second group of algorithms is based on representation of any polynomial of

[1] Работа выполнена при частичной поддержке гранта РФФИ 11-01-12126-офи-м-2011 и программы Президиума РАН № 18 "Алгоритмы и математическое обеспечение для вычислительных систем сверхвысокой производительности" при поддержке УрО РАН (проект 12-П-1-1028).

degree *m* (macro-polynomial) as a cellular structure consisting of polynomial of degree l where $m > l$ (micro-polynomial). For each type of polynomial the formulas connected parameters *D* and *H* are obtained. These analytical ratios are used for optimization of computations by complexity. For the some cases we obtained conditions for optimal polynomial in the analytical forms.

1. Введение

Степенные многочлены (полиномы) являются одними из основных вычислительных процедур для реализации алгоритмов решения задач повышенной сложности в реальном времени. Среди применений полиномов различного вида можно выделить цифровую фильтрацию (в том числе один из самых распространенных для алгоритмов навигационных систем – фильтр Калмана), криптографию, вычисление математических функций. Необходимость в быстрых методах вычисления полиномов обусловлена широким распространением этого математического аппарата практически во всех задачах вычислительной математики. Например, непрерывное усложнение компьютерных моделей при решении задач математической физики выдвигает на первый план создание новых, еще более быстрых, чем существующие, методов вычисления функций, а, следовательно, дальнейшего ускорения вычисления полиномов. В настоящей работе рассматриваются параллельные методы вычисления полиномов для целей приближения математических функций.

Среди возможных разновидностей суперкомпьютерных решений выделим класс суперкомпьютеров, встроенных в состав систем реального времени (*EScomp – Embedded Supercomputers*) [1]. Главное требование к таким суперкомпьютерам – сверхвысокая производительность, за счет чего обеспечивается реализация сложных вычислительных алгоритмов в реальном времени. В то же время они должны иметь минимальные объемно-массовые характеристики и потребляемую мощность. В пределе *EScomp* могут занимать ресурсы не более чем одной СБИС [2], что в ряде применений требуется уже сегодня. В связи с этим решение проблем оптимизации архитектур как по критериям производительности, аппаратных затрат, так и по интегральным (удельным) показателям приобретает первостепенное значение. По отношению к параллельным методам вычисления полиномов этот означает, прежде всего, их оптимизацию по критерию сложности вычислений. В качестве меры сложности в настоящей работе будем иметь в виду сложность функций в базисе схем из функциональных элементов L [3] как случай битовой сложности вычислений, введенной А.Н. Колмогоровым [4]. Соответственно, под быстрыми алгоритмами будем понимать алгоритмы, битовая сложность которых может быть оценена как $O\left(M\left(n\right)log^{c}n\right)$, где с – некоторая константа [5]. В настоящей работе параллельные методы вычисления полиномов рассматриваются в предположении их реализации с помощью специализированных процессорных архитектур СБИС. В этом случае функциональными элементами являются логические вентили, реализующие автоматную логику Мура или Мили.

Реализация параллельных процессорных архитектур в виде СБИС часто требует пересмотра критериев оптимизации вычислений. В частности, большое (если не решающее) значение приобретают параметры площади, которую занимает архитектура, реализующая алгоритм, и мощность, потребляемая предполагаемой схемой. В свое время в ряде работ было показано, что локальность информационных связей между

блоками и их комбинациями дает возможность оптимизации по указанным параметрам. Поэтому однородность связей и однотипность блоков (устройств) наилучшим образом отвечает специфике реализации архитектуры на уровне СБИС. С другой стороны, требование полной однородные архитектуры вносит слишком жесткие ограничения в реализуемость алгоритмов за приемлемое время, поэтому разумно говорить о частичной, т.е. не 100%-ной однородности. В настоящей работе предполагается реализация параллельных методов вычисления полиномов с помощью частично однородных двумерных (2*D*) процессорных архитектур.

В применении к графовой форме изображения алгоритмов будем использовать в качестве метрических характеристик сложности, следуя В.В. Воеводину [6], параметры ширины *D* и высоты *H* алгоритмических графов. Вершины алгоритмического графа однозначно связаны с конкретными вычислительными процедурами или функциональными преобразованиями, поэтому общее число вершин графа дает верхнюю оценку сложности вычислений в виде *L*. Для случая реализации вычислений на параллельных архитектурах параметр *D* косвенно связан с количеством параллельно работающих устройств обработки данных (процессоров, функциональных элементов), а параметр *H* – с числом тактов работы архитектуры. В связи с этим предметом исследования в настоящей работе являются алгоритмические графы, основными параметрами которых являются *D* и *H*. В содержательном плане это означает исследование сложности вычислений по критериям аппаратной и временной сложности [7].

Таким образом, настоящая работа посвящена исследованию параллельных методов вычисления полиномов, оптимизированных по параметрам аппаратной и временной сложности вычислений в базисе схем из функциональных элементов СБИС на классе частично-однородных процессорных 2*D*-архитектур.

2. Тривиальные параллельные методы вычисления полиномов

Будем рассматривать полиномы вида

(1)
$$P(X1, X2) = \sum_{i=o}^{m} A_{i(X1)} (X2)^i$$

где $A_{i(X1)}$ – коэффициенты полинома, которые в нашем случае представляют собой функции действительного аргумента m, для которых справедливо условие: $\lim_{i\to\infty} | A_{i(X1)} | \leq \alpha$, где α - произвольное действительное число; *m* – степень полинома. В общем случае $X1 \neq X2$, таким образом, полином P является функцией двух переменных. Полином в виде (1) обобщает множество полиномов различных типов.

Пример. Пусть $X1 = \sum_{j=1}^{n_1} a_j 2^{-j}$, $X2 = \sum_{k=1}^{n-n_1} b_k 2^{-(n_1+k)}$ где $a_j, b_k \in E_2$; $n1 < n$; $X1 + X2 = X$; n1 – число двоичных разрядов *X*1; n – число двоичных разрядов *X*. Это соответствует представлению некоторой аналитической функции действительного аргумента F(X) в виде ряда Тейлора. При этом *X*1 – значения аргумента, в которых заданы опорные значения функции F(*X*), а X2 – приращения к X1. Кроме того, это тот случай, когда множества значений X разбивается на подмножества X1 и X2 так что $\{X\}=\{X1\}\cup\{X2\}$ при

$\{X1\} \cap \{X2\} = \varnothing$. Тогда $X1 \in \left[0, 1 - 2^{-n_1}\right);\ X2 \in \left[0, 2^{-n_1} - 2^{-n}\right)$. Таким образом, для рассмотренного представления $X1$ и $X2$ имеем полином, представляющий произвольную аналитическую функцию одного аргумента, при этом имеет место простая разделительная декомпозиция X в виде X1 и X2, что важно с точки зрения реализации самого процесса вычисления полинома.

Вычисление P($X1$, $X2$) по формуле (1) может быть произведено различными способами, каждый из которых обладает соответствующей степенью параллелизма. Конкретизируем все разнообразие в параллельных алгоритмах вычисления полиномов следующими условиями:

– все разряды X обрабатываются одновременно;

– операции умножения выполняются на функциональных узлах, называемых, соответственно, полями конъюнкторов и сумматоров (ПС), которые реализуют одновременное перемножение всех разрядов сомножителей. Тем самым при выполнении умножения достигается максимальная степень параллелизма;

– операции сложения или вычитания реализуются на комбинационных многоразрядных сумматорах с одновременным формированием разрядных сумм и последовательным распространением переносов;

– вычисление коэффициентов полинома производится с помощью отдельных функциональных узлов, которые, в свою очередь, могут быть реализованы либо в виде таблиц ПЗУ, либо в виде схем из функциональных элементов.

Таким образом, любой алгоритм вычисления полинома будет содержать следующие базовые операции – сложение двух чисел, вычитание двух чисел, умножение двух чисел, целая степень числа, вычисление коэффициентов A_i, выполнение произвольных булевых функций от n переменных. Для базовых операций примем параметры высоты и ширины алгоритмических графов, равными $H_b = 1$, $D_b = 1$ соответственно.

Рассмотрим некоторые параллельные алгоритмы вычисления P($X1$, $X2$).

2.1. Метод непосредственного функционального преобразования множества значений аргумента X во множество значений полинома P(X1, X2) (вариант максимального параллелизма или P_{max}-метод)

В данном случае предварительно осуществляется конкатенация $X1$ и $X2$ в один аргумент X разрядностью $n = n_1 + n_2$ бит. Затем производится вычисление параллельно всех разрядов полинома P(X) – n_X разрядов аргумента $X = X1 \cdot 2^{n_2} + X2$ одновременно преобразуются в n_P разрядов функции полинома:

(2)
$$[P(X)]_0 = [P(X1 \cdot 2^{n_2} + X2)]_0 = \left[\sum_{i=o}^{m} A_{i(X1)} (X2)^i\right]_0 = \varphi_0(X)$$
$$[P(X)]_1 = [P(X1 \cdot 2^{n_2} + X2)]_1 = \left[\sum_{i=o}^{m} A_{i(X1)} (X2)^i\right]_1 = \varphi_1(X)$$
.....

$$[P(X)]_{n_P-1} = [P(X1 \cdot 2^{n_2} + X2)]_{n_P-1} = \left\lfloor \sum_{i=o}^{m} A_{i(X1)} (X2)^i \right\rfloor_{n_P-1} = \varphi_{n_P-1}(X)$$

где φ_i – некоторая логическая функция от n_x аргументов, $[P(X)]_j$ – j-ый разряд кода, представляющего $P(X)$.

При выполнении данного алгоритма все n_P разрядов $P(X)$ вычисляются независимо и одновременно. Для P_{max}-метода высота и ширина алгоритмического графа равны $H = 1$, $D = n_P$ соответственно.

Принципиальной (а в большинстве случаев – и фундаментальной) трудностью в случае P_{max}-методов является нахождение (или построение) оценок сложности реализации конкретных функций φ_j. Если $P(X)$ является монотонной функцией от "совокупного" аргумента X (случай элементарных математических функций), то, как показывают исследования [8], существуют эффективные методы вычисления "старших" разрядов $P(X)$, когда функции φ_{n_P-1}, φ_{n_P-2}, …, $\varphi_{n_P-\beta}$, где $n_P > \beta$, вычисляются как монотонные логические функции от разрядов аргументов $X_{n_X-1}, X_{n_X-2}, ..., X_{n_X-q}$, где $n_X > n_q$. Все же остальные n_p-β функций φ в общем случае представляют собой произвольные логические функции от n_x переменных. Их нахождение представляет собой задачу, относящуюся к классу переборных, поэтому в настоящее время эффективные методы реализации математических функций в виде только логических функций практически отсутствуют. С другой стороны, развитие микроэлектроники позволяет для $n \le 32$ рассматривать возможность реализации функций φ в виде таблиц ПЗУ для вариантов встроенных суперкомпьютеров без значительных ограничений на объемно-массовые характеристики (например, стационарные установки моделирующих комплексов), а для $n \le 16$ – и для бортовых суперкомпьютеров.

Таким образом, P_{max}-методы, имея минимальную высоту алгоритмических графов, равную 1, имеют максимальную ширину, равную n_X при том, что реализация таких методов для $n > 32$ является проблематичной.

2.2. Метод параллельного вычисления полинома (P-метод).

P-метод предусматривает независимое и параллельное вычисление коэффициентов $A_{i(X1)}$, i=0,…, m, степеней $(X2)^1$, $(X2)^2$, …, $(X2)^m$ и произведений вида $A_{i(X1)} \cdot (X2)^i$. Этому методу соответствует развернутая запись полинома:

$$P(X1, X2) = A_{0(X1)} + A_{1(X1)} \cdot X2 + A_{2(X1)} \cdot (X2)^2 + ... + A_{m(X1)} \cdot (X2)^m \quad (3)$$

ГСА P-метода в ярусно-параллельной форме (ЯПФ) дает оценки высоты и ширины алгоритмического графа – $H = \lfloor \log_2 m \rfloor + 2, D = 2m - 1$.

2.3. Метод параллельного вычисления полинома без автономного вычисления степеней X2 (P1-метод).

P1-метод реализует вычисление полинома следующим образом:

$$P(X1, X2) = A_0 + A_1 \cdot X2 + A_2 \cdot \psi_1 \cdot X2 + A_3 \cdot \psi_2 \cdot X2 + ... + A_{m-1} \cdot \psi_{m-2} \cdot X2 + A_m \cdot \psi_{m-1} \cdot X2$$

(4)

где $\psi_j = \psi_{j-1} \cdot X2$, j=2,3,…,$m$-1, $\psi_1 = X2$.

В ГСА этого метода отсутствуют автономные операторы вычисления степеней $X2$, а вместо этого на каждом ярусе производится вычисление функций вида ψ_j, представляющих собой рекуррентное умножение X2 на самого себя. В его состав входят лишь три вида операторов – операторы вычисления коэффициентов A_i, i=0,…,m, умножения и сложения. Параметры высоты и ширины алгоритмического графа для P1-метода имеют вид: $H = m+2$, $D = 4$. В дальнейшем эти параметры для ГСА различных методов будут сведены в Таблицу 1 (см. Приложение).

2.4. Метод параллельного вычисления полинома по схеме Горнера (P2-метод)

Данный метод, широко применяемый в практике вычислений, реализует вычисление полинома следующим образом:

$$P(X1, X2) = A_0 + X2(A_1 + X2(A_2 + ... + X2(A_{m-1} + A_m \cdot X2)...))$$

(5)

Для P2-метода характерно полное отсутствие вычисления степеней X2. Число типов операторов такое же, как в алгоритме P1. Высота и ширина его ГСА приведены в Табл. 1.

Особенностью метода является полное распараллеливание на начальном ярусе вычислений и последовательные вычисления на всех остальных ярусах. Такая несбалансированность ярусов обуславливают относительно низкую его эффективность – общее число его операторов меньше, чем в алгоритме P1 примерно на треть, а высота его ГСА почти в два раза больше.

Последовательный характер вычислений P($X1$, $X2$) на ярусах 2,…,2m+1 дает возможность реализовать их на итеративных схемах с обратными связями, что резко уменьшает общее число ярусов в таком варианте ГСА (до трех ярусов) и общее число операторов без существенного увеличения высоты ГСА алгоритма. Кроме того, коэффициенты также могут вычисляться не одновременно, а на соответствующих ярусах, что делает итеративную ГСА еще более обоснованной. Таким образом, алгоритм P2 фактически является последовательным.

3. Декомпозиционные алгоритмы вычисления полиномов

Основой декомпозиционных методов является представление полинома P($X1$, $X2$) в виде (3) в виде двух частичных полиномов – $P_{1P}(X1, X2)$ и $P_{1S}(X1, X2)$, которые объединены между собой операцией суммирования:

$$P(X1, X2) = P_{1P}(X1, X2) + P_{1S}(X1, X2)$$

(6)

Такую декомпозицию назовем 1-кратной. В случае, когда исходный полином представляется в виде D частичных полиномов, будем говорить о D-кратной декомпозиции.

Рассмотрим два основных декомпозиционных алгоритмов вычисления полинома для случая 1-кратной декомпозиции.

3.1. Частично-параллельный алгоритм (Sp-алгоритм)

Алгоритм Sp реализует вычисление P(*X*1, *X*2) следующим образом:

$$P(X1,X2)=A_0+X2(A_1+X2(A_2+...+X2(A_{w-2}+X2\cdot A_{w-1})...)+$$
$$+A_w(X2)^w+A_{w+1}(X2)^{w+1}+...+A_{m-1}(X2)^{m-1}+A_m(X2)^m=P_{1S}(X1,X2)+P_{1P}(X1,X2)$$

где

(7)
$$P_{1S}(X1,X2)=A_0+X2(A_1+X2(A_2+...+X2(A_{w-2}+X2\cdot A_{w-1})...)$$

$$P_{1P}(X1,X2)=A_w(X2)^w+A_{w+1}(X2)^{w+1}+...+A_{m-1}(X2)^{m-1}+A_m(X2)^m$$

Из (7) видно, что 1-кратная декомпозиция исходного полинома дает возможность реализовать параллельные и независимые вычисления двух частичных полиномов – последовательной части P_{1S} и параллельной части P_{1P}. Особенностью Sp-алгоритма является итерационное вычисление младших (степени 0,1,…,w-1) и параллельное вычисление старших членов полинома (степени w,w+1,…,*m*). Другими словами, для членов полинома со степенями 0,1,…,w-1 применяется алгоритм P2, а со степенями w,w+1,…,*m* – алгоритм P либо P1. Такой подход целесообразно применять в том случае, если алгоритм P2, взятый в начале с целью минимизации затрат, не удовлетворяет требованиям на время выполнения. В некоторых работах показано, что сложность вычисления членов исходного полинома, приближающего произвольную аналитическую функцию действительного аргумента, уменьшается с увеличением степени X2 [9]. Это дает возможность, начиная с максимальной степени, реализовать параллельную схему вычисления, охватив ею некоторое количество членов полинома. Этим самым можно направленно уменьшать высоту ГСА алгоритма P2. Высота и ширина его ГСА приведены в Табл. 1.

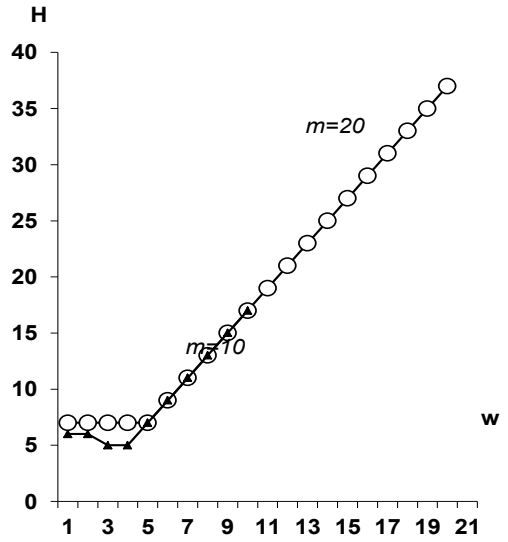

Рис.1. Зависимость высоты алгоритмического графа вычисления полинома от параметра w

Анализ параметра *H* для P2- и P-составляющих алгоритма Sp показывает, что имеет место уравнение:

(8)
$$2w-3=|\log_2(m-w)|$$

решение которого существует всегда и однозначно соответствует такому разбиению полинома, что граф для его части P2 будет иметь такую же высоту, что и для части P. Таким образом, номер w члена полинома, который определен как решение уравнения (8), соответствует 1-кратно декомпозиционному полиному, оптимизированному по критерию высоты его ГСА *H*, т.е. **по времени его вычисления**. Алгоритм Sp является весьма эффективным методом уменьшения высоты последовательного алгоритма P2, при этом незначительное увеличение числа операторов приводит к существенному

уменьшению высоты. На рис. 6 приведены зависимости параметра *H* данного алгоритма от различных *m* и w. Из анализа графиков видно, что корректно определенный параметр w может значительно уменьшить высоту графа (т.е. время вычисления) для последовательного алгоритма.

3.2. Частично-последовательный алгоритм (Ps-алгоритм)

Алгоритм Ps реализует вычисление полинома P(*X*1, *X*2) следующим образом:

$$P(X1,X2)=A_0+A_1\cdot X2+A_2\cdot(X2)^2+\ldots+A_{q-1}\cdot(X2)^{q-1}+(X2)^q[A_q+$$
$$+X2(A_{q+1}+\ldots+X2(A_{m-1}+X2\cdot A_m)\ldots)] \quad (9)$$

Как и в случае Sp-алгоритма имеем 1-кратную декомпозицию исходного полинома на два частичных – на P_{1P} и P_{1S}. При этом

$$P_{1P}(X1,X2)=A_0+A_1\cdot X2+A_2\cdot(X2)^2+\ldots+A_{q-1}\cdot(X2)^{q-1}$$

$$P_{1S}(X1,X2)=(X2)^q[A_q+X2(A_{q+1}+\ldots+X2(A_{m-1}+X2\cdot A_m)\ldots)] \quad (10)$$

Параметры высоты и ширина ГСА алгоритма Ps приведены в Табл. 1. Анализ формулы (9) показывает, что можно найти всегда такую степень q, начиная с которой в сторону ее увеличения, возможна последовательная (итерационная) схема вычисления фрагмента исходного полинома. При этом высота графа последовательного алгоритма (P2) не будет превышать высоты параллельного алгоритма (P). Степень q есть решение уравнения:

$$2(m-q)=|\log_2 q| \quad (11)$$

Между Sp- и Ps-алгоритмами имеется дуализм: первые w членов алгоритма Sp вычисляются по алгоритму P2, остальные (*m*-w)–по алгоритму P; в алгоритме Ps – наоборот.

Несмотря на некоторую схожесть математических формулировок эти алгоритмов, оценки высоты их графов существенно различаются. Например, при *m*=10 оптимальные степени полиномов лежат в диапазонах: для Sp-алгоритма – 2<w<4, для Ps-алгоритма – 8<q<9. Это приводит к различной сложности вычислений, в основном, за счет разного числа блоков вычисления коэффициентов полиномов. В алгоритме Sp требуется вычислять w логических функций от n1 переменных и 2(*m*-w) функций от (n-n1) переменных, а в алгоритме Ps нужно вычислять (*m*+1) функций от n1 переменных и (q-1) функций от (n-n1) переменных.

4. Клеточные алгоритмы вычисления полиномов

Основная идея клеточных алгоритмов вычисления полиномов заключается в выделении из полинома независимых блоков и представлении его комбинациями этих блоков. Один из таких методов может быть представлен в виде:

$$P(X1,X2)=(X2)^0\sum_{i=0}^{l-1}A_i(X2)^i+(X2)^1\sum_{j=0}^{l-1}A_{l+j}(X2)^j+\ldots+(X2)^{m-l+1}\sum_{k=0}^{l-1}A_{m-l+k+1}(X2)^k \quad (12)$$

Как видно из (11), этот метод позволяет независимо вычислять блоки полинома вида $(X2)^{n+1}\sum_{t=0}^{l-1}A_{n+1+t}(X2)^t$ с последующим итоговым суммированием. Мы добиваемся унификации алгоритмических блоков и связей между ними. Каждый блок содержит полином степени (l-1), который может быть вычислен по любому из описанных выше алгоритмов. Обозначим каждый i-ый "подполином" степени (l-1) как $P_i(X1,X2)$. Тогда исходный полином $P(X1, X2)$ может быть записан в следующем виде:

$$P(X1,X2)=P_0(X1,X2)+X2\cdot P_1(X1,X2)+\ldots+(X2)^{m-l+1}P_{m-l+1}(X1,X2) \quad (13)$$

Таким образом, мы получили клеточный полином, состоящий из клеток – частичных полиномов. Он также может вычисляться по описанным выше алгоритмам. Назовем $P_i(X1,X2)$ микрополиномом, а выражение (13) – макрополиномом. Ограниченный объем публикации не позволяет представить исчерпывающую классификацию всех видов подобных полиномов. Поэтому рассмотрим несколько клеточных параллельных алгоритмов вычисления функций, взяв за основу представление (12).

4.1. Параллельно-последовательные алгоритмы вычисления полиномов (PS-алгоритмы)

Общая запись таких алгоритмов выглядит так:

$$\begin{aligned}P(X1,X2)=A_0+X2\cdot(A_1+X2\cdot(A_2+\ldots+X2\cdot(A_{l-2}+X2\cdot A_{l-1})\ldots))+\\+(X2)^l(A_l+X2\cdot(A_{l+1}+\ldots+X2\cdot(A_{2l-2}+X2\cdot A_{2l-1})\ldots))+\\+(X2)^{m-l+1}(A_{m-l+1}+X2\cdot(A_{m-l+2}+\ldots+X2\cdot(A_{m-1}+X2\cdot A_m)\ldots))\end{aligned} \quad (14)$$

Как видно, параллельно и независимо производится вычисление всех $(m+1)/(l+1)$ микрополиномов по алгоритму P2 (схема Горнера) и их перемножение на "свои" степени $(X2)^{\alpha}$, где α =0, l, 2l,…, m-l+1 с последующим итоговым суммированием. Все микрополиномы имеют степень (l-1), что позволяет иметь одинаковую высоту для всех параллельных веток ГСА. Существуют две основные разновидности алгоритмов типа PS: PS1 – алгоритм с параллельным вычислением степеней X2 и PS2 – с последовательным вычислением степеней X2. Формула (14) соответствует алгоритму PS1. Обозначим каждый n_l-й микрополином как $(P2)_{nl}$, тогда (14) запишется в виде:

$$P(X1,X2)=(P2)_0+(X2)^l\cdot(P2)_l+\ldots+(X2)^{m-l+1}\cdot(P2)_{m-l+1} \quad (15)$$

Высота и ширина ГСА этого алгоритма приведены в Табл.1. Анализ показывает, что использование PS1-алгоритма существенно уменьшает высоту алгоритма P2 без увеличения его ширины.

Алгоритм PS2 выглядит так:

$$P(X1,X2)=(P2)_0+(X2)^l\cdot((P2)_l+\ldots+(X2)^l\cdot((P2)_{m-2l+1}+(X2)^l\cdot(P2)_{m-l+1})\ldots)) \quad (16)$$

Параметры высоты и ширины ГСА данного алгоритма приведены в Табл. 1. Алгоритм PS2 характеризуется значительно меньшей шириной, чем PS1 (примерно в

$m/l\lfloor \log_2 m/l \rfloor$раз. Клеточные алгоритмы вычисления полиномов вообще и алгоритмы PS1 и PS2, в частности, обладают следующим свойством: существует такая степень микрополинома $0 < l^* < l_{max}$, при которой $H=H_{min}$ среди всех возможных значений высоты.

Для алгоритма PS2, например, $l^*_{PS2} = \left\lfloor \sqrt{m+1} - 1 \right\rfloor$. Если учесть, что ширина H таких алгоритмов является монотонной функцией от l, то, выбрав l параметром декомпозиции, можно строить минимальные по высоте (времени) алгоритмы даже не для максимальных значений ширины.

4.2. Последовательно-параллельные алгоритмы вычисления полиномов (SP-алгоритмы)

Математические формулировки SP-алгоритмов, основой которых является (16), во многом похожи на формулировки PS-алгоритмов. Отличие состоит в том, что последовательно или итерационно вычисляется сам макрополином, в то время как микрополиномы параллельно вычисляются с помощью алгоритмов типа P. В Табл. 1 приведены параметры высоты и ширины одной из модификаций алгоритма SP.

5. Сравнительный анализ рассмотренных алгоритмов

Все рассмотренные алгоритмы образуют некоторую последовательность, на одном конце которой максимально параллельные методы вычисления функций с помощью полиномов (алгоритм P_{max}), а на другом – практически полностью последовательные методы (алгоритм P2). Тем не менее, рассмотренная совокупность не исчерпывает собой все возможное множество параллельных алгоритмов вычисления полиномов. В частности, в анализ не вошли, например, макрополиномы, в которых микрополиномы являются клеточными. Кроме того, мы не рассмотрели *D*-кратно декомпозиционные алгоритмы. Другим примером не рассмотренных параллельных алгоритмов вычисления функций являются алгоритмы, основанные на методах В. Пана [10]. Можно сказать, что подходы, изложенные в настоящей работе, являются начальными на пути построения общей классификации параллельных алгоритмов вычисления полиномов.

Приведем основные результаты сравнительного анализа рассмотренных алгоритмов.

5.1. Зависимость высоты ГСА алгоритмов от степени макрополиномов m

Все рассмотренные алгоритмы делятся с точки зрения влияния m на высоту H на два множества:

a. Алгоритмы с линейным увеличением H при линейном увеличении m (“линейные” алгоритмы). Это алгоритмы P1, P2, PS2, SP, зависимости $H(m)$ для которых приведены на рис. 2, 3, 4.

b. Алгоритмы с логарифмическим увеличением H при линейном увеличении m (“логарифмические” алгоритмы). Это алгоритмы P, Sp, PS1 зависимости $H(m)$ для которых приведены на рис. 2, 5, 6. Наиболее слабая зависимость H от m присуща алгоритму PS1.

Из анализа следует практическая рекомендация: если максимально возможная степень микрополинома ограничена величиной l, но по условию требуется $m >> l$, то с точки зрения обеспечения минимума увеличения высоты H предпочтительнее пользоваться алгоритмом PS1.

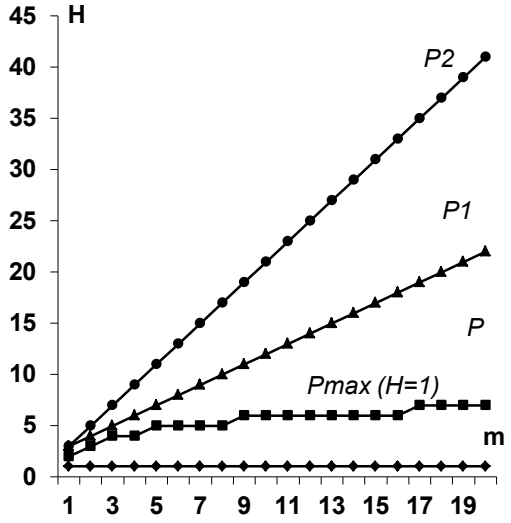

Рис. 2. Зависимость высоты ГСА алгоритмов Pmax, P, P1 и P2 от степени макрополинома

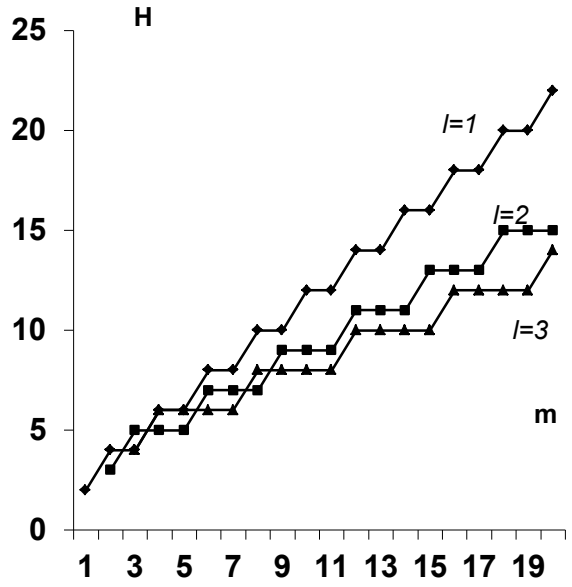

Рис. 3. Зависимость высоты ГСА алгоритма SP от степени макрополинома m для l=1,2,3

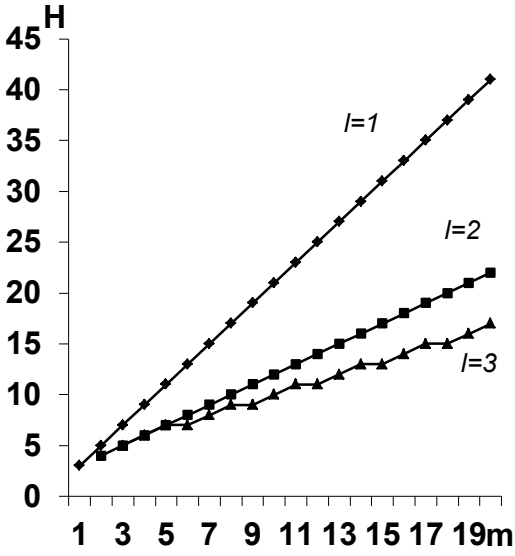

Рис. 4. Зависимость высоты ГСА алгоритма PS2 от степени макрополинома m для l=1,2,3

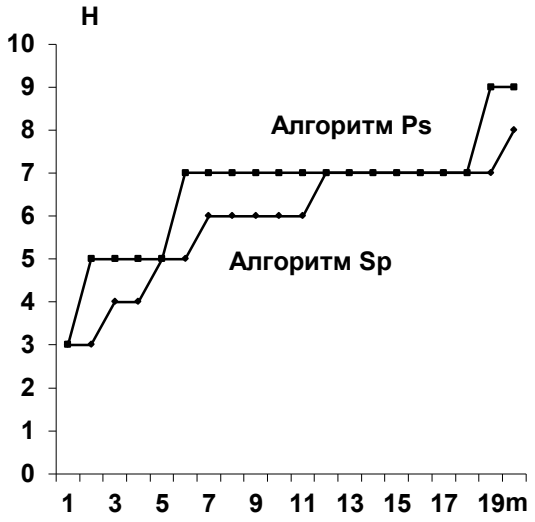

Рис. 5. Зависимость высоты ГСА алгоритмов S_P и P_S от степени макрополинома m для рациональных значений l

5.2. Зависимость высоты ГСА алгоритмов от степени микрополиномов l,w,q

Все рассмотренные алгоритмы с точки зрения влияния l,w,q на *H* делятся на следующие:

a. ”Линейные”. Это алгоритмы типа PS1, в которых имеется практически линейная зависимость *H* от l (см. рис. 7).

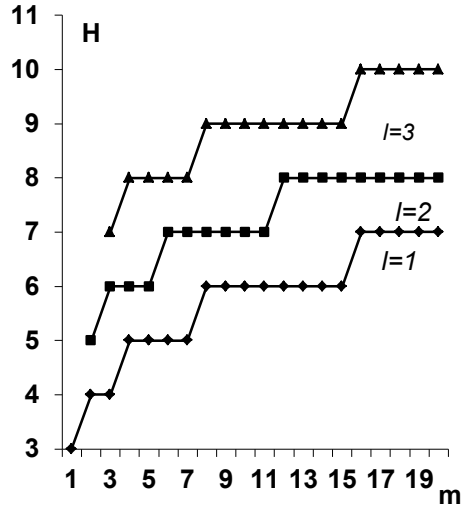

Рис. 6. Зависимость высоты ГСА алгоритма PS1 от степени макрополинома m для l = 1,2,3

b. ”Обратно пропорциональные”. Это алгоритмы типа SP, которых имеет место практически монотонное уменьшение *H* от l (см. рис. 8).

c. “Локально минимальные”. Это алгоритмы типа PS2, для которых характерно наличие окрестности локального минимума в зависимости *H*(l) (см. рис. 9), что может быть использовано для оптимизации по критерию высоты. Внимания заслуживают алгоритмы Sp и Ps. При рационально выбранных параметрах декомпозиции (w или q), когда обе главных ветви алгоритмов становятся равными по высоте, зависимость T(*m*) приобретает логарифмический характер (рис. 5). В этом случае возможно достижение минимальных по высоте алгоритмов.

Переменные l,w,q, являясь параметрами декомпозиции, связаны с количеством вычисляемых микрополиномов или, что то же самое, со степенью макропараллелизма. Для алгоритмов Sp, Ps и PS2 существуют величины эффективного макропараллелизма, всякое изменение которых приводит к увеличению высоты ГСА алгоритма. Поэтому для этих алгоритмов существуют пределы распараллеливания – для алгоритмов Sp и Ps –это рациональные значения w и q (см. (8),(11)), для PS2 – l^*_{PS2} .

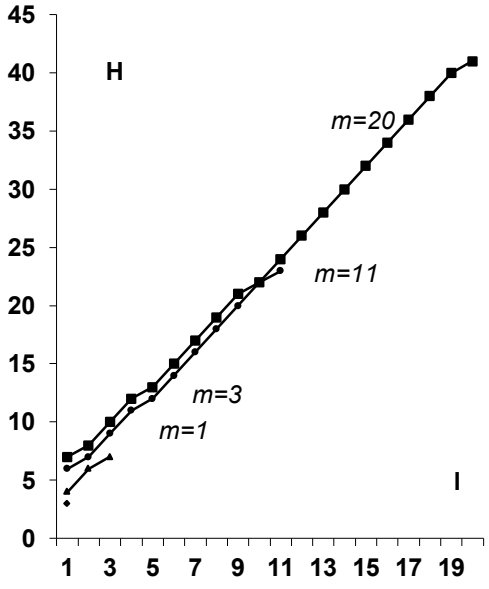

Рис. 7. Зависимость высоты ГСА алгоритма PS1 от степени микрополинома l для m=1,3,11,20

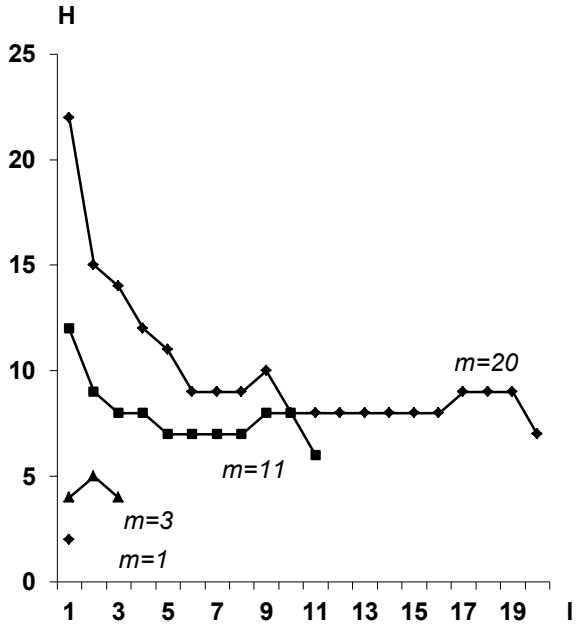

Рис. 8. Зависимость высоты ГСА алгоритма SP от степени микрополинома l для m=1,3,11,20

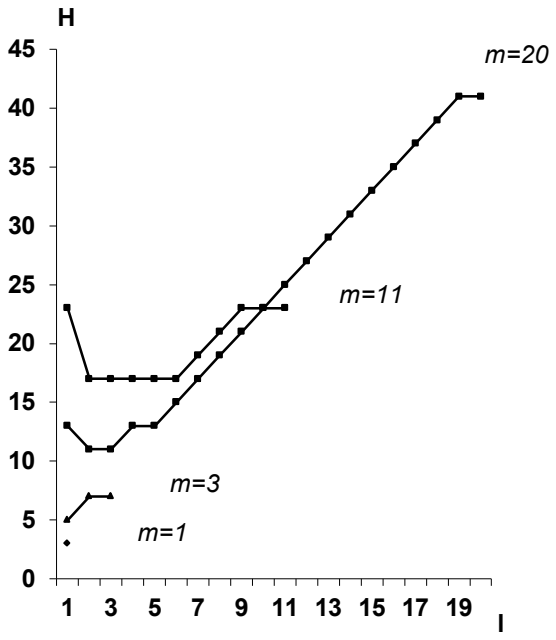

Рис. 9. Зависимость высоты ГСА алгоритма PS2 от степени микрополинома l для m=1,3,11,20

Для алгоритмов типа SP характерно практически монотонное уменьшение H с увеличением l, т.е. эти алгоритмы поддаются предельному распараллеливанию. Например, при l=m $H_{SP} \cong H_P$.

Сравнение минимальных значений высот ГСА для описанных алгоритмов показывает, что в случае алгоритмов типа Sp и Ps можно достичь таких же по порядку величин, как и для SP, в то время как минимальная высота ГСА алгоритма PS2 всякий раз будет примерно в O(l/log2l) раз больше, чем в алгоритмах Sp, Ps и SP. Поэтому алгоритмы Sp, Ps и SP более предпочтительны с точки зрения достижения минимальной высоты их ГСА.

5.3. Сравнение алгоритмов по критерию T·H

Сопоставив параметр H со временем реализации алгоритма, а D – с максимально необходимым числом процессоров, можно провести сравнение всех алгоритмов по обобщенному критерию сложности. Этот критерий отражает пространственно-временные затраты на реализацию алгоритма. Подобно распространенным критериям типа A·T или $A \cdot T^2$ [11], пусть соответствующий критерий в нашем случае имеет вид $H \cdot D$. В Табл. 2 (см. Приложение) приведены значения этого критерия для всех описанных в настоящей работе алгоритмов.

Как видно из таблицы, для всех алгоритмов параметр $H \cdot D$ зависит либо от m, либо от m и степеней микрополиномов l,w или q. Исключение составляет алгоритм P$m$ax, для которого характерна всегда практическая независимость высоты ГСА от этих параметров. Что же касается остальных алгоритмов, то:

- алгоритмы типов P, Sp, Ps являются самыми “затратными” из-за значительной ширины;
- алгоритмы типов SP и PS1 имеют высокое значение параметра $H \cdot D$ в области больших значений l, причем для SP это связано со значительной шириной, а для PS1–высотой. В области малых l эти алгоритмы весьма экономичны;
- алгоритмы P1,P2 и PS2 – самые экономичные, причем, для PS2 имеется абсолютный минимум $H \cdot D$ при $l = l^*$. Этот алгоритм следует признать рациональным.

Большинство из рассмотренных алгоритмов могут быть распараллелены по степеням макро - и микрополиномов, при этом практически всегда можно достичь требуемых значений высоты и ширины их ГСА. Таким образом, эти алгоритмы могут быть отнесены к параметрически настраиваемым.

6. Заключение

Исследованы основные типы параллельных алгоритмов вычисления полиномов.

Показано, что существует два основных параметра распараллеливания: степень макрополинома *m* и степени микрополиномов – l,w,q. Наилучшим сочетанием высоты и ширины обладают клеточные алгоритмы, в первую очередь, PS2. Установлено, что для всех декомпозиционных алгоритмов имеются рациональные степени микрополиномов, позволяющие однозначно достигать минимальных значений высоты их ГСА.

Проведенные исследования представляют начальный шаг классификации параллельных вычислений полиномов и являются основой методов синтеза алгоритмов, оптимизированных по критериям аппаратной или временной сложности.

ПРИЛОЖЕНИЕ

Таблица 1

Количественные характеристики параллельных алгоритмов вычисления полиномов P(*X*1, *X*2)

Наименование алгоритма	Высота ГСА (*H*)	Ширина ГСА (*D*)
Pmax	1	n
P	$\lceil \log_2 m \rceil + 2$	2*m*-1
P1	m+2	4
P2	2*m*+1	2
Sp	$\mathrm{Max}\{(2w-1),(\lceil \log_2 (m-w) \rceil + 2\} + 1$	2(*m*-w)+1
Ps	$\mathrm{Max}\{(\lceil \log_2 q \rceil + 2),(\lceil 2(m-q+1) \rceil\} + 1$	2q-1
PS1	$2l+1+\left\lceil \log_2 \left(\frac{m+1}{l+1}\right) \right\rceil$	$2\left\lceil \frac{m+1}{l+1} \right\rceil + 1$
PS2	$2l+1+2\left\lceil \frac{m+1}{l+1} - 1 \right\rceil$	3
SP	$\lceil \log_2 m \rceil + 2 + 2\left\lceil \frac{m+1}{l+1} - 1 \right\rceil$	2l

Таблица 2

Значение обобщенного критерия *HD* для параллельных алгоритмов вычисления полиномов

Наименование алгоритма	$H \cdot D$
Pmax	N
P	$(2m\text{-}1)\lceil \log_2 m \rceil + 2$
P1	$4(m+2)$
P2	$2(2m+1)$
Sp	$[\mathrm{Max}\{(2w-1),(\lceil \log_2 (m-w) \rceil + 2\} + 1](2(m\text{-}w)+1)$
Ps	$[\mathrm{Max}\{(\lceil \log_2 q \rceil + 2),(\lceil 2(m-q+1) \rceil\} + 1](2q\text{-}1)$
PS1	$(2l+1+\left\lceil \log_2 (\frac{m+1}{l+1}) \right\rceil)(2\frac{m+1}{l+1}-1)$
PS2	$3(2l+1+2\left\lceil \frac{m+1}{l+1}-1 \right\rceil)$
SP	$(\lceil \log_2 m \rceil + 2 + 2\left\lceil \frac{m+1}{l+1}-1 \right\rceil)2l$

СПИСОК ЛИТЕРАТУРЫ

1. *http://en.wikipedia.org/wiki/Embedded_Supercomputing.*
2. *V. Reddi. Supercomputer Performance on a Chip Powers Next-Generation Embedded Image Processing. http://rtcmagazine.com/articles/view/102184.*
3. *О.Б. Лупанов. Курс лекций по дискретной математике. Изд-во МГУ, 2006 г., 38 стр.*
4. *А.Н. Колмогоров. Теория информации и теория алгоритмов. Москва, Наука, 1987 г., стр. 199 – 203.*
5. *Е. А. Карацуба. Быстрые алгоритмы и метод БВЕ. http://www.ccas.ru/personal/karatsuba/alg.htm#m.*
6. *Воеводин В.В. Математические модели и методы в параллельных процессах. М.: Наука, 1986.*
7. *Лукин Н.А. Основы теории проектирования архитектур функционально-ориентированных процессоров для систем реального времени // Высокопроизводительные вычислительные системы // Материалы Пятой Международной научной молодежной школы и Пятой Международной молодежной научно-технической конференции, 31 августа – 6 сентября 2008, Таганрог – Таганрог: Изд-во ТТИ ЮФУ, 2008. – с. 115 – 166.*
8. *Исследования теоретических методов и алгоритмов синтеза функциональных преобразователей. Отчет о НИР.ИМ СОАН СССР, Новосибирск, 1989.*
9. *Лукин Н.А. Синтез многофункциональных преобразователей. Оценка сложности реализации. Ракетно-космическая техника. Серия XI: Научно-технический сборник. 1989. Вып.2.*

10. *Пан В.Я. Некоторые схемы для вычисления значений полиномов с вещественными коэффициентами. Проблемы кибернетики. Вып.5. М.: Наука, 1961.*
11. *Ульман Дж.Д. Вычислительные аспекты СБИС. М.: Радио и связь, 1990.*