УДК 004.3

Инкрементные алгоритмы решения систем линейных алгебраических уравнений и архитектура мультипроцессоров на программируемой логике

Гомозов О.В., Ладыженский Ю.В.
Донецкий национальный технический университет
blackswanny@gmail.com, ly@cs.dgtu.donetsk.ua

Abstract

Gomozov O., Layzhenskyy Y. Incremental algorithms of linear equations algebraic systems solving and multiprocessor architecture based on programmable logic*. The method of solving linear equations systems by incremental algorithms is described. Simulation with different conditions is performed to see their behavior in hardware implementation of abstract model, comparisons between these algorithms and other iterative ones are made. FPGA hardware structures are proposed for implementation of these algorithms.*

Введение

Во многих научно-технических сферах, инженерных задачах, гуманитарных и социально-ориентированных науках часто используются СЛАУ (системы линейных алгебраических уравнений). Ими описываются математические модели, физические процессы, динамические системы и задачи. Получаемые системы могут включать сотни и тысячи уравнений. При этом возникает проблема скорости решения таких задач и корректности полученных результатов. Обе проблемы решаются с использованием вычислительных машин. Существует два основных направления в решении СЛАУ: прямые и итерационные методы. Прямые методы находят решение за конечное число шагов, однако сложны в реализации и не всегда устойчивы. В итерационных методах для СЛАУ вида

$$A \cdot x = b, \qquad (1)$$

где A – матрица коэффициентов, b – вектор свободных членов, x- вектор неизвестных, решение происходит путем вычисления последовательных приближений x^k при $k \to \infty$, где k - номер итерации. Условием окончания итерации является выполнение неравенства $\left\| Ax^{(k)} - b \right\| \le eps$, где *eps* – погрешность.

В данной статье исследуются свойства специальных аппаратно-ориентированных итерационных алгоритмов решения СЛАУ. Разработана архитектура мультипроцессора на программируемой логике для эффективной аппаратной реализации этих алгоритмов.

Анализ инкрементных алгоритмов

СЛАУ приводим к эквивалентной системе линейных дифференциальных уравнений:

$$\frac{dx_i}{dt} = b_i - \sum_{i,j=1}^{n} a_{ij} x_i,\ i, j = 1,2....n,\ n - порядок\ СЛАУ$$

Применяя метод Эйлера[1], получим обобщенные формулы для инкрементных алгоритмов:

$$x_i^{k+1} = x_i^k + \Delta x_i^{k+1};$$
$$\Delta x_i^{k+1} = 2^{-p} \cdot sign(\varepsilon_i^k),$$
$$\varepsilon_i^{k+1} = \varepsilon_i^k + \Delta \varepsilon_i^{k+1};$$
$$\Delta \varepsilon_i^{k+1} = -\sum_{j=1}^{n} a_{ij} \Delta x_j^{k+1},\ \ где\ x_j^0 = 0, \varepsilon_i^0 = b_i, \quad (2)$$

где a_{ij} *– коэффициент матрицы A,*
b_i *– свободный член,* x_i *– неизвестное,*
Δx_i *– приращение неизвестного,*
ε_i *– невязка,* $\Delta \varepsilon_i$ *– приращение невязки,*
i *– номер неизвестного,* k *– номер итерации,*
p *– разрядность величин*

Формулы (2) показывают, что инкрементные алгоритмы ориентированы на использование простых операций сложения, а трудоемкое умножение заменяется операцией сдвига, так как приращение Δx_i является степенью числа 2.

Невязка $\varepsilon_i(x) = b_i - \sum_{i,j=1}^{n} a_{ij} x_i$ входит в критерий сходимости [2]:

$$\sum_{i=1}^{n}(\varepsilon_i^k)^2 \le E \qquad (3)$$

Вторым критерием окончания решения может быть ограничение на максимальное количество итераций. Псевдокод алгоритма (2) представлен на рис. 1.

```
BEGIN PROGRAM
k := 0
SET p, eps, Im
FOR i := 1 TO N
    ε_i := b_i
    x_i := 0
END FOR
REPEAT
cr := 0
FOR i := 1 TO N
    Δx_i := 2^(-p) · sign(ε_i)
    Δε_i := 0
    FOR j := 1 TO N
        Δε_i := Δε_i + a_ij Δx_j
    END FOR
    ε_i := ε_i − Δε_i
    x_i := x_i + Δx_i
    cr := cr + (ε_i)^2
END FOR
    k := k + 1
UNTIL (cr < eps) OR (k > Im)
END PROGRAM
```

Рисунок 1 – Псевдокод обобщенного инкрементного алгоритма

Модификации основного алгоритма обеспечиваются различными вариантами формулы нахождения приращения неизвестного Δx_i^{k+1}. Существует 4 метода [2]:

Метод №1:

$$\Delta x_i^{k+1} = 2^{-p} \cdot sign(\varepsilon_i^k) \qquad (4)$$

Метод №2:

$$\Delta x^{k+1} = 2^{-r} \cdot sign(\varepsilon^k), \qquad (5)$$

$$\text{где} \quad r = 1,2 \ldots p-1, p$$

Метод №3:

$$\Delta x^{k+1} = 2^{-\frac{p}{S}} \cdot sign(\varepsilon^k),$$

$$\text{где } S = r, \frac{r}{2}, \frac{r}{3}, \ldots .1, \qquad (6)$$

В методе №3 каждые $2^{\frac{p}{r}}$ итерации выбирается новое S(r – количество участков, на которые разбивается разрядность p)

Метод №4:

$$\Delta x^{k+1} = 2^{-z} \cdot U_i \cdot sign(\varepsilon_i^k);$$

$$z = \min_{i=1..n} m_i; \quad 2^{-m_i} \le \varepsilon_i^k < 2^{-m_i+1};$$

$$U_i = \begin{cases} 1, \varepsilon_i^k \ge 2^{-z} \\ 0, \varepsilon_i^k < 2^{-z} \end{cases} \qquad (7)$$

Количество итераций k для получения решения с заданной точностью зависит от разрядности данных. Для исследования описанных методов создана модель работы алгоритмов на языке программирования C для нескольких видов матриц, в частности диагональ-доминирующих, трехдиагональных и матриц Гильберта[6]. В таб. 1 представлены результаты исследования сходимости методов для матрицы с доминирующей диагональю. В строках указан метод решения, в столбцах разрядность данных, в ячейках – количество итераций.

Таблица 1. Количество итераций в инкрементных методах

	Разрядность				
Метод	16	24	32	48	64
№1	65536	$>10^6$	$>10^9$	$>10^9$	$>10^9$
№2	16	24	32	48	64
№3	512	8192	65536	$>10^6$	$>10^9$
№4	64	128	256	512	1024

Как видно из табл.1, наиболее эффективными являются методы №2 и №4. Важно отметить, исходя из формул (1), возможности параллельного вычисления каждого неизвестного на каждой итерации. На рис.2 приведен график зависимости ошибки решения Eps, равной сумме модулей невязок, от номера итерации K на примере матрицы Гильберта [6] размером 1000 уравнений. На итерациях второго порядка ошибка составляет менее 0,01 [3].

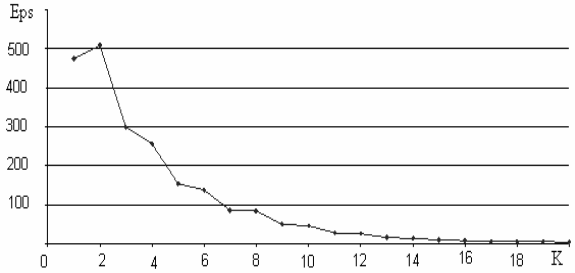

Рисунок 2 – Скорость сходимости методов

VHDL моделирование инкрементных алгоритмов

Для исследования свойств алгоритмов разработана модель на языке VHDL в среде Active-HDL. Модель обрабатывает числа с фиксированной запятой. В качестве тестовых

использовались матрицы: Dom – матрицы с доминирующими элементами на главной диагонали и Gil – плохо обусловленные матрицы Гильберта, для которых многие итерационные алгоритмы расходятся или решения являются ошибочными из-за чувствительности к погрешностям коэффициентов матрицы A. На рис. 3 - 6 представлены временные диаграммы решения системы с матрицей Dom, где: A – матрица коэффициентов, B – вектор свободных членов, Xx – вектор найденных неизвестных, k-количество итераций, N – порядок СЛАУ, P – разрядность данных, X – истинные значения неизвестных.

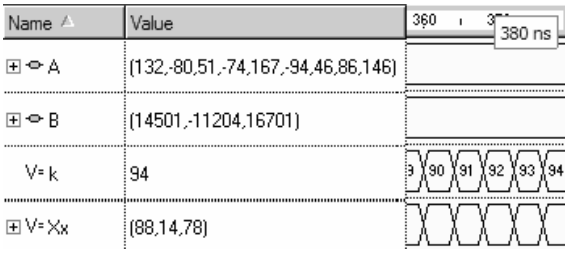

Рисунок 3 – Моделирование N=3,P=16,Dom,X=88,14,78

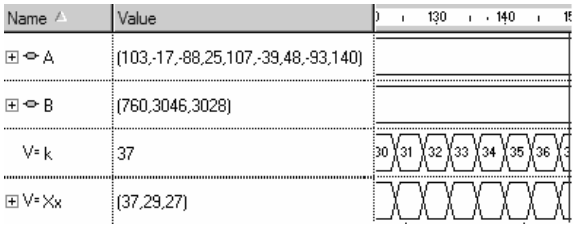

Рисунок 4 – Моделирование N=3,P=32,Dom,X=36,30,29

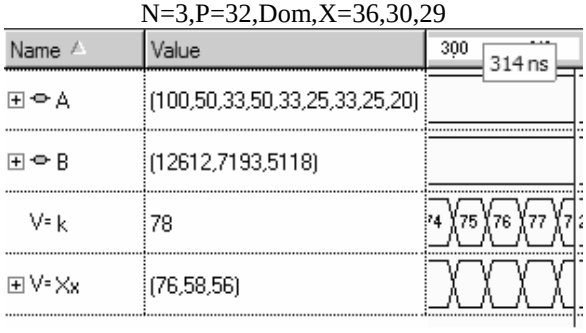

Рисунок 5 – Моделирование N=3,P=48,Dom,X=77,58,56

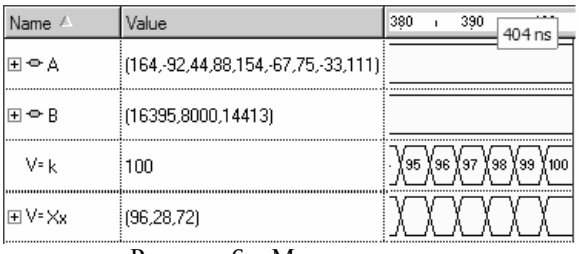

Рисунок 6 – Моделирование N=3,P=64,Dom,X=96,28,72

Как видно из рис. 3-6 VHDL-модель решает СЛАУ быстро и с погрешностью, не превышающей младшего разряда, в том числе при малой разрядности данных. Однако в случае с матрицей Гильберта возрастает погрешность вычисления, связанная с ошибкой округления входных данных. Для противодействия этому эффекту возможно увеличение разрядности данных, результаты представлены на рис. 7 и рис.8.

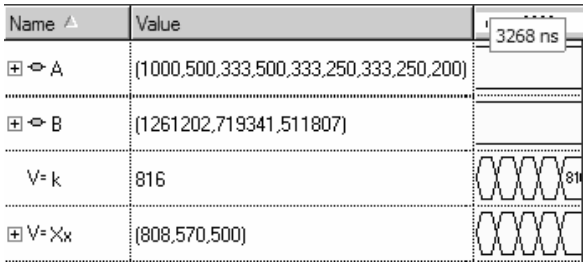

Рисунок 7 – Моделирование N=3,P=32,Gil,X = 809,585,480

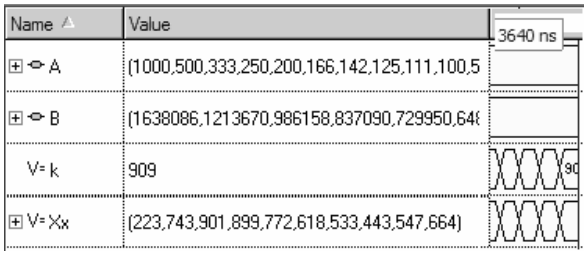

Рисунок 8 – Моделирование N=10,P=64,Gil,X = 210,780,894,897,754,611,422,466,567,840

Модификации (5)-(7) алгоритма способны уменьшить количество итераций, и увеличить скорость сходимости алгоритма [3] .

Архитектура мультипроцессора для реализации инкрементных методов

Для аппаратной реализации вышеприведенных алгоритмов можно использовать интегральные схемы FPGA[4], выпускаемые фирмами Xilinx, Altera, Lattice и др. В состав FPGA входят отдельные программируемые блоки памяти и шины данных. Такая структура дает возможность сконфигурировать и запрограммировать микросхемы, наиболее эффективно используя достоинства инкрементных алгоритмов, в частности, распараллеливание вычислений [5]. Реализация на FPGA является дешевым, расширяемым и масштабируемым решением. Микросхемы FPGA позволяют создавать многомодульные и многоядерные вычислительные системы всего на одном кристалле, сконфигурированные под целевые задачи.

Обобщенная функциональная схема процессора, реализующего инкрементные методы, представлена на рис.9. Блоки 1,2,3 и 4 –

запоминающие устройства для хранения матрицы A, вектора невязок E, вектора неизвестных X и вектора приращений неизвестных dX соответственно. В начале работы в блоки 1 и 2 загружаются матрица A и вектор свободных членов b. Блоки 5 и 8 реализуют адресацию строки в памяти ЗУ для текущего i-го уравнения. Аналогично реализуют адресацию блоки 11 и 13 для индекса столбца j матрица A.

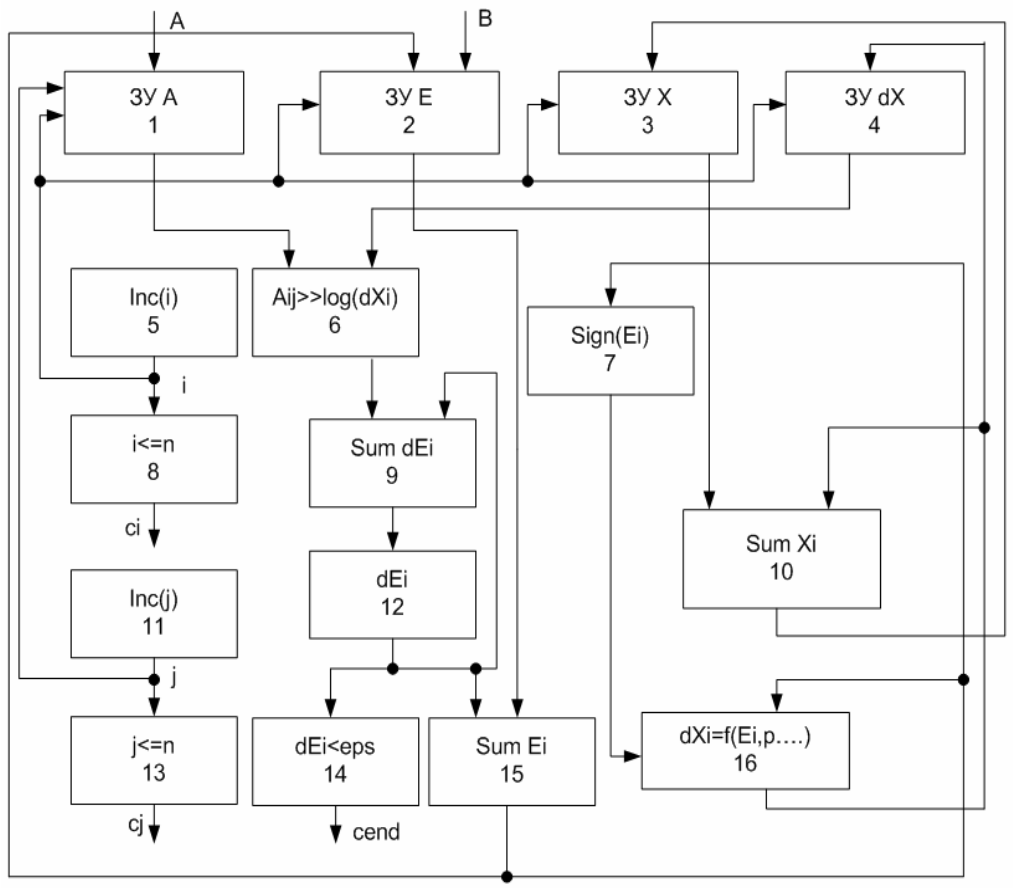

Рисунок 9 – Обобщенная структура процессора

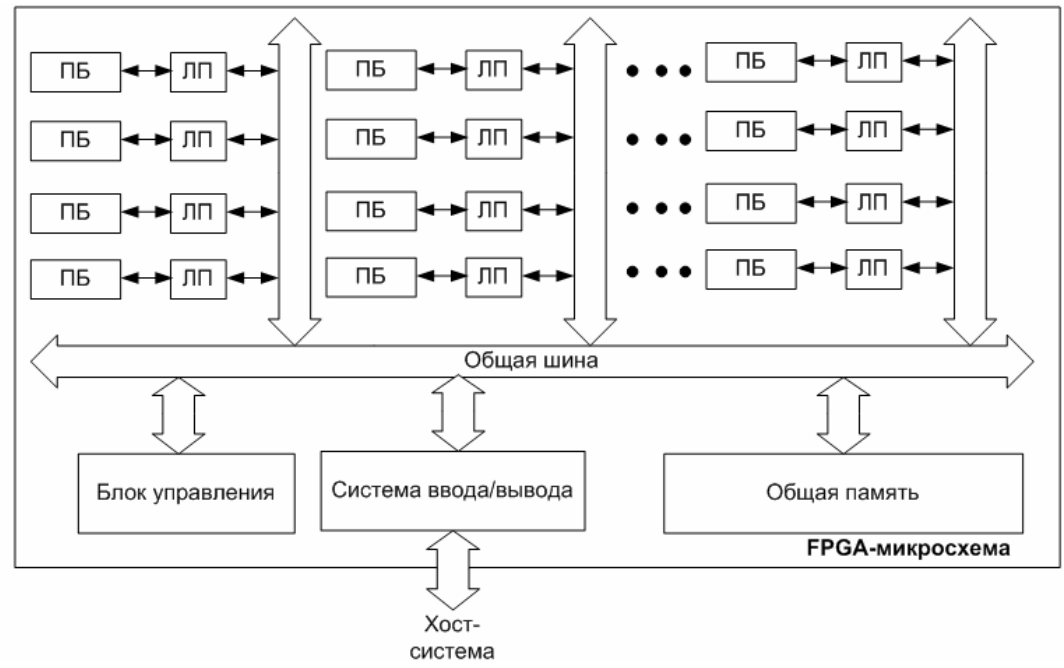

Рисунок 10 – Структура мультипроцессора

Блок 6 производит операцию сдвига коэффициента $a_{i,j}$ на количество разрядов, равное $\log_2 dX_i$, что равнозначно операции $a_{i,j} * dX_i$.

Блоки 9 и 12 – это сумматор и аккумулирующий регистр для вычисления значения приращения невязки dE_i. В блоке 14 проверяется условие конца решения, т.е. условие сходимости невязок к некоторой величине погрешности eps. В блоке 15 происходит вычисление текущей невязки по формуле $E_i = E_i - dE_i$.

Знак невязки, полученный в блоке 7, используется в блоке 16 для выполнения одного из алгоритмов нахождения приращения неизвестного, в каждом из которых участвует операция сдвига невязки. Последняя операция нахождения неизвестного происходит в блоке 10 по формуле $x_i = x_i + dx_i$.

В данной структуре самыми ресурсоемкими являются блоки суммирования, а все операции умножения заменены простыми и быстрыми операциями сдвига. Также возможны другие модификации, например, использование одного сумматора, мультиплексора данных и группы накапливающих регистров вместо трех сумматоров, приведенных на рис. 9.

Процессоры, описанные на рис. 9, могут объединяться в многопроцессорные схемы, как показано на рис. 10. Каждый из них представляет собой процессорный блок (ПБ) со своей локальной памятью (ЛП), работающий параллельно с остальными. Перед началом работы в общую память загружаются исходные данные (матрица коэффициентов, вектор свободных членов, параметры) через систему ввода/вывода, связывающую устройство с внешней хост-системой. Блок управления проверяет условие решения после каждой итерации, посылает управляющие сигналы. Для быстрого доступа локальная память процессорного блока кеширует данные, которые загружаются в неё из общей памяти. Каждый ПБ может произвести итерацию для одного уравнения СЛАУ. Если количество ПБ меньше количества уравнений, блок управления формирует очередь (конвейер) этих уравнений, отдавая данные свободным ПБ до тех пор, пока в текущей итерации не будут вычислены все. При этом скорость решения будет уменьшаться пропорционально отношению количества уравнений к количеству ПБ.

Рассматриваемые мультипроцессорные схемы могут быть реализованы на FPGA, так как они позволяют создание таких архитектур. При параллельной реализации условное количество вентилей на один процессорный блок с локальной памятью равно:

$$C = 3У_A + 3У_E + 3У_X + 3У_{dX} + 3 * SUM + \\ + 2 * SREG + 2 * CS \qquad (8)$$

,где $3У_A, 3У_E, 3У_X$ *и* $3У_{dX}$ - запоминающие устройства (1 вентиль на бит) соответствующих данных, SUM – сумматор (9 вентилей на бит), SREG – сдвиговый регистр (3 вентиля на бит), CS – схема сравнения (2 вентиля на бит). При разрядности данных M=32 бита и размере СЛАУ N=100 уравнений

$$C = 100 * 32 + 32 + 32 + 32 + 3 * 32 * 9 + \\ + 2 * 32 * 3 + 2 * 2 * 32 = 1600$$

из которых 1184 вентиля используются для процессорного блока и 416 вентилей - для локальной памяти.

Реализация блока приращения 16 зависит от выбранного инкрементного алгоритма. Возможны следующие схемы, представленные на рис. 11-14:

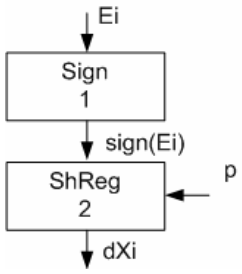

Рисунок 11 – Блок приращения в алгоритме 1

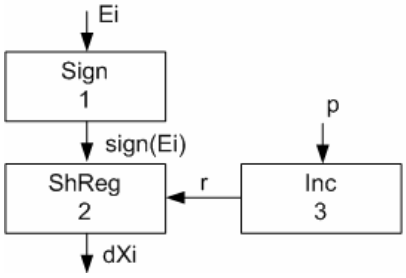

Рисунок 12 – Блок приращения в алгоритме 2

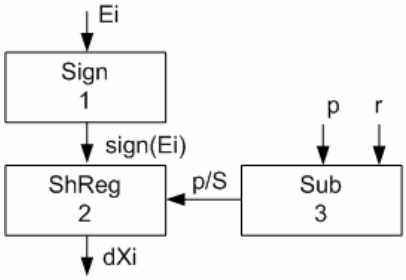

Рисунок 13 – Блок приращения в алгоритме 3

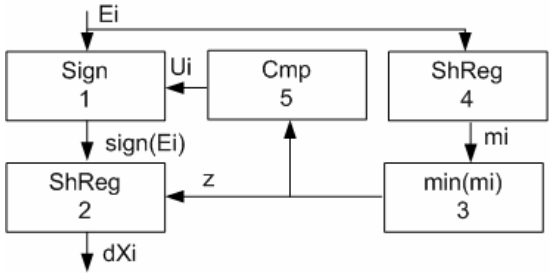

Рисунок 14 – Блок приращения в алгоритме 4

Блоки 1 и 2 на рис. 11-14 одинаковы для всех алгоритмов и реализуют соответственно схему анализа знака и сдвиговый регистр. Блок 3 на рис. 12 является инкрементором для нахождения r, а на рис. 13 вычитателем для нахождения величины p/S. На рис.14 блоки 3 (схема сравнения) и 4 (сдвиговый регистр) вычисляют значение z, необходимое для схемы сравнения (блок 5).

Выводы

Проведены исследования итерационных инкрементных алгоритмов решения СЛАУ, изучены их свойства и особенности. С применением моделирования получены различные характеристики алгоритмов: скорость сходимости, устойчивость к видам и разрядности данных. Предложены варианты аппаратной реализации алгоритмов на FPGA с использованием параллельных вычислений.

Показано, что инкрементные методы обладают важными для практического применения свойствами:
- высокая скорость решения задач и простота арифметических операций;
- устойчивость и отсутствие ограничений для многих классов задач;
- возможность эффективной реализации на базе микросхем FPGA.

Литература

1. Малиновский Б.Н., Боюн В.П., Козлов Л.Г. Алгоритмы решения систем линейных алгебраических уравнений, ориентированные на структурную реализацию. – “Управляющие системы и машины”. Вып. №5. 1977, с.79-84.
2. Боюн В.П., Козлов Л.Г., Малиновский Б.Н., Третьяков С.И. Устройства для решения систем линейных алгебраических уравнений. Автор. свид. № 543943 – БИ, 1977, № 3.
3. Куркчи В.А., Гомозов О.В., Ладыженский Ю.В. Исследование инкрементных методов решения СЛАУ. – Донецк, ДонНТУ: Всеукраинская научно-техническая конференция “КМиТ”, 2009, с. 190-193.
4. Hongyan Yang, Sotirios G. Ziavras. FPGA-based Vector Processor for Algebraic Equation Solvers. http://web.njit.edu/~ziavras/Wmatrix.pdf
5. Максфилд К. Проектирование на ПЛИС. – М.: ”Додека-XXI”, 2007, 408с.
6. Ланкастер П. Теория матриц. – М.:”Наука”, 1973, 282с.

Приложение. Матрицы для экспериментов

Для исследования свойств инкрементных методов и проведения моделирования были выбраны виды матриц, имеющие свои особенности и область использования, покрывающие наиболее известные классы задач по различным критериям.

Диагональ-доминирующая матрица подчиняется условию:

$$a_{i,i} \geq \sum_{j=1}^{n} \left| a_{i,j} \right|, \text{где } i \neq j; i, j = 1..n \quad (8)$$

т.е. элемент главной диагонали больше либо равен сумме модулей остальных элементов строки. Такая матрица часто встречается, и известные итерационные методы для нее сходятся.

В сеточных задачах математической физики, при расчете электрических цепей и во многих областях используются ленточные матрицы. Частным случаем ленточной является ***трехдиагональная*** матрица. У неё ненулевые элементы содержатся только на главной диагонали, а также смежными с ней нижней и верхней диагоналями, т.е.:

$$a_{i,i} \geq 0, a_{i,j-1} \geq 0, a_{i,j+1} \geq 0, \text{где } i, j = 1..n \quad (9)$$

Наибольший интерес представляет ***матрица Гильберта***. Она является плохо обусловленной, почти вырожденной матрицей, т.е. её определитель близок к нулю. Один из примеров матрицы Гильберта определяется следующей формулой:

$$a_{i,i} = \frac{1}{i + j - 1}, \text{где } i, j = 1..n \quad (10)$$

Такие СЛАУ чувствительные к точности коэффициентов матрицы, что показывает число обусловленности. Из-за этого при решении многими методами возникают ложные корни, что особенно актуально при ошибках округления, возникающих в вычислительных устройствах.

Поступила в редакцию 15.03.10