ДИСТРИБУТИВНЫЕ МЕТОДЫ ФОРМИРОВАНИЯ ПАРАЛЛЕЛЬНЫХ МОДЕЛЕЙ ДИНАМИЧЕСКИХ СИСТЕМ

Святный В.А, Баженов Л.А.
Донецкий Государственный Технический Университет, кафедра ЭВМ.
tel:35-45-89
e-mail:lechat@dstu.donetsk.ua

Abstract

Svjatnyj V., Bazhenov L. The methods of parallel model creation. *The problem of simulation of dynamic systems is very important today. This paper describes methods of simulation of dynamic systems using SIMD-architecture. It is assumed that a model is coded by means of the table of connections (which can be created by means of some visual tool). The task is to simulate the model using SIMD computer.*

Введение

На сегодняшний день актуальной является проблема моделирования динамических систем (ДС) с использованием параллельных компьютеров [1]. В статье обсуждаются методы моделирования динамических систем с использованием супер-ЭВМ SIMD архитектуры. При обсуждении методов будем полагать, что модель кодируется матрицей соединений (которая может быть сформирована, или визуальными средствами, или же, как результат трансляции script-кода описания модели). Задача состоит в формировании параллельной модели динамической системы с сосредоточенными параметрами (ДССП) и моделирования на SIMD компьютере.

1. Выделение множества модельных объектов

ДССП описывается обыкновенным дифференциальным уравнением **n-ого** порядка или системой **n** обыкновенных дифференциальных уравнений первого порядка, выраженных относительно производной:

$$\begin{cases} \frac{dy_1}{dt} = a_{11}y_1 + a_{12}y_2 + \ldots + a_{1,n}y_n + b_1 f_1; \\ \frac{dy_2}{dt} = a_{21}y_1 + a_{22}y_2 + \ldots + a_{2,n}y_n + b_2 f_2; \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \frac{dy_n}{dt} = a_{11}y_1 + a_{n2}y_2 + \ldots + a_{n,n}y_n + b_n f_n; \end{cases} \qquad (1)$$

По виду системы (1) можно выделить минимальное количество модельных объектов, которое необходимо для моделирования ДССП (рис.1) [5].

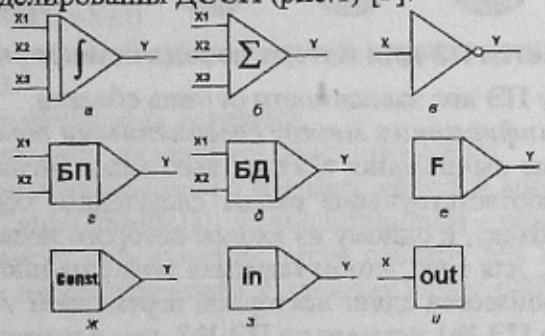

Рис.1. Минимально возможное множество модельных объектов.
а) интегро-сумматор; б) сумматор; в) инвертор; г) блок произведения; д) блок деления; е) генератор заданной функции; ж) генератор константы; з) точка входа модели; и) точка выхода модели.

Блок интегро-сумматор реализует следующую формулу: $Y=\int(\alpha_1X_1+\alpha_2X_2+\alpha_3X_3)dx+Y_0$. Сумматор предназначен для реализации формулы: $Y=\beta_1X_1+\beta_2X_2+\beta_3X_3$. Коэффициенты α_i и β_j являются действительными числами. Инвертор меняет знак входного сигнала на противоположный. Блок произведения работает по формуле: $Y=X_1 \bullet X_2$; а блок деления - $Y=X_1/X_2$. Блок "F" генерирует заданную функцию, определенную формулой или таблицей. Блок "Const" выдает в течении всего времени моделирования одно и тоже значение, и не является необходимым блоком как выше приведенные, так как может быть реализован генератором функций. Блок "IN" - точка входа модели. Блок "OUT" - точка выхода модели. Также этот блок может быть использован в любой точке модели для определения сигналов в этих точках.

2. Описание дистрибутивных методов для моделирования динамических систем с сосредоточенными параметрами с использованием SIMD систем

Пусть формальное описание модели представлено таблицей, имеющей вид:

№ объекта	Тип объекта	Вход X_1	Вход X_2	...	Вход X_n
...	...	...	...	...	...

где *№ объекта* - порядковый номер объекта в модели;

тип объекта - идентификатор объекта;

вход X_i - порядковый номер объекта, который соединен с i-ым входом объекта в строке.

Основная идея обсуждаемых дистрибутивных методов состоит в распределении объектов модели по сетке процессорных элементов. Каждый ПЭ реализует свою функцию сопоставленного ему объекта. Назовем шагом моделирования расчет выходных сигналов всех объектов модели при полной передачи данных между ними за текущий отрезок времени. Количество шагов моделирования равно количеству отрезков времени, на которое делится все время моделирования. На каждом этапе на всех ПЭ, которым сопоставлены объекты одного типа, параллельно производится счет по одной и той же формуле в силу структуры SIMD системы. Следовательно, количество этапов равно количеству типов объектов. После завершения счета для всех ПЭ на каждом из них в результате будут храниться выходные значения соответствующих объектов. Полученные выходные значения должны быть переданы на входы объектов, с которыми соединен рассматриваемый объект. После того, как все данные переданы соответствующим процессорным элементам, текущий шаг моделирования заканчивается.

2.1. Метод передачи информации по цепочке

Выбор топологии сетки ПЭ. На рис.2 приведена структура связей ПЭ.

Распределение объектов модели на сетке ПЭ. Каждый объект модели последователь-

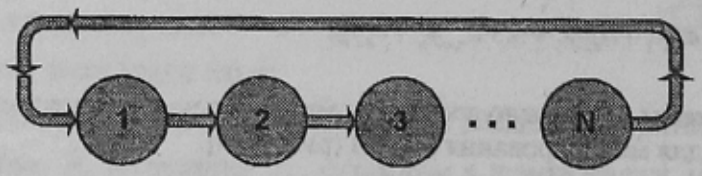

Рис.2. Топология сетки ПЭ для метода передачи информации цепочкой.

но назначается отдельному ПЭ вне зависимости от типа объекта.

Способ передачи информации между соединенными объектами модели. На текущем шаге моделирования, после вычисления вектора выходных сигналов, информация с выходов объектов передается на соответствующие входы следующим образом. Пусть, например, на ПЭ №3 (рис.2) загружен объект, к одному из входов которого подключен выход объекта, загруженного на ПЭ №1. Тогда, для того, чтобы передать информацию из ПЭ №1 в ПЭ №3 необходимо, в данном случае, произвести сдвиг векторной переменной vY вправо дважды. После таких сдвигов элемент vY на ПЭ №1 попадет на ПЭ №3, что и позволит произвести присваивание векторной переменной хранящей входные значения объектов, значение вектора vY. Таким образом, после распределения объектов модели, необходимо выполнить расчет количества сдвигов для каждого ПЭ, необходимых для передачи информации с ПЭ источника.

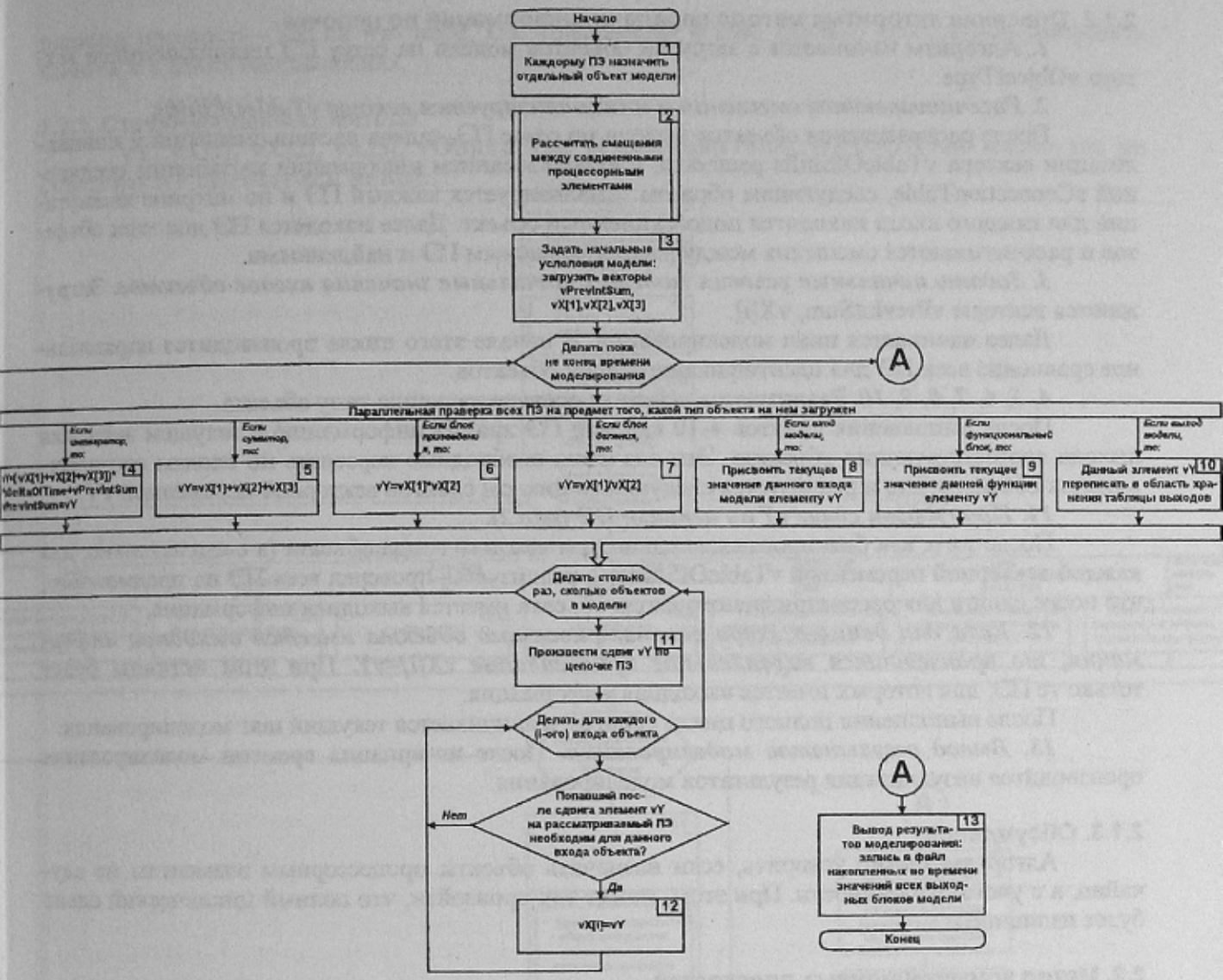

Рис.3. Блок-схема алгоритма метода передачи информации цепочкой.

2.1.1. Структура данных метода.

vY - вектор выходных сигналов. Каждому ПЭ назначен отдельный объект. Выходное значение объекта на каждом шаге моделирования передается в соответствующий элемент вектора vY.

vX[1], vX[2], vX[3] - три вектора входных сигналов, соответствующие входам X1, X2 и X3 объектов. Если объект имеет менее трех входов, то соответствующий элемент вектора не используется. Для вычисления выхода каждый модельный объект принимает входную информацию из векторов **vX[1], vX[2], vX[3]**.

vObjectType - вектор типов объектов. Каждый элемент этого вектора хранит тип объекта присвоенный данному ПЭ.

vPrevIntSum - вектор начальных значений.

sConnectionTable - матрица соединений объектов модели.

sDeltaOfTime - шаг моделирования по времени; является константой.

vTableOfShifts[1], vTableOfShifts[2], vTableOfShifts[3] - таблицы смещений для каждого входа объекта; предназначена для хранения числа сдвигов для данного ПЭ после которого выходное значение объекта, подсоединенного к данному входу рассматриваемого объекта, попадет на данный ПЭ.

На рис.3 приведена блок-схема алгоритма метода, в котором используется передача информации вдоль цепочки ПЭ. Рассмотрим метод более подробно.

2.1.2. Описание алгоритма метода передачи информации по цепочке

1. Алгоритм начинается с загрузки объектов модели на сетку ПЭ инициализацией вектора vObjectType.

2. Рассчитываются смещения и инициализируется вектор vTableOfShifts.

После распределения объектов модели по сетке ПЭ, задача расчета смещений и инициализации вектора vTableOfShifts решается, с использованием информации из таблицы соединений sConnectionTable, следующим образом. Анализируется каждый ПЭ и по матрице коммутаций для каждого входа находится подсоединенный объект. Далее находятся ПЭ для этих объектов и рассчитываются смещения между рассматриваемым ПЭ и найденными.

3. Задать начальные условия модели и начальные значения входов объектов. Загружаются векторы vPrevIntSum, vX[i].

Далее начинается цикл моделирования. В начале этого цикла производится параллельное сравнение всех ПЭ для идентификации типов объектов.

4, 5, 6, 7, 8, 9, 10. Реализуются действия соответствующие типу объекта.

После выполнения пунктов 4-10 каждый ПЭ хранит информацию о текущем значении выхода соответствующих объектов. Эти значения необходимо переслать по входам соответствующих объектов, что и реализуется следующим циклом сдвигов векторной переменной vY.

11. Произвести сдвиг vY по цепочке ПЭ (рис.2).

После того, как был произведен сдвиг, для каждого входа объекта (а следовательно, для каждой векторной переменной vTableOfShifts[i], реализуется проверка всех ПЭ на предмет того, что после сдвига для рассматриваемого входа объекта имеется выходная информация.

12. Если для данного входа рассматриваемого объекта имеется выходная информация, то производится параллельное присваивание vX[i]=vY. При этом активны будут только те ПЭ, для которых имеется выходная информация.

После выполнения полного цикла сдвигов заканчивается текущий шаг моделирования.

13. Вывод результатов моделирования. После исчерпания времени моделирования производится визуализация результатов моделирования.

2.1.3. Обсуждение

Алгоритм можно ускорить, если назначать объекты процессорным элементам не случайно, а с учетом связанности. При этом, может так произойти, что полный циклический сдвиг будет излишним.

2.2. Метод коммутационных плоскостей

Выбор топологии сетки ПЭ. В методе каждый ПЭ связан с четырьмя соседями (рис. 4), исключая граничные элементы.

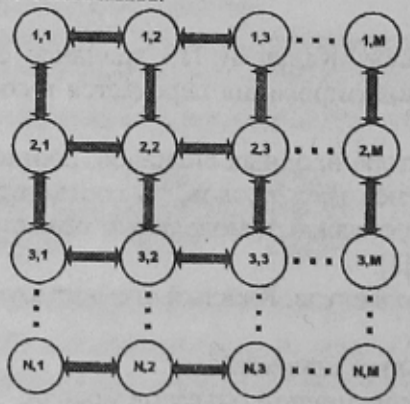

Рис. 4. Топология сетки ПЭ для метода коммутационных плоскостей.

Распределение объектов модели на сетке ПЭ. Каждый объект модели последовательно назначается каждому ПЭ вне зависимости от типа объекта.

Способ передачи информации между соединенными объектами модели. Для каждого соединения создается коммутационная плоскость, на которой существует канал связи между двумя объектами модели. Коммутационных плоскостей столько, сколько пар объектов связаны между собой в модели. Пусть, например на ПЭ №1,1 находится объект модели, генерирующий информацию для объекта, загруженного на ПЭ №2,3 (рис. 4). Тогда, после просчета выходных сигналов (vY), векторную переменную vY можно сначала сдвинуть вниз один раз, а затем вправо два раза. Таким образом, информация от ПЭ №1,1 попадет на ПЭ №2,3. Итак, коммута-

ционная плоскость - это ни что иное, как информация о том, сколько необходимо совершить сдвигов и в каких направлениях.

2.2.1. Структура данных метода

vY, vX[1], vX[2], vX[3], vObjectType, sConnectionTable, sDeltaOfTime имеют тот же смысл как и в предыдущем методе.

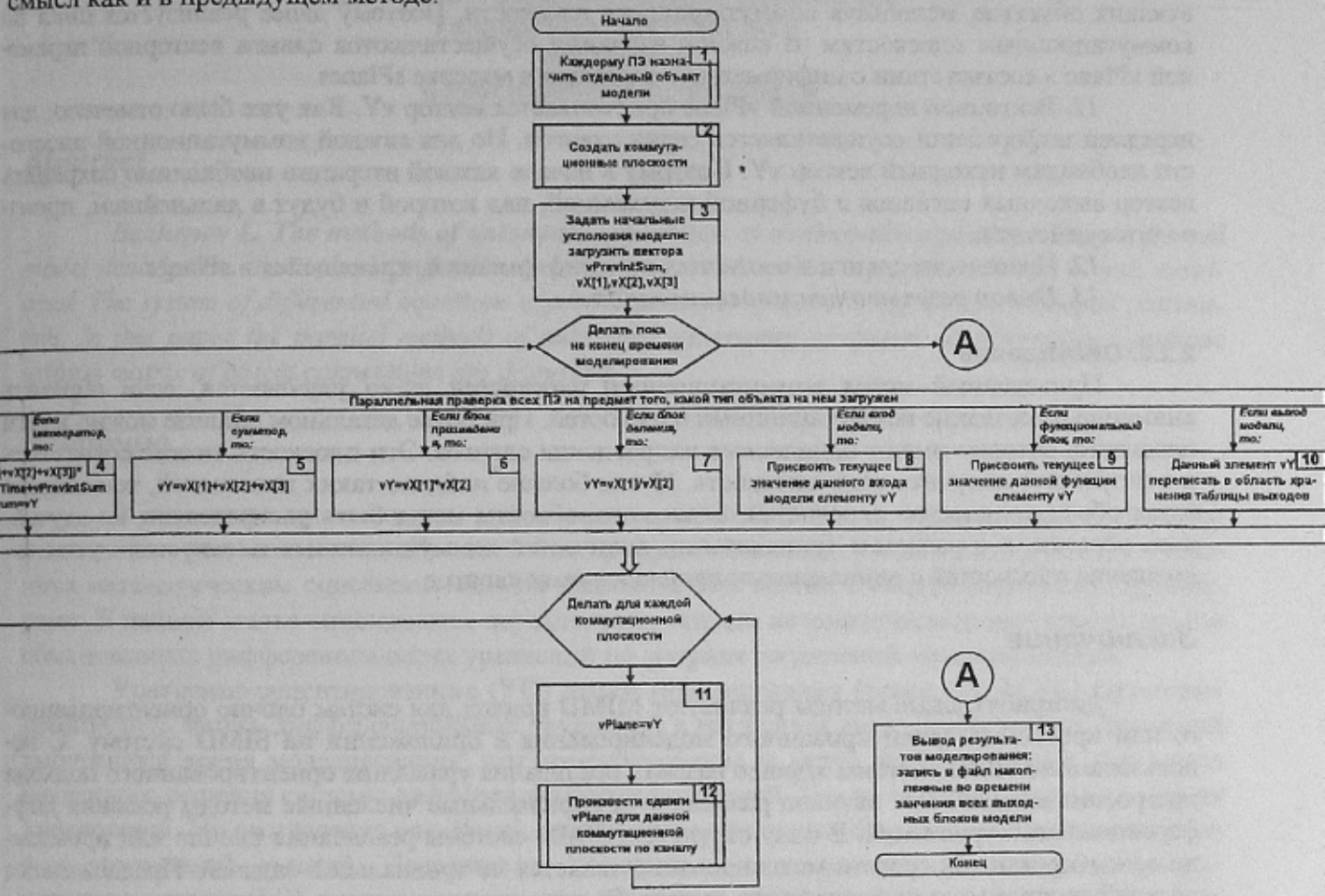

Рис. 5. Блок-схема алгоритма метода коммутационных плоскостей.

sPlanes[i][j] - массив коммутационных плоскостей. Максимальное значение индекса i равно количеству связей пар объектов в модели (количество коммутационных плоскостей). Вторая размерность массива ответственна за хранение информации о сдвигах. При j=1 хранится номер ПЭ источника (кроме двумерной нумерации каждый ПЭ имеет свой порядковый номер); j=2 - номер ПЭ приемника; j=3 - номер входа приемника; j=4 - если не нуль, то код первого сдвига; j=5 - если не нуль, то код второго сдвига. Код сдвига - это код направления: вверх, вниз, влево или вправо. Если код сдвига равен нулю, то сдвиг не осуществляется.

vPlane - векторная переменная, хранящая вектор vY для данной коммутационной плоскости.

На рис. 5 приведена блок-схема алгоритма метода коммутационных плоскостей. Рассмотрим метод более подробно.

2.2.2. Описание алгоритма метода коммутационных плоскостей

1. Алгоритм начинается с загрузки объектов модели на сетку ПЭ.

*2. **Создаются коммутационные плоскости.*** Выше была подробно описана структура коммутационных плоскостей. Поэтому процедуру создания коммутационных плоскостей можно определить следующим образом. После загрузки всех ПЭ объектами модели (блок 1), необходимо просматривать матрицу коммутаций sConnectionTable и для каждого входа текущего объекта находить объект источник выходной информации. Для найденной таким образом пары объектов находятся ПЭ. Далее заполняется массив sPlanes.

*3. **Задать начальные условия модели и начальные значения входов объектов.*** Загружаются вектора vPrevIntSum, vX[i].

Далее начинается цикл моделирования. В начале этого цикла производится параллельное сравнение всех ПЭ на предмет того, какой тип объекта на нем загружен.

4, 5, 6, 7, 8, 9, 10. Реализуются действия соответствующие типу объекта.

После выполнения пунктов 4-10 каждый ПЭ хранит информацию о текущем значении выхода соответствующих объектов. Эти значения необходимо переслать на входы соответствующих объектов, используя коммутационные плоскости. Поэтому далее реализуется цикл по коммутационным плоскостям. В каждой итерации осуществляются сдвиги векторной переменной vPlane в соответствии с информацией, заданной в массиве sPlanes.

11. Векторной переменной vPlane присваивается вектор vY. Как уже было отмечено, для передачи информации осуществляется серия сдвигов. Но для каждой коммутационной плоскости необходим исходный вектор vY. Поэтому в начале каждой итерации необходимо сохранять вектор выходных сигналов в буферной переменной, над которой и будут в дальнейшем, производится действия.

12. Произвести сдвиги в соответствии с информацией, хранящейся в sPlanes.

13. Вывод результатов моделирования.

2.2.3. Обсуждение

Приведенный метод коммутационных плоскостей легко улучшается, если обратить внимание на создание коммутационных плоскостей. При более детальном анализе можно найти плоскости, которые имеют одинаковые направления сдвигов. Эти плоскости можно совместить и в результате получить одну плоскость. И чем больше найдено таких плоскостей, тем меньше будет общее количество плоскостей. Более того, объекты могут быть распределены не случайным образом, а с расчетом минимизации количества коммутационных плоскостей путем совмещения плоскостей с одинаковыми направлениями сдвигов.

Заключение

Дистрибутивные методы реализуют MIMD подход для систем блочно ориентированного или проблемно ориентированного моделирования в приложении на SIMD систему. С использованием SIMD системы хорошо развита реализация уравнение ориентированного подхода построения моделей, т.к. активно развиваются параллельные численные методы решения дифференциальных уравнений. В силу структуры SIMD системы реализация блочно или проблемно ориентированных средств моделирования является не тривиальной задачей. Предложенные дистрибутивные методы решают эту задачу. Ее актуальность связана в актуальности реализации блочно ориентированных и проблемно ориентированных средств моделирования на компьютерах SIMD архитектуры. Сейчас ведутся исследования в направлении сравнения эффективности использования численных методов и дистрибутивных методов в моделировании динамических систем.

Литература

1. Anoprijenko A., Bräunl T., Reuter A, Svjatnyj V., Zeitz M. *Massiv parallele Simulationsumgebung für dynamische Systeme mit konzentrierten und verteilten Parametern.* In G.Kampe, M.Zeitz (Hrsg):Tagungsband 9. Symposium Simulationstechnik ASIM'94 in Stuttgart, Verlag Vieweg 1994.
2. Bräunl T. *Parallel Programming.* Prentice-Hall, 1993.
3. Fischer G. *Beyond Human-Computer Interaction.* Mensch Computer Kommunication. 1993.
4. Schneider-Hufschmidt M. *Eine Entwicklungsumgebung fur adaptierbare Benutzungsoberflachen.* Mensch Computer Kommunication. 1993.
5. Святный В. *Гибридные вычислительные системы.* Киев. "Вища школа".1980.