МЕТОДЫ АВТОМАТИЧЕСКОЙ ГЕНЕРАЦИИ МАТЕМАТИЧЕСКИХ УРАВНЕНИЙ ПО ФОРМАЛЬНОМУ ОПИСАНИЮ МОДЕЛЕЙ ДИНАМИЧЕСКИХ СИСТЕМ

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

Abstract

Bazhenov L. The methods of automatic generation of mathematic equations using a formal model description. *The calculating methods of system of differential equations solving is well developed. The system of differential equations expresses a model of dynamic system with lumped parameters. In this paper the parallel methods of automatic generating of system of differential equations using a matrix of object connections are discussed.*

Введение

В настоящее время хорошо развит математический аппарат числительных параллельных методов решения систем обыкновенных дифференциальных уравнений [1], которые являются математическим описанием моделей динамических систем с сосредоточенными параметрами. В данной статье описываются параллельные методы автоматического построения систем обыкновенных дифференциальных уравнений по матрице соединений объектов модели.

Уравнение-ориентированные (УО) языки моделирования (такие как ACSL) позволяют производить запись дифференциальных уравнений в наглядном виде. После записи уравнений выбираются время моделирования, шаг интегрирования и метод интегрирования или метод численного решения системы дифференциальных уравнений. Так же может задаваться и другая информация. Но составление уравнений для объекта моделирования реальной сложности является трудоемкой задачей. Параллельно УО-языкам моделирования развиваются блочно-ориентированные (БО) языки моделирования (такие как ASRSIM). Созданы БО-языки для конкретных предметных областей, в которых необходимо производить моделирование, называемые проблемно или объектно-ориентированные (ОО). Реальные объекты данной предметной области являются прототипами модельных объектов используемых в БО-языке. Таким образом, эксперт предметной области может строить модель, используя знакомые ему объекты. Но для проведения модельных экспериментов необходимо либо составить уравнения по построенной модели, либо использовать БО или ОО средства моделирования.

В статье обсуждается проблема генерации математических уравнений по модели, построенной с помощью БО или ОО языка моделирования.

1. Основная идея метода

Пусть предметная область динамических систем выражена объектами [5], каждый из которых можно выразить как множество соединенных интеграционных объектов, сумматоров, объектов произведения и деления, объектов генерации функции и константы [4]. Назовем объекты генерации информации и объекты, не имеющие выходы, терминальными объектами. Объектами генерации информации в БО системе моделирования могут быть функциональные объекты (объекты генерации заданной функции), объекты генерации заданных констант, объекты, являющиеся входами модели. Остальные объекты будем называть нетерминальными объектами. Данные определения говорят о том, что нетерминальные объекты могут быть заменены функциональной зависимостью, аргументами которой являются другие нетерминальные и терминальные объекты. Аргументами этой функциональной зависимости служат объекты, выходы

которых подсоединены на вход рассматриваемого нетерминального объекта. Пусть систему ОДУ необходимо сформировать в виде, приведенном в работе [5] (система уравнений (1)). Причем, коэффициенты a_{ij} и b_i могут быть функциями от времени t или же выражениями, состоящими из таких функций. Из каждого уравнения системы видно, что на вход интегрального объекта подается результат вычисления правой части, а на его выходе формируется значение y_i. Таким образом, интегральные объекты - это нетерминальные объекты, с которых необходимо начинать строить уравнения. Но на этих объектах также и заканчивается генерирование данной ветви уравнения. Количество уравнений равно количеству интегральных объектов в модели. Если бы автоматическое формирование уравнений производилось с использованием последовательного алгоритма, то он выглядел бы следующим образом. Каждый интегральный объект трансформируется в левые части уравнений системы. Правые части строятся по следующему правилу. Последовательно просматриваются входы каждого интегрального объекта. Все входы интегрального объекта суммируются. Поэтому необходимо произвести подстановки для аргументов суммы. В зависимости от объекта соединенного на рассматриваемый вход, в качестве аргумента суммы подставляется его функция с аргументами, зависящими от соединенных на его вход объектов. Данная ветвь подстановок заканчивается по достижению терминального объекта.

Для супер-ЭВМ SIMD архитектуры параллельный алгоритм автоматического формирования системы ОДУ может выглядеть следующим образом. Таблица соединений объектов построчно загружается на сетку процессорных элементов (ПЭ). Таким образом, каждый ПЭ содержит информацию как о загруженном на нем объекте, так и об объектах, выходы которых подсоединены на вход данного объекта. Т.к. формирование каждого уравнения системы необходимо начинать с интегрального объекта, то левые части уравнений можно считать сформированными и находящимися на ПЭ, хранящих интегральные объекты. Левые части будут формироваться на этих же ПЭ. На первом шаге цикла генерации уравнений производится активизация всех ПЭ, на которых производится формирование уравнений. В начале алгоритма - это ПЭ, хранящие интегральные объекты. Имея информацию о номерах объектов в модели, соединенных на входы интегральных, можно получить и всю информацию об объекте. Но она расположена на другом ПЭ. Для получения информации об объектах соединенных на входы интегральных необходимо производить сдвиги информации по сетке ПЭ и проверять, не поступила ли нужная информация на ПЭ, на котором формируются уравнения. После поступления необходимой информации недостающие места правых частей уравнений дополняются операциями, соответствующими типам объектов. Если на входы этих объектов подаются нетерминальные объекты, то весь процесс повторяется, иначе рассматриваемая ветвь уравнения заканчивается, т.е. достигнут терминальный символ уравнения. Таким образом, цикл генерации уравнений состоит из сдвигов информации по сетке ПЭ и формирования недостающих частей уравнений. При этом сами объекты становятся операциями в уравнениях, а их входы - операндами. Для функционального объекта формируется указание на функцию, для объекта, являющегося входом модели и интегрального объекта - переменная (терминальный символ). ПЭ, которые в начале алгоритма содержали интегральные объекты, при окончании алгоритма будут хранить закодированные уравнения. Цикл генерации уравнений для каждого ПЭ, хранящего формируемые уравнения, заканчивается, когда достигнуты все терминальные символы или же переменные, характеризующие интегральные объекты.

2. Кросс-метод генерации систем ОДУ

2.1. Топология сетки процессорных элементов для метода

Топология сетки процессорных элементов для данного метода представлена в работе [5] и аналогична топологии сетки ПЭ в дистрибутивном методе коммутационных плоскостей. Таблица соединений объектов загружается на сетку ПЭ не одной цепочкой, а дублируется столько раз, сколько максимальное число входов у объектов модельной предметной области. При фор-

мировании уравнений с добавлением операций и операндов происходит и копирование цепочек.

2.2. Структура данных метода

Будем полагать, что максимальное число входов объектов равно, для определенности, трем. Для изложения предлагаемого метода это не имеет значения. В рассматриваемом методе используются следующие векторные и скалярные переменные. Для удобства будем имена векторных переменных предварять префиксом **v**, а скалярных переменных - префиксом **s**.

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

vObjectID - вектор модельных номеров (ID) объектов.

sConnectionTable - таблица соединений объектов модели, которая необходима лишь для начальной загрузки отмеченных выше векторов. Таблица соединений объектов может, при загрузке векторных переменных, считываться с файла. В этом случае необходимость в *sConnectionTable* отпадает.

Нетрудно заметить, что введенные выше векторные переменные предназначены для хранения значений соответствующих полей таблицы соединений объектов.

Введем переменные, которые хранят сформированную систему ОДУ. Для этого необходимо определиться с методом кодирования уравнений. Произведем кодирование в следующем виде. Будем использовать префиксную форму записи уравнений. Каждая переменная в уравнении будет кодироваться парой TID.OID,

где TID - идентификатор типа объекта; OID - идентификатор объекта в модели. Каждая операция кодируется такой же парой, но за кодом следует в скобках список операндов. Пусть в системе ОДУ встречается следующее уравнение:

$$\frac{dy_1}{dt} = a_{11}x_1 + a_{12}f_1 \qquad (1)$$

тогда его можно закодировать, например следующим образом:

1.1=2.2(3.3(5.4,7.5),3.6(5.7,6.8)) (2)

где, рассматривая цифры до точки: 1-код интегрального объекта, 2-код суммирующего объекта, 3-код объекта, производящего произведение, 5-код объекта, генерирующего константу, 6-код объекта, генерирующего функцию, 7-код объекта, являющегося входом модели; а цифры после точки являются идентификаторами объектов в модели. Так 4-код объекта, генерирующего константу a_{11}.

Строки вида (2) накапливаются на ПЭ, хранящих формируемые уравнения.

vNext - идентификационные номера объектов которые необходимо найти для дальнейшего конструирования уравнений.

vEquation - векторная переменная для хранения формируемых уравнений.

vObjectInX - вектор, хранящий коды объектов, соединенных на вход рассматриваемого объекта.

2.3. Описание алгоритма кросс метода автоматической генерации математических уравнений

На рис.1. приведена блок-схема алгоритма кросс-метода автоматического формирования системы ОДУ. Рассмотрим алгоритм подробнее.

1.Загрузить вектора vObjectType, vObjectID, vObjectInX.

2.Размножить цепочку. Полученная на первом шаге цепочка ПЭ сдвигается дважды вниз (в нашем случае максимальное число входов равно трем). В результате получим три идентичных цепочки. Во второй и третьей производит модификацию вектора *vObjectInX* значениями для второго и третьего входов объектов. Таким образом, получили три цепочки, каждая из которых соответствует определенному входу объектов.

Далее производится параллельная проверка всех ПЭ на которых загружены интеграционные объекты.

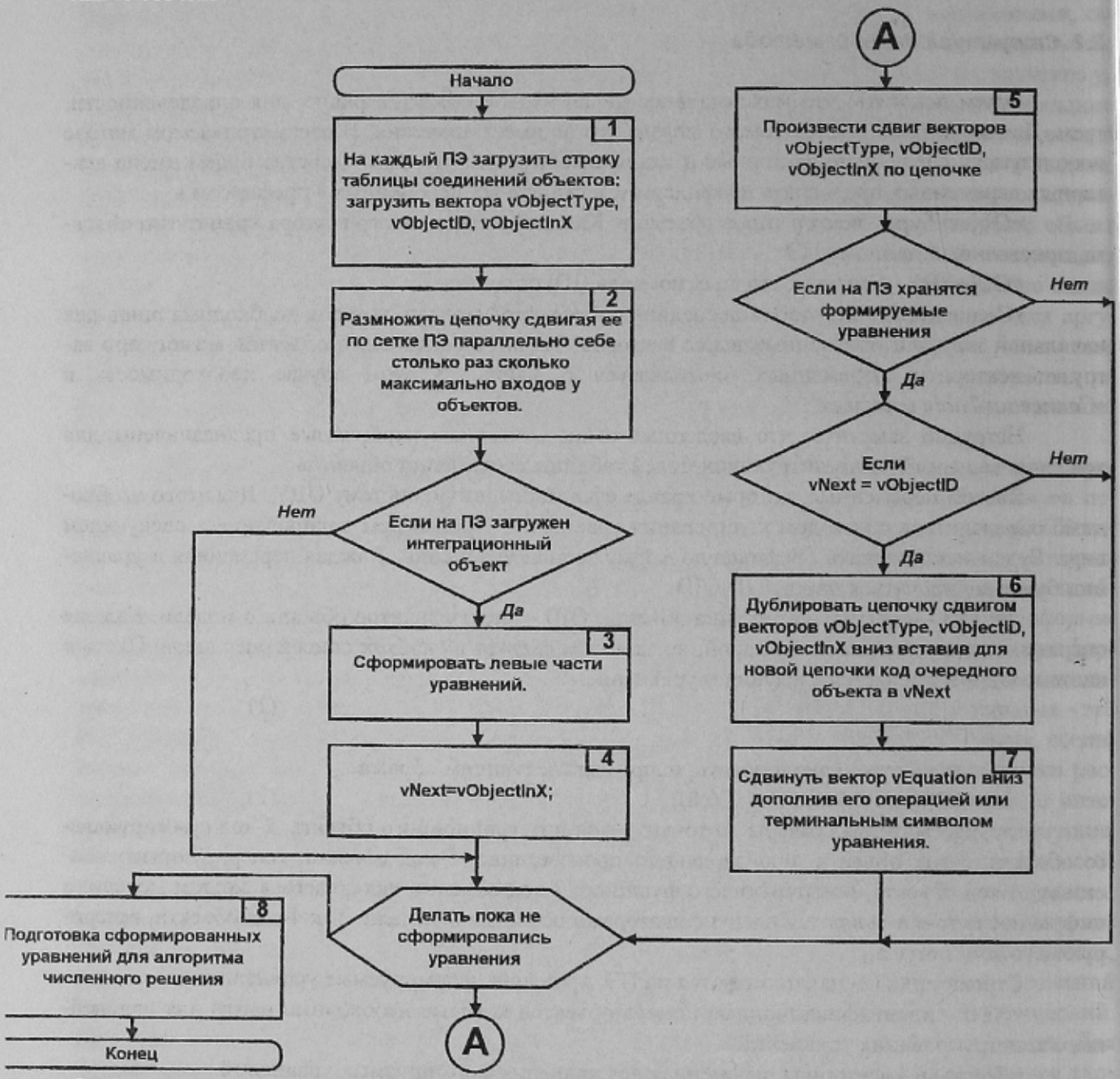

Рис.1. Алгоритм кросс-метода автоматической генерации математических уравнений.

3.Сформировать левые части уравнений. Рассмотрим простой пример. Пусть имеется следующая модель (рис.2). Это модель повторителя косинуса. На вход модели подастся функция Cos(t) - на выходе Out получаем ту же функцию.

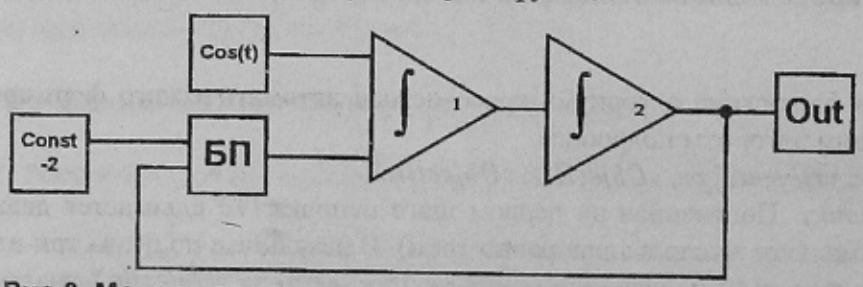

Рис.2. Модель повторителя косинуса.

В модели также использованы: объект генерации константы -2, объект «БП», выполняющий произведение своих входов, два интегрирующих объекта и объект «Out», накапли-

вающий выходные значения моделирования. Тогда результатом первых двух шагов алгоритма может быть следующее распределение объектов по сетке ПЭ (табл.1).

В нумерации процессорных элементов используется индекс i, что означает дублирова-

Таблица 1.

	ПЭ(i,1)	ПЭ(i,2)	ПЭ(i,3)	ПЭ(i,4)	ПЭ(i,5)	ПЭ(i,6)
Вход 1	Const -2	ПБ	Cos(t)	∫1	∫2	Out
Вход 2	Const -2	ПБ	Cos(t)	∫1	∫2	Out
Вход 3	Const -2	ПБ	Cos(t)	∫1	∫2	Out

ние цепочек по строкам. В данном случае уравнения будут генерироваться на процессорных элементах ПЭ(i,4) и ПЭ(i,5). В результате после данного шага алгоритма векторная переменная *vEquation* будет содержать правые части двух дифференциальных уравнений в следующем формате (табл.2).

Таблица 2.

	ПЭ(i,1)	ПЭ(i,2)	ПЭ(i,3)	ПЭ(i,4)	ПЭ(i,5)	ПЭ(i,6)
ПЭ(1,j)	0	0	0	Cos(t)	∫1	0
ПЭ(2,j)	0	0	0	ПБ	0	0
ПЭ(3,j)	0	0	0	0	0	0

В табл.2, вместо модельных номеров объектов, использованы их имена для наглядности. Из таблицы видно, что каждое из уравнений системы ОДУ будет распределено по столбцу ПЭ, которые содержат интегрирующие объекты. При каждом появлении нового нетерминального символа (операнда) будет производиться очередное дублирование цепочки вниз и вставка его кода в *vEquation*.

*4.*Прозводится присваивание векторной переменной *vNext* вектор номеров объектов следующих по дереву синтаксического разбора (соединенных на входы интеграторов в данном случае).

Далее алгоритм входит в цикл формирования уравнений.

5.Произвести сдвиг векторов vObjectType, vObjectID, vObjectInX.

Активизируются все ПЭ, на которых формируются уравнения.

После активизации необходимых процессорных элементов производится сравнения векторов *vNext* и *vObjectID*. Эта проверка необходима для поиска объектов являющихся еще не найденными операндами уравнений.

6.Дублировать цепочку сдвигом векторов vObjectType, vObjectID, vObjectIDInX вниз вставив для новой цепочки код очередного объекта в vNext. Если найден такой объект, то необходимо создать очередную цепочку и для нее - в векторную переменную *vNext* записать номера объектов, которые являются очередными недостающими операндами. Такие действия необходимы, если найденный операнд является нетерминальным символом, т.е. операцией.

7.Сдвинуть вектор vEquation вниз, дополнив его операцией или терминальным символом уравнения.

8. Подготовка сформированных уравнений для алгоритма численного решения.

3. Заключение

Предложенный метод автоматического генерирования системы ОДУ для модели ДССП параллелен. Его положительной стороной является не только параллелизм, но и возможность прямого использования параллельного алгоритма численного решения сформированной системы ОДУ. Таким образом, в данном случае необходимы следующие этапы полного цикла моделирования (рис.3).

Модель может быть построена либо визуальным способом, либо с помощью script-

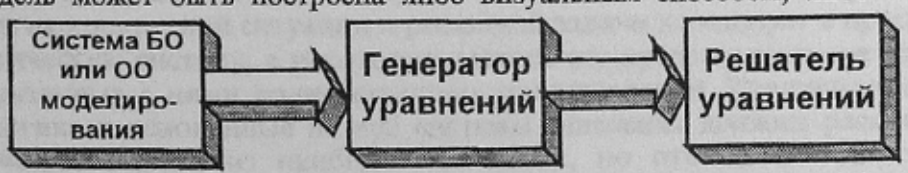

Рис.3. Этапы моделирования при автоматическом формировании математических уравнений.

языка. В любом случае формируется матрица соединений объектов, которая подается на вход генератора уравнений. Сам генератор реализует один из алгоритмов предложенных методов, используя тот язык параллельного программирования, компилятор которого имеется на выбранной супер-ЭВМ SIMD архитектуры. Результаты работы генератора передаются решателю систем ОДУ. Этот решатель также реализован с помощью параллельного языка программирования. Таким образом, рутинный этап предварительного построения уравнений перед моделированием динамической системы заменяется параллельным формированием систем ОДУ. Более того, как формирование уравнений, так и собственно их решение, производится на одной и той же машине, что позволяет формировать один исполняемый модуль, в котором вначале будет производиться генерация математических уравнений, а затем вызываться процедура решения сформированных уравнений.

Следует отметить, что способ кодирования уравнений может быть выбран различный. При обсуждении данного метода был использован метод явного кодирования математических уравнений, т.е. символьная запись системы ОДУ отображалась в цифровую. Но сам метод не зависит от конкретного типа кодирования уравнений. Например, в некоторых приложениях алгоритмы решения систем ОДУ могут принимать на вход матрицы, которые описывают уравнения.

Литература

1. Хокни Р., Джесссхоуп К. *Параллельные ЭВМ. Архитектура, программирование и алгоритмы.* Москва. "Радио и связь" 1986.
2. Anoprijenko A., Bräunl T., Reuter R., 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.
3. Schneider-Hufschmidt M. *Eine Entwicklungsumgebung fur adaptierbare Benutzungsoberflachen.* Mensch Computer Kommunication. 1993.
4. Святный В. *Гибридные вычислительные системы.* Киев. "Вища школа".1980.
5. Святный В., Баженов Л. *Дистрибутивные методы формирования параллельных моделей.* Данный сборник.