МАТЕМАТИЧЕСКИЕ МЕТОДЫ СИСТЕМНОГО АНАЛИЗА И УПРАВЛЕНИЯ

УДК 004.7

РАСПОЗНАВАНИЕ ИЗОМОРФНОГО ВЛОЖЕНИЯ АЛГОРИТМИЧЕСКИХ СЕТЕЙ

С. Н. Плотников

Воронежский филиал государственного университета морского и речного флота имени адмирала С.О. Макарова

Поступила в редакцию 07.05.2014 г.

Аннотация. Статья посвящена определению изоморфного вложения алгоритмических сетей и описанию необходимых преобразований для реализации, используя принцип деления вершин на классы. Также в этой статье представлен и подробно описан алгоритм распознавания изоморфности алгоритмических сетей и аспекты его применения при поиске в базах моделей.

Ключевые слова: алгоритмическая модель (*AM*) – формализованное описание сценария предметного специалиста для моделируемого процесса. Алгоритмическая сеть (*AC*) – конечный ориентированный нагруженный граф, вершинам которого сопоставлены операторы, а дугам – переменные, связываемые операторами. Изоморфизм – сходство двух или более объектов по форме или строению. Абстрагирование – это мысленное выделение, вычленение некоторых элементов конкретного множества и отвлечение их от прочих элементов данного множества. Алгоритм – точный набор инструкций, описывающих порядок действий исполнителя для достижения результата решения задачи за конечное время.

Annotation. This article is devoted to the definition of isomorphous investment of the algorithmic networks and description of conversions for implementation, using principle to division vertices into classes. Also in this article presented and described algorithm to identification of isomorphism algorithmic networks in detail and its using for base of algorithmic models.

Keywords: algorithmic model (AM) is a formalized description of the script of the subject expert for modelled process.Algorithmic network is a final focused loaded columns to which tops operators are compared, and the variables connected by operators are compared to arches. Isomorphism is a similarity of two or more objects under the form or a structure. Abstraction is a mental allocation, exarticulation of some elements of specific set and their derivation from other elements of the given set. Algorithm – a precise set of the instructions describing the order of actions of the executor for achievement of result of the solution to the problem for final time.

ВВЕДЕНИЕ

В настоящее время в математическом моделировании развиваются два равноправных взаимодополняющих направления. Первое направление основывается на технологии вычислительного эксперимента в трактовке А. А. Самарского [1], как новой технологии научных исследований, направленной на создание «фундаментальных моделей» как новых парадигм науки. Подход предполагает чрезвычайно высокую математическую подготовку главных участников и жесткое разделение труда между ними и специалистами предметной области. Технология вычислительного эксперимента, основанная на триаде «модель–алгоритм–программа» не рассчитана на массовое внедрение в повседневную работу конечных пользователей, которые лишь опосредованно участвуют в разработке моделей.

Второе направление названо Г. С. Поспеловым новой информационной технологией моделирования и ориентировано на конечного пользователя, как на непосредственного разработчика модели [1]. Подход ориентирован на создание языков представления моделей, доступных для непрограммирующего пользователя, и инструментальных средств с элементами искусственного интеллекта, обеспечивающих поддержку в разработке моделей, выборе численного метода решения, автоматизации синтеза расчетной программы, организации интерфейса. Направление рассчитано на внедрение методов моделирования в среду неподготовленных конечных пользователей.

Проблема снижения барьера между пользователем и ЭВМ появилась сразу же, как появилась ЭВМ, и остается актуальной и в настоящее время. Эта проблема формулируется как задача разработки дружественного интерфейса. Одним из путей решения проблемы является использование графического представления, которое может реально уменьшить барьер между человеком и ЭВМ. В этом ряду стоит и формализм алгоритмических сетей, предназначенный для описания алгоритмических моделей, предложенный В. В. Иванищевым около 30 лет назад [2]. Под алгоритмической моделью понимается формализованное описание сценария предметного специалиста для моделируемого процесса, структура которого сопоставима со структурой причинно-следственных и временных зависимостей между явлениями моделируемого процесса, вместе со всей информацией, необходимой для ее программной реализации. Разработка теории алгоритмических сетей, методов представления моделей на основе данного формализма и построение систем автоматизации моделирования выполнялись в Санкт-Петербургском институте информатики и автоматизации РАН [3]. Алгоритмические сети использовались для создания ряда моделей, которые эксплуатировались в различных организациях (например, Госплан РСФСР) и используются и в настоящее время. Однако язык представления требует средств поддержки автоматизированного построения моделей. Актуальным моментом автоматизации построения моделей является возможность использования фрагментов ранее созданных моделей, использование баз данных моделей.

Прежде чем перейти к описанию методов поиска фрагментов в базах моделей введем их формальные определения [3].

Начнем с определения алгоритмической сети, так как она является отображением основной части алгоритмическая модели, ее вычислительной схемы.

Алгоритмическая сеть определяется как ориентированный гиперграф без петель при вершинах $G(V, X)$, в котором дуги обозначают модельные переменные $x_i \in X$, $i = \overline{1,n}$, а вершины $v_j \in V$ – функциональные соотношения (операторы), $f_j \in F$, $j = \overline{1,m}$, связывающие модельные значения переменных на интервале времени, соответствующем Δt [3].

Структура переменных сети $X = X_{ВХ} \cup X_{ВЫХ} \cup X_{ВН}$, где $X_{ВХ}$, $X_{ВЫХ}$, $X_{ВН}$ – соответственно входные, выходные и внутренние переменные, причем, $X_{ВХ} \cap X_{ВЫХ} = \varnothing$, $X_{ВХ} \cap X_{ВН} = \varnothing$, $X_{ВЫХ} \cap X_{ВН} = \varnothing$. Учитывая, что все вершины AC наблюдаемы, можно задать $X = X_{ВХ} \cup X_{ВЫЧ}$, $X_{ВЫЧ}$ – вычисляемые переменные АС, $X_{ВЫЧ} = X_{ВЫХ} \cup X_{ВН}$, где $X_{ВЫЧ}$ – вычисляемые переменные.

Формально алгоритмическая сеть (AC) может быть определена следующим образом:

$$AC := < P, Q, X, F, P \to F, Q \to X >$$

где P – множество вершин сети; Q – множество дуг сети; X – множество переменных, при этом $X = \cup X_i$; X_i – множество переменных i -й вершины; I – множество всех индексов вершин; F – множество операторов; $P \to F$ – изоморфное отображение P на F, т. е. индекс оператора можно считать индексом вершины; $Q \to X$ – отображение Q на X.

Оператор $f_i \in F$ описывается как $f_i := < X_{i\text{in}}, f^i, X_{i\text{out}} >$, где $X_{i\text{in}}$, $X_{i\text{out}}$ – множество входных и выходных переменных оператора, f^i – символ операции или функции. Для некоммутативных операторов в $X_{i\text{in}}$,

$X_{i\text{out}}$ должен быть задан частичный порядок для их элементов. Так, для операции вычитания в $X_{i\text{in}}$ необходимо отметить уменьшаемое, порядок следования остальных переменных несущественен. В общем случае f^i может быть именем некоторого программного модуля или другой AC. Операторы могут быть определены над действительными числами, логическими переменными, скалярами, векторами, матрицами, информационными структурами и др. В состав операторов AC включается оператор задержки, который задает исходное состояние моделируемому процессу и описывает переход модели на следующий шаг или определяет задержку появления некоторой переменной в части AC. В AC допускаются контура только если они включат в себя оператор задержки. AC будем называть канонической, если все операторы задержки определяют рекуррентные соотношения относительно одной и той же величины и с одинаковым шагом ее изменения. Иными словами, в канонической AC все операторы задержки срабатывают одновременно и обеспечивают формирование одного внешнего цикла счета операторов сети. В дальнейшем рассматриваем именно такие AC. Алгоритмическая модель формально определяется:

$$AM ::= < IM, OPO, IO, OP, AC, T, MD, X \to T, F \to T >$$

где IM – идентификатор модели; OPO – описание предметной области; IO – описание объекта, для которого разработана модель; OP – описание процесса, для которого разработана модель; AC – алгоритмическая сеть (описание структуры модели); T – множество терминов предметной области, используемых в модели, $T \subseteq TPO$, где TPO – множество всех терминов предметной области; MD – множество вариантов законов изменения значений переменных модели; $X \to T$ – отображение множества переменных AC модели в множество терминов предметной области, используемых в модели, возможно $X' \to T$, где $X' \subseteq X$; $F \to T$ – отображение множества операторов AC модели в множество терминов предметной области, возможно $F' \to T$, где $F' \subseteq F$; $X \to T \cap F \to T = \varnothing$; если модель не пуста, то всегда не пусты IM и AC.

Для AC определены отношения равенства и изоморфности. Две AC равны $(AC_1 = AC_2)$, если и только если их множества F совпадают $(F_1 = F_2)$. AC изоморфны $(AC_1 \sim AC_2)$, если и только если между их множествами F можно определить взаимно-однозначное отображение $(F_1 \sim F_2)$, при условии что поставленные в соответствие элементы множеств F_1, F_2 имеют совпадающие символы операций f^i и одинаковые мощности множеств $\text{in}(f_i)$ и $\text{out}(f_i)$. То есть изоморфные AC отличаются друг от друга только обозначениями переменных. Соответственно изоморфное вложение для AM будем рассматривать как установление изоморфизма одной AC фрагменту другой AC. Для поиска фрагментов возможны два случая:

1. Все переменные моделей относятся к одному тезаурусу, тогда для поиска фрагментов AC достаточно использовать веденные для AC операции пересечения, на которой основаны алгоритмы установления равенства и вложения [3].

2. Переменные моделей относятся к разным тезаурусам. В данном случае, для поиска полной сети может быть использован алгоритм, описанный в [4], но для поиска фрагмента, данный алгоритм не подходит. Необходимо разработать новый алгоритм.

Алгоритм установления изоморфного вложения требует установить соответствие между вершинами двух AC и носит переборный характер. Каждая вершина характеризуется своим индексом, множеством переменных связанной с ней X_i, символом оператора, сопоставленного вершине f^i, который у соответствующих друг другу вершин сравниваемых AC всегда один и тот же. То есть, при установлении соответствия вершин, требуется определить соответствие переменных вершин имеющих одинаковые f^i и одинаковое число входных и выходных переменных.

Если сгенерировать все возможные варианты переобозначения переменных одной модели по образцу обозначения другой, а потом их все перебрать и проанализировать, то если сети изоморфны или изоморфно вложи-

мы, всегда можно будет найти, хотя бы один такой вариант. Эта идея положена в основу алгоритмов установления изоморфизма и изоморфного вложения графов и конечных автоматов.

Используем следующее допущение: разбиение множеств вершин каждого из графов на непересекающиеся классы и подклассы, между элементами которых возможно сопоставление имён переменных, должно, как минимум, не увеличивать, а зачастую уменьшать число возможных вариантов сопоставления имён переменных сравниваемых сетей; увеличение числа классов и подклассов увеличивает вероятность снижения числа вариантов.

Проведём предварительное преобразование анализируемых AC. Разорвем контура на входах операторов задержек и выполним для обеих AC их упорядочение по уровням вычислимости. То есть первый уровень будет состоять только из тех вершин, которые могут быть вычислены только на основании входных переменных, второй из тех вершин, в которых хоть одна входная переменная вычисляется в первом уровне (но не в других уровнях) и т. д., выходы операторов задержки рассматриваем как входные переменные. В результате в каждой AC множества вершин разобьётся на классы. Каждая вершина в AC характеризуется собственными $\left|\mathrm{in}\left(f_i\right)\right|$ (множество входных переменных оператора), $\left|\mathrm{out}\left(f_i\right)\right|$ (множество выходных переменных оператора) и f^i (символ операции с индексом), соответствующим ей уровнем вычислимости и списком вершин AC связанных с ней или по входу или по выходу. Каждая вершина из такого списка также характеризуется собственными $\left|\mathrm{in}\left(f_i\right)\right|$, $\left|\mathrm{out}\left(f_i\right)\right|$ и f^i, соответствующим ей шагом алгоритма. Все вершины, имеющие одинаковые перечисленные характеристики попадают в один класс. Таким образом имеем достаточно подробное деление на классы, которое в подавляющем большинстве практических случаев обеспечивает наличие одного или нескольких классов вершин состоящих из одного элемента, что существенно сокращает число возможных сопоставлений, а в ряде случаев однозначно определяет возможность переобозначения переменных в одной из AC. Реализация алгоритма разбиения на классы требует многократных просмотров множества F, но, в силу своей однонаправленности и однозначности определения принадлежности вершины к какому-либо классу, будет иметь полиномиальную оценку верхней границы сложности.

AC изоморфно вложимы (обозначим это $\sim\subseteq AC_2$), если и только если между их множеством $F1$ и некоторым подмножеством $F2$ можно определить взаимно-однозначное отображение (изоморфизм), при условии что поставленные в соответствие элементы множества $F1$ и подмножества $F2$ имеют совпадающие символы операций f^i и одинаковые мощности множеств $\left|\mathrm{in}\left(f_i\right)\right|$, $\left|\mathrm{out}\left(f_i\right)\right|$. То есть изоморфные AC отличаются друг от друга только обозначениями переменных.

Алгоритм распознавания изоморфности AC построен на основе аналогичных алгоритмов распознавания строгой эквивалентности алгоритмов. На изоморфность имеет смысл проверять только сети имеющие одинаковое число классов вершин, и если соответствующие друг другу классы равномощны. Алгоритм организует просмотр всех возможных вариантов переобозначения и заключается в поочерёдном просмотре вершин одной AC и выделенных классов в другой и выбор для каждой вершины одной AC вершины другой AC принадлежащей к тому же классу, переобозначению переменных выбранной вершины, с учётом ранее сделанных переобозначений. Если возникает противоречие с ранее сделанными переобозначениями, то алгоритм возвращается и начинает формировать другой вариант преобразования для анализируемой вершины. Если таких элементов в классе нет, то запоминает вариант переобозначения для всех ранее переобозначенных вершин как недопустимый и возвращаются к началу просмотра и т.д. Если удалось найти хотя бы один вариант переобозначения, то сети изоморфны. Если варианта переобозначения не найдено, то сети не изоморфны. Предпочтительно начинать анализ с классов

имеющих наименьшее число элементов. Данный алгоритм подробно рассмотрен в [4].

Рассмотрим теперь процесс поиска фрагмента AC_2, который может быть эквивалентен AC_1. Прежде всего необходимо, чтобы число уровней вычислимости было в AC_2 было не менее чем в AC_1, далее, чтобы мощности и число всех выделенных подклассов AC_2 в каждом уровне были не менее чем в AC_1. После данных проверок проводим анализ на уровне классов, ищется сначала в первом уровне вычислимости AC_2 похожие классы вершин равномощные или с большей мощности чем в AC_1. Далее отслеживаются классы вершин, с которыми они связаны (то есть, в которых есть связанные вершины) на следующем уровне, и так до тех пор пока не будут уровни вычислимости AC_2. После получения «графа классов» производится удаление из отобранных классов всех вершин, которые не имели связи с вершинами в других классах, затем производится выбор некоторого набора вершин из класса первого уровня вычислимости выделенного фрагмента AC_2, прослеживаются их связи до последнего уровня вычислимости фрагмента и запускается алгоритм установления изоморфизма. Если ответ алгоритма отрицательный, то выбирается новый набор и так далее.

ЗАКЛЮЧЕНИЕ

Данный эвристический алгоритм соответствует определению изоморфного вложения AC, заканчивается в конечное время и имеет переборный характер. Алгоритм позволяет перебрать все варианты переобозначений и поэтому, если сети изоморфны, всегда найдёт соответствующий вариант переобозначений. В данном своём варианте он имеет экспоненциальную от числа вершин AC оценку верхней границы сложности. На основе имеющегося практического опыта известно, что используемый принцип деления вершин на классы, почти во всех случаях даёт такое число одноэлементных классов, которое позволяет однозначно определить существует ли вариант допустимого переобозначения переменных и его вид. В практических реализациях алгоритма можно использовать менее подробное деление на классы.

СПИСОК ЛИТЕРАТУРЫ

1. *Иванищев В.В., Марлей В.Е.* Введение в теорию алгоритмических сетей. – СПб. : СПбГТУ, 2000. – 180 с.

2. *Пономарёв В.М.* Алгоритмические модели в задачах исследования систем. – М. : Наука, 1980. – с. 4–8.

3. Введение в математическое моделирование: Учеб. Пособие / под. ред. П.В. Трусова. – М. : Логос, 2004.

4. *Самарский А.А., Михайлов А.П.* Математическое моделирование. – М. : Физматлит, 2005.

Плотников Сергей Николаевич – ст. преподаватель кафедры информационных систем и технологий, Воронежский филиал «Государственного университета морского и речного флота имени адмирала С.О. Макарова». Тел. 8-920-226-00-44, 473-227-96-33. E-mail: 279622@mail.ru

Sergey Plotnikov – Senior Lecturer, Department of Information Systems and Technology, Voronezh branch «of the State University of Marine and River Fleet Admiral S.O. Makarov». Тел. 8-920-226-00-44, 473-227-96-33. E-mail: 279622@mail.ru