Опубликовано в трудах Второй Всероссийской научной конференции «Методы и средства обработки информации» .– М.: МГУ, 2005.– с. 385–387.

Корнеев Г.А., Шалыто А.А

ПРЕОБРАЗОВАНИЕ ПРОГРАММ В СИСТЕМУ ВЗАИМОДЕЙСТВУЮЩИХ КОНЕЧНЫХ АВТОМАТОВ

Санкт-Петербургский государственный университет информационных технологий, механики и оптики, Санкт-Петербург
kgeorgiy@rain.ifmo.ru, shalyto@mail.ifmo.ru

С 1991 г. в России развивается SWITCH-технология, которая базируется на автоматном программировании [1]. При развитии этой технологии встает вопрос о формальных методах построения конечных автоматов или их систем по программам на императивных процедурных языках программирования. В работе [2] был предложен метод преобразования (не содержащих рекурсии) программ. В дальнейшем это метод был развит, что позволило преобразовывать рекурсивные программы [3].

Отметим, что в области аппаратного обеспечения эта задача рассматривалась уже в 70-х годах и ее решение для одной процедуры приведено в работе [4].

Во всех указанных работах строился один автомат, который позволял осуществлять трассировку только в прямом направлении, в то же время возможность трассировки назад в некоторых приложениях является весьма важной [5]. Кроме того, рассматривались только программы, состоящие из одной процедуры.

В работе [6], при участии авторов, был предложен метод преобразования программы из произвольного числа рекурсивных процедур в систему взаимодействующих автоматов. Получаемая система автоматов обеспечивает трассировку исходной программы, как в прямом, так и в обратном направлении.

В настоящей работе предлагается основанный на статье [6] формальный метод построения такой системы автоматов по программе. Для системы автоматов, построенной в результате применения предлагаемого метода, доказывается, что трассировка в обоих направлениях осуществляется корректно.

Предлагаемый метод позволяет преобразовывать программы, состоящие из блочных операторов, операторов цикла с предусловием, полного и укороченного ветвления, присваивания и вызова процедуры.

С одной стороны, данный язык является достаточно простым для формального описания процесса преобразования, а с другой — содержит операторы присваивания, ветвления, цикла и блочный оператор, и как следствие, является достаточно богатым для описания произвольной программы [7] и позволяет описывать рекурсивные программы.

В предлагаемом методе по программе строится система взаимодействующих автоматов, содержащая по два автомата для каждой процедуры: *прямой* (для трассировки вперед) и *обратный* (для трассировки назад). Автоматы, построенные по одной процедуре, имеют общее множество состояний, но разные функции переходов [6].

Для систем конечных автоматов, построенных в соответсвии с предлагаемым методом доказываются следующие свойства:

- *Адекватность* — система автоматов выполняет те же действия, что и исходная программа.
- *Обратимость* — при выполнении действий обратного автомата будут восстановлены исходные значения всех переменных, при условии, что в точке входа они были такими же, как и в точке выхода при прямом проходе.
- *Полнота* — для каждого состояния, не являющегося конечным, при любых значениях переменных условие на одном из переходов должно быть истинно.
- *Непротиворечивость* — условия на переходах из одного состояния не могут быть истинными одновременно.
- *Отсутствие недостижимых состояний* — любое состояние может быть достигнуто по переходам из начального состояния. При этом состояние называется достижимым, если оно начальное или в него существует переход из достижимого состояния. Таким образом, проверяется только принципиальная возможность достижения состояний, а условия на переходах фактически игнорируются.

Доказательство *адекватности* и *обратимости* позволяет утверждать, что трассировка в обоих направлениях осуществляется корректно. *Полнота* и *непротиворечивость* обеспечивают детерминированность построенных автоматов. *Отсутствие недостижимых состояний* является естественным требованием.

Для каждого узла в дереве вывода преобразуемой программы строится фрагмент автомата — набор состояний и переходов. При этом у некоторых переходов, принадлежащих фрагменту может быть не определено начальное или конечное состояние. Такие переходы называются *входами* и *выходами* фрагмента соответственно. Заметим, что один переход может быть входом и выходом одновременно. В

данной работе рассматриваются только фрагменты автоматов с одним входом и одним выходом.

Рассмотренные свойства доказываются посредством структурной индукции на дереве вывода преобразуемой программы.

Для обеспечения трассировки назад создаваемая система автоматов использует стек, над которым производятся следующие операции:

- push(expr) — поместить значение выражения expr на вершину стека;
- pop() — прочитать значение на вершине стека и удалить его.

При преобразовании вершины дерева вывода программы во фрагменты автоматов, к уже построенным фрагментам добавляется не более двух состояний и четырех переходов. Таким образом, суммарное количество состояний и переходов в полученной системе автоматов линейно по количеству вершин в дереве вывода, а, следовательно, и по числу операторов в исходной программе.

Предложенный метод является математической основой создания пакета *Vizi*, предназначенного для создания визуализаторов алгоритмов, используемых при обучении основам программирования и дискретной математики.

Работа выполнена при поддержке РФИИ по грантам №02-07-90114 и №05-07-90011.

1. *Шалыто А. А.* SWITCH-технология. Алгоритмизация и программирование задач логического управления. СПб.: Наука, 1998.
2. *Шалыто А.А., Туккель Н. И.* Преобразование итеративных алгоритмов в автоматные // Программирование.— 2002. — № 5. — с.12-26.
3. *Туккель Н. И., Шалыто А. А., Шамгунов Н. Н.* Реализация рекурсивных алгоритмов на основе автоматного подхода // Телекоммуникации и информатизация образования. — 2002. — № 5.
4. *Баранов С. И.* Синтез микропрограммных автоматов (граф-схемы и автоматы). Л.: Энергия, 1979.
5. *Казаков М. А., Столяр С. Е.* Визуализаторы алгоритмов как элемент технологии преподавания дискретной математики и программирования // Международная научно-методическая конференция “Телематика-2000”. СПб.: 2000. – с.189-191.
6. *Казаков М. А., Корнеев Г. А., Шалыто А. А.* Метод построения логики работы визуализатора алгоритмов на основе конечных

автоматов // Телекоммуникации и информатизация образования. — 2003. — №6. — с. 27-58.
7. *Грис Д.* Наука программирования. М.: Мир, 1984.