Опубликовано в трудах X Всероссийской научно-методической конференции «Телематика-2003» .– СПб.: СПбГИТМО (ТУ), 2003.– с. 378–379.

ПОСТРОЕНИЕ ЛОГИКИ РАБОТЫ ВИЗУАЛИЗАТОРОВ АЛГОРИТМОВ НА ОСНОВЕ АВТОМАТНОГО ПОДХОДА

Г.А. Корнеев, М.А. Казаков, А.А. Шалыто

Санкт-Петербургский государственный институт точной механики и оптики (технический университет)

При изучении алгоритмов обработки информации, представляемой различными структурами данных [1, 2], важную роль играют визуализаторы алгоритмов, позволяющие в наглядной форме динамически отображать детали их работы.

Визуализатор — это программа, в процессе работы которой на экране компьютера динамически демонстрируется применение алгоритма к выбранному набору данных. Визуализаторы позволяют изучать работу алгоритмов в пошаговом режиме, аналогичном режиму трассировки программ. При изучении большинства алгоритмов наряду с режимом "шаг вперед" весьма полезен также и режим "шаг назад" [3], позволяющий более быстро и полно понять алгоритм. Например, в алгоритмах поиска с возвратом, для того чтобы понять, почему та или иная ветвь поиска отброшена, часто необходимо сделать несколько шагов назад.

Можно утверждать, что к настоящему времени основные достижения в проектировании визуализаторов относятся к сфере педагогики [3], а достижения в сфере технологии создания визуализаторов практически отсутствуют. В частности, отсутствует метод, позволяющий по алгоритму формально и единообразно создавать логику работы визуализатора.

В работе [4] было предложено использовать особенности автоматных программ для построения визуализаторов. В настоящей работе предлагается метод построения логики работы визуализатора по заданному алгоритму. Метод позволяет представить логику работы визуализатора системой взаимосвязанных конечных автоматов. Система состоит из пар автоматов, каждая из которых состоит из “прямого” и “обратного” автоматов, которые обеспечивают пошаговое движение по алгоритму вперед и назад соответственно.

Для реализации логики работы алгоритма выбраны автоматы Мура, в которых действия выполняются только в состояниях. Состояния получаемых автоматов взаимооднозначно соответствуют операторам преобразуемой программы. Действия в состоянии осуществляются только при переходе в него из другого состояния. Это позволяет отображать операции, выполняемые алгоритмом и результаты этих операций наиболее естественным образом.

Предлагаемый метод состоит из следующих шагов.

1. По визуализируемому алгоритму пишется реализующая его процедурная программа.

2. Используемые переменные (в том числе, переменные циклов) выносятся в модель данных (например, в структуру, класс или запись). При этом процедурам, применяемым в программе, параметры должны передаваться через модель данных. Результаты работы процедур так же должны передаваться через модель данных.

3. Программа преобразуется с целью использования в ней следующих операторов, первые пять из которых являются управляющими:
– последовательность операторов;
– полный оператор ветвления (`if-then-else`);
– укороченный оператор ветвления (`if-then`);
– цикл с предусловием (while);
– вызов процедур;
– оператор присваивания.

4. Программа преобразуется в систему взаимосвязанных автоматов формальным образом.

Опишем более подробно смысл некоторых шагов.

Второй шаг необходим для того, чтобы полученную программу можно было разбить на систему автоматов, не заботясь о передаче значений локальных переменных между автоматами.

В результате третьего шага получается программа, более удобная для преобразования в систему автоматов, так как она не содержит громоздких синтаксических конструкций. Данное преобразование основано на теореме структурирования [5], в соответствии с которой любую программу можно преобразовать так, что она будет содержать три типа управляющих конструкций (с одним входом и одним выходом каждая): последовательность, один из операторов ветвления и один из операторов цикла. Из циклов выбран цикл с предусловием (while), так как он является наиболее универсальным.

В результате применения описанного метода получается модель данных и система взаимосвязанных автоматов, управляющих ей.

При выполнении четвертого шага для каждого оператора программы строятся фрагменты прямого и обратного автоматов. Каждый фрагмент имеет один вход и один выход. Полученные фрагменты объединяются в соответствии с формальными правилами, и образуют автоматы, позволяющие пошагово эмулировать исходную программу в прямом и обратном направлении. При этом прямые и обратные автоматы, имеют одинаковые состояния и отличаются только функциями переходов и действиями, выполняемыми в состояниях. Предложенный метод позволяет преобразовывать как итеративные, так и рекурсивные программы. Для сохранения информации, требуемой для осуществления обратного прохода, используется стек. Автоматы управляются посредством воздействия на тактовый генератор через органы управления визуализаторов.

Работа выполнена при поддержке Российского фонда фундаментальных исследований по гранту 02-07-90114 “Разработка технологии автоматного программирования”.

Литература

1. Кнут Д. Искусство программирования. Том 1. Основные алгоритмы. М.: Вильямс, 2000.

2. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы. Построение и анализ. М.: МЦНМО, 1999.

3. Казаков М.А., Столяр С.Е. Визуализаторы алгоритмов как элемент технологии преподавания дискретной математики и программирования //Тезисы докладов международной научно-методической конференции "Телематика-2000". СПб.: СПбГИТМО (ТУ), 2000.

4. Казаков М.А., Шалыто А.А., Туккель Н.И. Использование автоматного подхода для реализации вычислительных алгоритмов //Труды международной научно-методической конференции "Телематика-2001". СПб.: СПбГИТМО (ТУ), 2001.

5. Лингер Р., Миллс Х., Уитт Б. Теория и практика структурного программирования. М.: Мир, 1982.