ТЕХНИЧЕСКИЕ НАУКИ

Ефанова Надежда Викторовна
магистрант
Муромцев Виктор Владимирович
канд. техн. наук, доцент, заведующий кафедрой

ФГАОУ ВПО «Белгородский государственный национальный исследовательский университет»
Белгородская область, г. Белгород

ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, ОСНОВАННОГО НА СПОСОБАХ ЗАДАНИЯ И ПОСТРОЕНИЯ ТАБЛИЦ РЕШЕНИЙ

Аннотация: *в данной статье рассматриваются вопросы целесообразности использования таблиц решений при проектировании программного обеспечения.*

Ключевые слова*: таблица решений, метод ветвления, блок-схема, логика, автоматическая обработка таблиц.*

Таблица принятия решений или таблица решений – это способ компактного представления модели со сложной логикой. Как и условные операторы в языках программирования, они устанавливают связь между действиями и условиями.

В отличие от традиционных языков программирования, таблицы принятия решений в своей обычной форме представляют связи между множеством независимых действий и условий.

Таблицы принятия решений, как правило, разделяются на четыре квадранта, как показано ниже в таблице 1.

Таблица 1

Таблица принятия решений

Условия	Варианты выполнения условий
Действия	Необходимость действий

В простейшем случае здесь:

– *условия* – это список всевозможных условий;

– *варианты выполнения условий* – это комбинация из выполнения, а также невыполнения условий из списка;

– *действия* – это список всевозможных действий;

– *необходимость действий* – это указание на выполнение или не выполнение соответствующих действий для каждой из комбинации условий.

Например, вариантов выполнения условия может быть несколько, не только два: да или нет, а также могут быть использованы цвета (красный, оранжевый, синий). Возможно применение нечёткой логики в более сложных таблицах.

Все действия могут быть элементарными или ссылаться на другие таблицы принятия решений. Необходимость выполнения действий может быть неупорядоченной и упорядоченной. В последнем случае, если при определённой комбинации выполнения условий возможно выполнение нескольких действий, то в таблице решений указывается их приоритет.

Таблицы решений представляют собой один из наиболее содержательных способов описания задачи. По своему содержанию они служат той же цели, что и стандартные блок-схемы. Отличительная особенность в том, что блок-схемы приспособлены исключительно для безошибочной разработки и модификации простейших в логическом смысле программ, а таблицы решений позволяют осуществлять записи логически сложных задач, но в наиболее простой форме, которая ориентирована на автоматическую разработку и модификацию алгоритмов.

Задача, записанная в форме таблиц решений, содержит все возможные блок-схемы ее решений. В этом множестве можно выбрать наилучшую с точки зрения заданного критерия эффективности объектной программы. Нет такой задачи, которая не могла бы быть сформулирована в терминах таблиц решений.

Эффективность использования таблиц решений зависит от логической сложности задачи: ведь чем выше сложность задачи, тем выше эффективность применения таблиц решений [2].

Таблицы решений можно разделить на два типа: с ограниченным и расширенным входом [3]. Они состоят из условий, действий и данных, т.е. из основных элементов всех программ, и поэтому используются как гибкий инструмент для описания данных и значительной части логики любой программы.

Как правило, условия формулируются так, что они становятся булевыми функциями, принимая два значения: «истина» и «ложь», соответственно обозначив эти условия значениями «Y» и «N».

Термин «расширенный вход» применяется для обозначения любого другого типа заполнения клеток. Таблицы решений позволяют систематически проверять каждую комбинацию значений условий так, чтобы быть уверенным, что не пропущена ни одна из них.

Для таблицы с ограниченным входом число возможных исходов равно 2^n, где n – число условий, т.е. можно заранее проверить общее число возможных исходов и выяснить, полна ли данная таблица.

Обычно применяют два способа, позволяющих сократить объем работы: первый состоит в использовании тире для обозначения того, что значение условия несущественно для данного правила [3]. В таблице решений с ограниченным входом каждое тире означает замену двух правил решений на одно. Если в правиле решений записано n тире, то одним столбцом заменено 2^n правил решений.

Второй способ заключается в использовании правила «иначе» [3]. Выписывают только те правила решения, для которых требуется специальная обработка, а к остальным применяют специальное правило «иначе», записываемое справа.

Существуют различные методы автоматической обработки таблиц решений: использование макропрограмм, использование интерпретатора, использование компиляторов, использование препроцессоров.

Для одной и той же таблицы решений можно построить несколько блок-схем. Они одинаковы в логическом смысле, т.е. при любом предположении о состоянии условий этой таблицы приходим к одному и тому же действию.

Проблема состоит в выборе лучшей из этих блок-схем. Если рассматривать всех их с точки зрения числа проверок, которые надо хранить, и среднего времени получения решения, то может оказаться, что одна блок-схема лучше с точки зрения затрат памяти, а другая – с точки зрения среднего времени получения решения. Поэтому следует с недоверием относиться к тому, что некоторый блок-схемный эквивалент таблицы решений является наилучшим, если неизвестны затраты на проверки условий и вероятность появления ключей и пока не установлена относительная важность двух названных критериев [3].

В процессе трансляции построение той или иной блок-схемы определяется выбором последовательности проверок. Следовательно, надо найти метод, позволяющий определить, какое условие необходимо проверить далее. Простым методом выбора очередной проверки может служить следующая схема: посчитать тире в каждой строке и выбрать ту, в которой число тире наименьшее (алгоритм Поллака).

Список литературы

1. Andersen H. R. An Introduction to Binary Decision Diagrams, Lecture Notes, 1999, IT University of Copenhagen.

2. Binary Decision Diagrams: Theory and Implementation. – Springer, 1998.

3. Meinel Ch., Theobald T. Algorithms and Data Structures in VLSI-Design: OBDD – Foundations and Applications. – Springer-Verlag, Berlin, Heidelberg, New York, 1998.

4. Knuth D. Fun With Binary Decision Diagrams (BDDs) [Электронный ресурс]: видео лекция / Режим доступа: http://myvideos.stanford.edu/player/slplayer.aspx?coll=ea60314a-53b3-4be2-8552-dcf190caocob&co=18bcd3a8-965a-4a63-a516-a1ad74af1119&0=true

5. Shank R. Decision Tables. – MDI Publications. – New York. – 1968.

6. Хамби Э. Программирование таблиц решений. – Пер. с англ. – М.: Мир, 1976.

7. Якоби Х. Введение в технику работы с таблицами решений. – М.: Энергия, 1979. – 112 с.