УДК 519.682.1

УНИФИКАЦИЯ И ТИПИЗАЦИЯ АЛГОРИТМИЧЕСКИХ СРЕДСТВ ПРИ ПРОЕКТИРОВАНИИ АВТОМАТИЗИРОВАННЫХ СИСТЕМ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ

И.В. Чумаченко, канд. техн. наук, В.В. Косенко

Национальный аэрокосмический университет им. Н.Е. Жуковского "ХАИ"

В данной работе предлагается метод унификации и типизации алгоритмических средств при проектировании автоматизированных систем обработки информации и управления, основанный на применении универсальных алгоритмических преобразователей. Рассмотрены принципы построения и способы настройки. Сформулированы основные задачи, возникающие при проектировании алгоритмических преобразователей и методы их решения.

* * *

У даній роботі пропонується метод уніфікації та типізації алгоритмічних засобів при проектуванні автоматизованих систем обробки інформації та управління, в основі якого лежить концепція універсальних алгоритмічних перетворювачів. Розглянуті принципи побудови та способи настройки. Сформульовани основні задачі, що виникають при проектуванні алгоритмічних перетворювачів та методи їх рішення.

* * *

The analysis of logic efficiency is made through the developed method of adjustment of algorithmic converters. The adjustments of algorithm for private and universal algorithms are determined. The research of program and hardware realization of universal algorithm and is carried out. The functional bases of algorithm and universal algorithmic converter are determined.

Теория проектирования автоматизированных систем обработки информации и управления в настоящее время интенсивно развивается. Это сложный многогранный процесс, для которого требуется соответствующая методология, качественно новый формальный подход к проектированию систем, который позволил бы проводить проектирование формальных языков, структур данных, алгоритмов и программ, инвариантных относительно входа и выхода автоматизированных систем. Причем, на всех этапах проектирования систем должно сохраняться методическое единство языковых и математических средств [1].

Анализ известных АСОИУ выявил их недостатки:

- недостаточно развитые инструментальные средства алгоритмической поддержки; что препятствует широкому применению проблемно-ориентированных языков программирования;
- отсутствует гибкая настройка системы на реализацию алгоритмов в соответствии с ее архитектурой, т.е. отсутствует специальный механизм для настройки системы на реализацию необходимых функций в зависимости от входной информации;
- отсутствие универсальной системы компиляции, настраиваемой на реализацию определенного языка из заданного множества.

Поиски решения указанной проблемы ведутся в разных направлениях, и в том числе на путях, предусматривающих пересмотр традиционных методов построения систем управления [2].

Одним из перспективных направлений является унификация алгоритмических, программных и аппаратных средств, т.е. создание универсальных в заданном классе средств, реализующих при соответственном преобразовании заданное множество типовых решений (алгоритмических, программных или аппаратных).

Вопросы унификации и типизации алгоритмических средств имеют первостепенное значение при разработке автоматизированных систем обработки информации и управления. Под унификацией понимают рациональное сокращение типов изделий или процессов одинакового функционального назначе-

ния, а под типизацией - разработка типовых решений, в которых отражаются общие для ряда процессов элементы или характеристики [3]. Унификация и типизация алгоритмов обработки информации позволяет снизить общее их число, что упрощает организацию адаптивного комплекса алгоритмов. Унификация и типизация приводят к тому, что модернизация объектов или расширение их состава не влечет за собой существенных изменений математического обеспечения.

Примером алгоритмических преобразователей является разработанное устройство обработки информации [4], предназначенное для реализации типовых алгоритмов обработки информации.

На рис. 1. представлена функциональная схема устройства.

Устройство содержит шину данных 1, шину результата 2, операционные блоки 3, 4, 5, 6, управляющие входы 7, 8, 9, мультиплексоры 10, 11, 12.

Работает устройство следующим образом.

Обозначим функции, реализуемые операционными блоками 3, 4, 5, 6 соответственно A, B, C, D, а значения логических переменных на входах 7 и 8 соответственно α и β.

В зависимости от заданного режима работы устройство реализует типовые относительно Tn классификации алгоритмы обработки информации для двух условных операторов.

Если на управляющий вход 9 подается сигнал 0, а на управляющие входы 7 и 8 соответственно сигналы α и β, то на шине результата реализуется функция

$$F_1 = (A \vee B)^{\alpha} (C \vee D)^{\beta}.$$

Если на управляющий вход 7 подается сигнал 0, на управляющие входы 8 и 9 соответственно сигналы α и β, то на шине результату реализуется функция

$$F_2 = (A (C \vee D)^{\beta} \vee B)^{\alpha}.$$

При разработке унифицированных алгоритмических, программных и аппаратных средств, универсальных в заданном классе, возникают следую

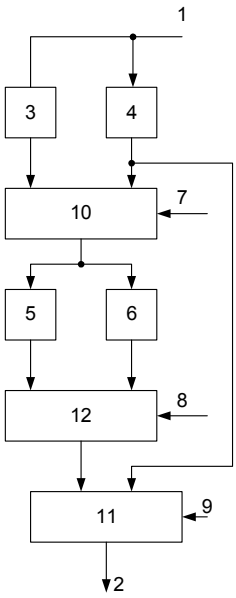

Рис. 1. Устройство обработки информации

щие задачи:

- исследовать способы преобразования алгоритмических, программных и аппаратных средств, реализующих заданное множество типовых решений;

- разработать методы проектирования универсальных преобразователей.

Множество алгоритмов обработки информации и управления можно разбить на два класса, в зависимости от вида условных операторов: с некоммутативными условиями и с коммутативными условиями. Каждый класс алгоритмов имеет свои особенности, что повлияло на методы проектирования универсальных преобразователей.

Для алгоритмов с некоммутативными условиями в качестве типовых решений предлагаются параметрические r- формулы], которые более адекватно соответствуют проблеме непосредственной реализации языков на основе РАПЛ. Идея этих формул заключается в создании типовых структур (супермодулей), порождающих языки определенного класса для реализации:

1. Арифметических и логических выражений, а так же выражений общего вида.

2. Дизъюнктивных форм в АРЕКС и Р-СИАА.

3. Квазирегулярных форм в АРЕКС и Р-СИАА.

4. Рекурсивных форм в АРЕКС и Р-СИАА.

5. Универсальных и специализированных формальных языков и т.д.

Метод построения универсальных алгоритмических преобразователей этого типа подробно описан в работе [5].

Рассмотрим способы настройки алгоритмических преобразователей

Пусть $X = \{ x_1, \ldots, x_к\}$ – множество переменных алгоритма $A(x_1, \ldots, x_к)$.

Настройка алгоритма – это подстановка в качестве переменных некоторых фиксированных значений (0 или 1). При этом множество переменных X разбивается на два непересекающихся подмножества X_1 и X_2, где X_1 – подмножество фиксированных переменных, а X_2 – множество независимых переменных. В результате настройки регулярное выражение, описывающее некоторый алгоритм $A(x_1, \ldots, x_к)$, преобразуется в тождественное ему при соответствующих значениях фиксированных переменных, который назовем частным алгоритмом. Алгоритм, реализующий заданное множество частных алгоритмов, будем называть универсальным в заданном классе алгоритмом. В принципе, любой алгоритм можно рассматривать как универсальный для меньшего количества переменных, а полученные в результате настройки алгоритмы – как частные алгоритмы.

Если для всех заданных настроек множество X_1 фиксировано, т.е. в настройке участвуют одни и те же переменные, то будем говорить, что используются разделимые переменные, в противном случае – неразделимые переменные. Выбор разбиения переменных зависит от требований к настройкам. Количество настроек для неразделимых настроечных переменных больше, чем для разделимых настроечных переменных. Это обусловлено тем, что каждой настройке соответствует рассмотрение на алгоритмической позиционной диаграмме определенных конфигураций.

Пример 1. Алгоритм *A(a, b, c, d, e),* описывается регулярным выражением

$$A(a, b, c, d, e) = (B(D \vee E)^b (F \vee G)^c \vee H(P \vee W(Q \vee V)^e)^d)^a$$

и при соответствующих настройках преобразуется в частные алгоритмы от меньшего числа переменных:

$$A_1 = A(\alpha, 0, \beta, 0, \gamma) = (BE(F \vee G)^{\beta} \vee HW(Q \vee V)^{\gamma})^{\alpha};$$

$$A_2 = A(1, 0, \alpha, \beta, 0) = BE(F \vee G)^{\alpha}.$$

При программной реализации универсального алгоритма, реализующего заданное множество частных алгоритмов, все операционные блоки и универсальный алгоритм оформляются в виде процедур (подпрограмм или функций), в которых формальные параметры соответствуют переменным $x_1, \ldots, x_к$. Любой из формальных параметров представляется в виде параметров – переменных любого типа, например, BOOLEAN, WORD или INTEGER .

Механизм замены формальных параметров на фактические позволяет нужным образом настроить универсальный алгоритм, реализованный в виде процедуры, на выполнение соответствующего частного алгоритма. При этом множество фактических параметров состоит из параметров, принимающих фиксированное значение и параметров – переменных, которые для различных частных алгоритмов, в общем случае, различны.

Ниже приведен фрагмент программы на языке PASCAL, реализующий универсальный алгоритм для примера 1.

```
PROGRAM  EXAMPLE1;
  VAR …
  {  реализация универсального алгоритма }
PROCEDURE  ALGORITHM_A(a, b, c, d, e);
  BEGIN
  …
```

```
END;
{ основная программа }
BEGIN
{ настройка на реализацию частного алгоритма A1 }
ALGORITHM_A (α, 0, β, 0, γ);
{ настройка на реализацию частного алгоритма A2 }
ALGORITHM_A (1, 0, α, β, 0);
END.
```

При аппаратной реализации универсальных алгоритмических преобразователей настройка производится путем подачи настроечных и управляющих сигналов на соответствующие входы дискретного устройства , реализующего универсальный алгоритм .

При исследовании алгоритмов возникают две задачи:

- оценить логическую эффективность алгоритма при заданном множестве настроек, т.е. определить множество реализуемых им различных частных алгоритмов;
- построить универсальный алгоритм, реализующий с помощью настроек заданное множество частных алгоритмов.

Для оценки логической эффективности необходимо для каждой настройки определить вид реализуемого частного алгоритма, что удобно произвдить с помощью алгоритмических позиционных диаграмм, т.к. каждой настройке соответствует определенная конфигурация на диаграмме.

При анализе логической эффективности, в большинстве, случаев интересует не только множество реализуемых с помощью настройки частных алгоритмов, а множество реализуемых частных алгоритмов, принадлежащих некоторым классам эквивалентности. Для оценки логической эффективности в этом случае необходимо определить множество реализуемых частных алгоритмов, определить его тип и отобрать множество различных реализуемых типов. Структура алгоритма влияет на его логическую эффективность и рассмотренный метод позволяет анализировать логическую эффективность алгоритмов и выбирать наиболее эффективные.

Литература

1. Жихарев В.Я., Илюшко В.М., Чумаченко И.В. Математические основы проектирования рекурсивных автоматов с программируемой логикой: Монография.- Харьков: Факт, 1999. - 144 с.

2. Жихарев В.Я., Илюшко В.М., Чумаченко И.В. Проектирование электронных компиляторов: Монография.- Харьков: Факт, 1999. - 88 с.

3. Диагностирование и прогнозирование технического состояния авиационного оборудования/В.Г. Воробьев, В.В. Глухов, Ю.В. Козлов и др. Под ред. И.М.Синдеева. М.: Транспорт, 1984.- 191с.

4. Патент України № 38733 A, G06F17/11. Пристрій обробки інформації / Чумаченко І.В.- № 2000095247; Заявлено 12.09.2000; Опубл. 15.05.2001, Бюл. № 4.- 5 с.

5. Методы проектирования символьных процессоров: Монография / В.Я. Жихарев, В.М Илюшко, Н.В. Нечипорук, И.В. Чумаченко - Харьков: Факт, 2000. - 184 с.