Math-Net.Ru

Общероссийский математический портал

А. А. Тюгашев, Алгебраические модели управляющих алгоритмов и программ реального времени для космических аппаратов, *Вестн. Сам. гос. техн. ун-та. Сер. Физ.-мат. науки*, 2005, выпуск 38, 19–25

DOI: <https://doi.org/10.14498/vsgtu365>

Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и согласны с пользовательским соглашением
<http://www.mathnet.ru/rus/agreement>

Параметры загрузки:

IP: 52.87.193.239

26 апреля 2023 г., 17:22:31

Информатика

УДК 681.51:629.78

А. А. Тюгашев

АЛГЕБРАИЧЕСКИЕ МОДЕЛИ УПРАВЛЯЮЩИХ АЛГОРИТМОВ И ПРОГРАММ РЕАЛЬНОГО ВРЕМЕНИ ДЛЯ КОСМИЧЕСКИХ АППАРАТОВ

Описывается основанная на логико-алгебраическом подходе математическая модель для представления управляющих алгоритмов и программ реального времени, предназначенных для координации работы бортовой аппаратуры космических аппаратов. Предлагается модель семантики управляющего алгоритма реального времени. Приводится формальная теория (исчисление) управляющих алгоритмов. Рассмотрены примеры описания параллельных управляющих алгоритмов с помощью основанной на предлагаемой модели инструментальной графической программной системы.

В настоящее время особо актуально вопросы моделирования сложных автоматизированных систем, в работе которых управление и контроль над взаимосвязанными параллельными процессами, протекающими в реальном времени, имеют центральное значение. К примерам такого рода систем относятся системы многопроцессорной обработки; системы массового обслуживания (например абонентов сотовой телефонной сети); приложения, связанные с совместным решением одной задачи совокупностью ЭВМ, объединенных в локальную либо глобальную сеть; системы управления сложными технологическими комплексами; бортовые системы управления космических аппаратов [1] и др.

Для моделирования подобного рода систем применяются такие подходы, как системы переходов, таймированные автоматы, временная логика, алгебра процессов реального времени и др. [2–6].

В настоящей статье описывается специальная математическая модель управляющего алгоритма реального времени (УА РВ), базирующаяся на применении логико-алгебраического подхода. Вводится определение семантики УА РВ, показана возможность конструирования УА РВ путем последовательного применения логических и временных операций к элементам базового множества элементарных функциональных задач (ФЗ), в качестве которых могут рассматриваться как целые программы со сложной внутренней структурой, так и простые операторы языка программирования. Определяются понятия многовариантной и многоходовой моделей управляющего алгоритма, а также логико-временной схемы алгоритма. Кратко описываются возможности основанной на приведенных моделях инструментальной программной системы, которая позволяет с использованием визуального конструктора описывать УА РВ и автоматически получать его информационные и временные характеристики, а также генерировать управляющую программу, реализующую алгоритм.

При изучении семантики обычной (вычислительной) программы, как правило, она определяется результатом, получаемым по окончании ее работы.

Под результатом при этом подразумевается некоторый набор выходных данных, полученных в результате обработки набора исходных данных (ИД). При этом состояния программы (алгоритма), как начальное и конечное, так и промежуточные, определяются совокупностью значений в имеющейся программной памяти.

Например, известна аксиоматическая формализация семантики программы, исходящая из формулировки так называемых постусловий и предусловий [2]. Данные условия есть часть языка *алгоритмических логик*, предложенных Флойдом, Хоаром [7], математиками польской логической школы практически одновременно в конце 1960-х гг., и включающего условия вида $\{U\} S \{B\}$,

читающиеся следующим образом: « U – условие, относящееся к исходным данным программы S , истинное до ее выполнения, B – условие, относящееся к выходным данным программы S , которое должно быть истинным после ее исполнения».

Данный подход не может быть прямо применен для случая управляющих алгоритмов реального времени, поскольку для них критически важным является осуществление корректного управления (под которым понимается выдача соответствующих управляющих воздействий) на всем промежутке времени функционирования алгоритма. Более того, для данного случая неприменим, без внесения соответствующих поправок, вообще классический подход к алгоритму как к набору действий для получения *по завершении* его работы определенного результата, т.к. в общем случае алгоритм управления может работать бесконечно, выполняя, тем не менее, возложенные на него функции. Как говорит основоположник использования темпоральной логики в исследовании программ А. Пнуели: «Нужно ли понятие внешнего времени, или темпоральности, для рассуждений о программах?.. для последовательных программ темпоральность не является существенной. Это так потому, что сами эти программы имеют «внутренние часы», а именно, само выполнение. Зная метку в программе и значения программных переменных, мы можем точно определить, в каком месте выполнения мы находимся. Поэтому для таких программ простые временные понятия, как «до» и «после» выполнения программного сегмента... адекватны. Но при обращении к программам индетерминистическим, параллельным, в которых выполнение состоит из перемешанных между собой операций из различных процессов, мы должны различать «где» и «когда», и сохранять внешнюю временную шкалу, независимую от выполнения» [6].

Будем рассматривать алгоритмы и программы дискретного реального времени, имеющие определенное время старта и завершения, то есть такие алгоритмы, функционирование которых синхронизируется некоторыми часами, и моменты времени выполнения действий задаются целыми числами, принадлежащими отрезку $[t_0, t_k]$. Единичное приращение на шкале времени соответствует одному «тику» системных часов $\Delta_{t_{\min}}$.

Сохраняя логический подход, определим семантику работы УА РВ, как успешную выдачу всех необходимых управляющих воздействий (выполнение всех целевых задач) во все требуемые моменты времени, или в обозначениях пред- и постусловий:

$$\{ U(D_0, t_0) \} \text{ } YS \{ B(F_1(t_1), F_2(t_2), \dots F_k(t_k)) \},$$

то есть в момент времени начала функционирования УА t_0 истинно условие корректного задания исходных данных D_0 , включающих программу полета, а к моменту завершения работы t_k управляющего алгоритма истинен предикат B означающий успешное формирование всех управляющих воздействий F_k во все заданные моменты времени $t_1, t_2, \dots t_k$ на протяжении функционирования системы.

Иначе говоря, управляющий алгоритм должен в заданные моменты времени осуществить запуск на выполнение тех или иных функциональных задач. При этом дополнительным уточняющим фактором является то, что, вообще говоря, многие управляющие воздействия должны производиться на протяжении заданного промежутка времени (например, в случае управления космическим аппаратом работа реактивного двигателя должна происходить в течение заданного отрезка времени, чтобы сообщить необходимый импульс движения).

Тогда семантика управляющего алгоритма может быть представлена в виде набора четверок объектов:

$$UAPB = \{ \langle f_i, t_i, \tau_i, \bar{l}_i \rangle, i = \overline{1, N} \},$$

где f_i — функциональная задача (действие), t_i — момент начала исполнения действия (t_i есть неотрицательное целое число), τ_i — длительность действия, $\bar{l}_i$ — логический вектор, обуславливающий действие. Компонентами логического вектора являются логические переменные (ЛП). При этом на логические переменные возлагается сразу несколько функций: во-первых, они отражают текущее состояние параметров внешней среды (включая показания датчиков, отражающих характеристики управляемого объекта); во-вторых, это могут быть предикаты, отражающие те или иные свойства обрабатываемых алгоритмом данных; в-третьих, они также могут являться «флагами» или «признаками», отражающими текущее состояние вычислительного процесса, в том числе, можно сопоставить каждой функциональной задаче связанный с ней предикат, который истинен тогда, когда данная ФЗ выполняется в параллельной вычислительной среде. Предположим, что всего в алгоритме задействовано M логических переменных $\alpha_j, j = \overline{1, M}$. В этом случае размерность логического вектора будет равна $M : \bar{l}_i = (\alpha_1, \alpha_2, \dots \alpha_M)$.

Каждому включению функциональной задачи в некоторый момент времени t_i сопоставим обуславливающий его логический вектор. Если значение некоторой логической переменной должно влиять на выполнение ФЗ, в соответствующей позиции логического вектора записывается значение 1 (ИСТИНА), либо 0 (ЛОЖЬ), если же значение данной ЛП не влияет на запуск функциональной задачи в момент времени t_i , в соответствующей компоненте логического вектора записывается Н (НЕВАЖНО). Таким образом, здесь применяется трехзначная логика.

В наборе среди четверок объектов задаются относительные смещения времени запуска ФЗ на выполнение, то есть присутствует некоторая базовая величина $t_{\text{вА}} = \min (t_i)$, $i = \overline{1, N}$, относительно которой смещения задаются времена старта всех остальных задач. Ее можно принять за нулевую метку времени, или за некоторую целую константу, и таким образом привести весь алгоритм к абсолютной временной шкале. При этом общая длительность исполнения алгоритма будет равна $T = \max (t_i + t_i)$.

В отличие от известных моделей семантик программ реального времени (таймированные автоматы, системы переходов, дискретные системы, системы алгебраических алгебр Глушкова и др.), в которых явным образом предполагается задание отношений передачи управления, предлагаемая семантика не накладывает ограничений на последовательность передач управления и в этом смысле может быть реализована различными программами, имеющими различные логико-временные схемы, или уграфы.

Сделаем следующее замечание. Значение истинности фигурирующей в описании УА логической переменной, в общем случае может меняться на протяжении его исполнения. Поэтому можно ввести в рассмотрение предикат, описывающий истинность значения ЛП α_k в момент времени t_i : $p(\alpha_k, t_i)$.

Предложенная семантика отражает сложную природу управляющего алгоритма РВ, который имеет как логическую, так и временную структуру. В этом смысле данная семантика аналогична логике ветвящегося времени Пнуели, а также расширенным моделям алгебры процессов АСМ с добавлением в них условий реального времени.

На совместное параллельное выполнение функциональных задач обычно накладывается ряд временных ограничений. Физический смысл подобных ограничений может быть пояснен, в частности, такими примерами, как использование результатов выполнения одной ФЗ другой функциональной задачей, или невозможность одновременного выполнения двух ФЗ из-за использования ими обеими одного и того же уникального ресурса.

Выделим следующие варианты согласования выполнения ФЗ.

1. Совпадение по началу $f_1 \text{ CH } f_2$, когда $t_1 = t_2$.
2. Совпадение по концу $f_1 \text{ CK } f_2$, когда $t_1 + \tau_1 = t_2 + \tau_2$.
3. Непосредственное следование $f_1 \rightarrow f_2$, когда $t_1 + \tau_1 = t_2$.
4. Простое предшествование $f_1 < f_2$, когда $t_1 + \tau_1 < t_2$.
5. Запрет на наложение $f_1 < \supset f_2$, когда $(t_1 + \tau_1 < t_2) \vee (t_2 + \tau_2 < t_1)$.

Введем в рассмотрение также задержки, или «временные интервалы» (ВИ), по семантике это «пустые» с точки зрения выполняемых действий функциональные задачи, имеющие при этом некоторую длительность и используемые для согласования во времени исполнения других функциональных задач. Кроме того, возможны функциональные задачи с нулевой длительностью, которые соответствуют по семантике мгновенным событиям (например, выдаче некоторого управляющего импульса на некоторый прибор).

Обуславливание исполнения некоторой ФЗ каким-либо предикатом будем записывать следующим образом:

$$(\alpha_1) \Rightarrow f_1,$$

что означает выполнение f_1 только в том случае, если логическая переменная α_1 принимает значение ИСТИНА (данная запись эквивалентна записи $\alpha_1 \supset f_1$ исчисления высказываний). Возможна также «полная» запись, соответствующая условному оператору языков программирования

$$(\alpha_1) \Rightarrow f_1 + (\neg \alpha_1) \Rightarrow f_2$$

и читающаяся как исполнение ФЗ f_1 в случае ложности α_1 и исполнение f_2 в противном случае.

Данный подход может быть формализован с помощью модифицированного исчисления УА РВ (впервые исчисление как модель формального представления УА РВ была предложена А. А. Калентьевым [8], им были введены символы СН, СК и $\rightarrow$ как знаки для записи операций над управляющими алгоритмами).

Дополним определение нашей формальной теории. Она будет представлять собой эквивалентную спецификацию [6], то есть в ней присутствует эквивалентность (равенство), отражаемая символом $=$, что позволяет записывать уравнения.

Два УА РВ являются семантически эквивалентными, если они включают одни и те же функциональные задачи в те же моменты времени при одних и тех же условиях.

Алфавит образован символами f с численными индексами для обозначения ФЗ, символами α с численными индексами для обозначения логических переменных, а также включает символы логического отрицания $\neg$, символ логического следствия (импликации) $\Rightarrow$, символ логического выбора $+$, символы временных соотношений СН, СК, $\rightarrow$, символ эквивалентности $=$, и круглые скобки. Символы с численными индексами используются в метатеории для обозначения составных термов.

Выражения исчисления УА РВ включают в себя логические выражения, и термы, полученные применением специфических операций. Введем также в язык исчисления возможность подстановки.

Множество термов определяется индуктивно:

1. Если t_1 и t_2 – термы, α – логическая переменная, то

$(t_1), t_1 \text{ CH } t_2, t_1 \text{ CK } t_2, t_1 > t_2, t_1 < t_2, t_1 <> t_2, \alpha \Rightarrow t_1, \neg \alpha \Rightarrow t_1, (\alpha) \Rightarrow t_1 + (\neg \alpha) \Rightarrow t_2$ — тоже термы.

Аксиомы модифицированного исчисления УА РВ включают в себя нижеследующие схемы аксиом (t_1, t_2 – термы, α – логическая переменная).

Аксиома равенства

A0. $t_1 = t_1$.

Аксиомы идемпотентности

A1. $t_1 \text{ CH } t_1$

A2. $t_1 \text{ CK } t_1$

Аксиомы коммутативности

A3. $t_1 \text{ CH } t_2 = t_2 \text{ CH } t_1$

A4. $t_1 \text{ CK } t_2 = t_2 \text{ CK } t_1$

A5. $(\alpha) \Rightarrow t_1 + (\neg \alpha) \Rightarrow t_2 = (\neg \alpha) \Rightarrow t_2 + (\alpha) \Rightarrow t_1$

Аксиомы ассоциативности

A6. $(t_1 \text{ CH } t_2) \text{ CH } t_3 = t_1 \text{ CH } (t_2 \text{ CH } t_3)$

A7. $(t_1 \text{ CK } t_2) \text{ CK } t_3 = t_1 \text{ CK } (t_2 \text{ CK } t_3)$

A8. $(t_1 \rightarrow t_2) \rightarrow t_3 = t_1 \rightarrow (t_2 \rightarrow t_3)$

A9. $(t_1 \rightarrow t_2) \rightarrow t_3 = t_1 \rightarrow (t_2 \rightarrow t_3)$

Аксиомы дистрибутивности

A10. $(t_1 \rightarrow t_2) \text{ CH } (t_1 \rightarrow t_3) = t_1 > (t_2 \text{ CH } t_3)$

A11. $(t_1 \rightarrow t_2) \text{ CK } (t_3 \rightarrow t_2) = (t_1 \text{ CK } t_3) \rightarrow t_2$

A12. $(\alpha) \Rightarrow (t_1 \rightarrow t_2) + (\neg \alpha) \Rightarrow (t_1 \rightarrow t_3) = t_1 \rightarrow ((\alpha) \Rightarrow t_2 + (\neg \alpha) \Rightarrow t_3)$

A12. $(\alpha) \Rightarrow (t_1 \rightarrow t_2) + (\neg \alpha) \Rightarrow (t_3 \rightarrow t_2) = ((\alpha) \Rightarrow t_1 + (\neg \alpha) \Rightarrow t_3) \rightarrow t_2$

A14. $(\alpha) \Rightarrow (t_1 \text{ CH } t_2) + (\neg \alpha) \Rightarrow (t_1 \text{ CH } t_3) = t_1 \text{ CH } ((\alpha) \Rightarrow t_2 + (\neg \alpha) \Rightarrow t_3)$

A15. $(\alpha) \Rightarrow (t_1 \text{ CH } t_2) + (\neg \alpha) \Rightarrow (t_3 \text{ CH } t_2) = ((\alpha) \Rightarrow t_1 + (\neg \alpha) \Rightarrow t_3) \text{ CH } t_2$

A16. $(\alpha) \Rightarrow (t_1 \text{ CH } t_2) + (\neg \alpha) \Rightarrow (t_1 \text{ CH } t_3) = t_1 \text{ CH } ((\alpha) \Rightarrow t_2 + (\neg \alpha) \Rightarrow t_3)$

A17. $(\alpha) \Rightarrow (t_1 \text{ CK } t_2) + (\neg \alpha) \Rightarrow (t_3 \text{ CK } t_2) = ((\alpha) \Rightarrow t_1 + (\neg \alpha) \Rightarrow t_3) \text{ CK } t_2$

A18. $(\alpha) \Rightarrow (t_1 \rightarrow t_2) = ((\alpha) \Rightarrow t_1) \rightarrow ((\alpha) \Rightarrow t_2)$

A19. $(\alpha) \Rightarrow (t_1 \text{ CH } t_2) = ((\alpha) \Rightarrow t_1) \text{ CH } ((\alpha) \Rightarrow t_2)$

A20. $(\alpha) \Rightarrow (t_1 \text{ CK } t_2) = ((\alpha) \Rightarrow t_1) \text{ CK } ((\alpha) \Rightarrow t_2)$

Правила вывода включают следующие:

1. Если в схемах аксиом присутствует равенство $t_1 = t_2$, то выводимы также $\Theta(t_1) = \Theta(t_2)$, где Θ — произвольная подстановка (правило подстановки).
2. Если выводимы $t_1 = t_2$ и $t_2 = t_3$, то выводимо также $t_1 = t_3$.
3. Если выводимо $t_1 = t_2$, то выводимо также $t_2 = t_1$.
4. Если выводимы $t_1 < t_2$ и $t_2 < t_3$, то выводимо также $t_1 < t_3$.
5. Если выводимы $t_1 <> t_2$ и $t_2 <> t_3$, то выводимо также $t_1 <> t_3$.

Если распространить на логические переменные возможность применения к ним стандартных операций логики предикатов и соответственно добавить к ней аксиомы и правила вывода исчисления предикатов, то можно привести данную теорию к теории первого порядка.

В качестве интерпретации изложенной формальной теории можно использовать многоосновную алгебру УА РВ, имеющую в качестве множеств-носителей множество наборов четверок, описывающих семантику УА, и множество логических условий. При этом на множестве наборов четверок определяются операции во временном пространстве — совпадение по началу (СН), совпадение по концу (СК), непосредственное следование ($\rightarrow$), на множестве логических условий — операции трехзначной логики согласно нижеследующей таблице истинности.

Вводится также операция обуславливания выполнения управляющего алгоритма истинностью или ложностью некоторого условия ($\Rightarrow$). На множестве наборов четверок объектов в вышеуказанном смысле определяются также отношения (предикаты) неналожения $<>$ и временного предшествования $<$.

Приведем некоторые примеры интерпретации операций и соотношений формального исчисления УА РВ, в алгебраической системе УА РВ.

x	y	$x \vee y$ (ИЛИ)	$x \wedge y$ (И)	$\neg x$ (НЕ)
0	0	0	0	1
1	1	1	1	0
Н	Н	Н	Н	Н
0	1	1	0	1
1	0	1	0	0
0	Н	Н	0	1
1	Н	1	1	0
Н	1	1	Н	Н
Н	0	Н	0	Н

Пример 1. Операция СН (совпадение по началу). Пусть $t_1 = \{ \langle f_1, t_1 = t_{\min t_1}, 20, \bar{l}_1 \rangle, \langle f_3, t_3 = t_1 + 50, 100, \bar{l}_3 \rangle \}$, две ФЗ в УА, длительность 100 $t_2 = \{ \langle f_2, t_2 = t_{\min t_2}, 60, \bar{l}_2 \rangle \}$, одна ФЗ в УА, длительность 60.

Тогда $t_3 = t_1 \text{ СН } t_2$ будет $t_3 = \{ \langle f_1, t_{\text{pez}} = t_{\min t_3}, 20, \bar{l}_1 \rangle, \langle f_2, t_{\text{pez}} = t_{\min t_3}, 40, \bar{l}_2 \rangle, \langle f_3, t_{\text{pez}} + 50, 100, \bar{l}_3 \rangle \}$, три ФЗ в УА, длительность 100.

Пример 2. Операция СК (совпадение по концу). Пусть $t_1 = \{ \langle f_1, t_1 = t_{\min t_1}, 20, \bar{l}_1 \rangle, \langle f_3, t_3 = t_1 + 50, 100, \bar{l}_3 \rangle \}$, $t_2 = \{ \langle f_2, t_2 = t_{\min t_2}, 60, \bar{l}_2 \rangle \}$.

Тогда $t_3 = t_1 \text{ СК } t_2$ будет $t_3 = \{ \langle f_1, t_1 = t_{\min t_3}, 20, \bar{l}_1 \rangle, \langle f_2, t_2 = t_1 + 90, 60, \bar{l}_2 \rangle, \langle f_3, t_3 = t_1 + 50, 100, \bar{l}_3 \rangle \}$. (длительность $50+100=90+60=150$).

Пример 3. Операция $\Rightarrow$ (логическое обуславливание). Пусть

$$t_1 = \{ \langle f_1, t_1 = t_{\min t_1}, 20, \bar{l}_1 = (\alpha_1 = 1, \alpha_2 = \text{Н}, \alpha_3 = 0) \rangle, \langle f_3, t_3 = t_1 + 50, 100, \bar{l}_3 = (\alpha_1 = \text{Н}, \alpha_2 = \text{Н}, \alpha_3 = \text{Н}) \rangle \}.$$

Тогда $t_3 = (\neg \alpha_2) \Rightarrow t_1$ будет $t_3 = \{ \langle f_1, t_1 = t_{\min t_3}, 20, \bar{l}_1 = (\alpha_1 = 1, \alpha_2 = 0, \alpha_3 = 0) \rangle, \langle f_3, t_3 = t_1 + 50, 100, \bar{l}_3 = (\alpha_1 = \text{Н}, \alpha_2 = 0, \alpha_3 = \text{Н}) \rangle \}$.

Задача построения (генерации) управляющей программы, реализующей управляющий алгоритм с заданной семантикой, сводится к построению логико-временной схемы функционирования, то есть в терминах теории схем программ — схемы программы, но с учетом аспекта реального времени. Будем считать схемой программы реального времени ориентированный взвешенный граф, в котором дуги без пометок соответствуют простым передачам управления, а дуги, помеченные положительным целым числом — передачам управления с соответствующей задержкой по времени. Одна из вершин выделяется в качестве начальной (стартовой) вершины, как и в обычном уграфе. При этом должно выполняться следующее условие корректности: в случае, если имеется несколько путей от стартовой вершины до некоторой вершины графа,

суммы всех чисел из пометок на дугах во всех путях, ведущих от стартовой вершины к данной вершине, должны совпадать между собой. Назовем величину такой суммы *текущим временем* выполнения данной вершины.

При этом можно осуществить операцию группировки вершин по одинаковой величине текущего времени. Такое разбиение задает на множестве вершин управляющего графа отношение эквивалентности. Назовем группу вершин с одинаковым текущим временем *входом* (включением) управляющего алгоритма реального времени [9].

В данной работе принято называть такую схему *многовходовой моделью* управляющего алгоритма.

В наиболее простом случае число включений равно просто количеству отличающихся друг от друга моментов времени t_i , содержащихся в семантике УА, и соответствующих моментам запуска входящих в управляющий алгоритм отдельных функциональных задач f_i .

Соединенная передачами управления между входами совокупность входов образует граф:

$$MWM = \langle W, U \rangle,$$

где W — множество вершин (входов); U — множество дуг (передач управления между входами), т.е. бинарное отношение $W \times W$.

В случае ациклического графа, наиболее распространенного варианта, должно выполняться условие последовательного запуска при передаче управления: если существует дуга (W_i, W_j) , то $t_{0W_i} < t_{0W_j}$, где t_{0W_i} — момент включения входа W_i , t_{0W_j} — момент включения входа W_j .

Каждый вход может быть представлен, в свою очередь, логической схемой (последовательностью проверок условий α_k и действий f_i , реализующих логику входа).

Внутреннюю структуру входа можно рассматривать как совокупность линейных участков алгоритма и ветвлений по результатам проверки логических условий.

Линейным участком будем называть последовательность функциональных задач без проверок логических условий и передач управления. Линейный участок может включать несколько f_i , или не одной.

Тогда логическая схема входа будет представлять собой граф (дерево):

$$W = \langle LU, Y \rangle,$$

где LU — множество вершин графа (линейных участков), Y — множество дуг — передач управления (ветвлений).

В случае внутренней структуры входа граф рассматривается уже не как взвешенный, то есть все передачи управления происходят без временной задержки.

Каждая вершина логической схемы входа (помимо листьев, соответствующих заключительным линейным участкам входа), будет иметь две исходящих из нее дуги, соответствующих вариантам выполнения и невыполнения логического условия, то есть дерево будет бинарным.

Вариант взаимосвязи линейных участков внутри входа представлен на рисунке.

Логические условия, фигурирующие в многовходовой модели, по времени их значимости (актуальности) могут относиться к одному из трех типов: $\alpha(t)$, то есть нас интересует значение логической переменной в текущий момент времени t , и с течением времени значение α может меняться; $\alpha(t_0)$, когда значение переменной может меняться, но нас интересует ее значение в определенный момент времени t_0 (часто это момент начала исполнения УА) и последующие изменения значения во времени не должны влиять на исполнение ПКФ; α_{const} , когда значение логической переменной не претерпевает изменений на всем промежутке исполнения УА.

Взаимосвязь линейных участков

Очевидно, что подход при интерпретации этих трех типов логических переменных при построении многовходовой модели должен быть разным. Для первого случая $\alpha(t)$ необходима проверка актуального значения в каждый текущий момент, то есть внутри каждого входа при обусловленности некой f_i логическим условием оно должно обязательно проверяться.

Для случая $\alpha(t_0)$ необходимо один раз в начале выполнения УА выяснить, к какому варианту необходимо прибегнуть, то есть один раз в момент времени t_0 необходимо проверить все логические условия, фигурирующие во всех логических векторах, обуславливающих выполнение всех f_i , входящих в данный УА.

Для случая статической $\alpha_{\text{стат}}$ проверку условия можно проводить в любой удобный момент, и соответственно становится возможным гибким образом распределить проверки логических условий между входами и организовать передачи управления между входами в многовходовой модели.

Таким образом, для случаев $\alpha(t_0)$ и $\alpha_{\text{стат}}$ можно считать, что от входа к входу логическая обусловленность «наследуется», и образуется цепочка входов, логически обусловленная аналогично логической обусловленности отдельной функциональной задачи f_i в модели семантики УА. При этом распределенная во времени цепочка входов является аналогом логической ветви исполнения обычного алгоритма, не выполняющегося в реальном времени, или одной из ветвей времени в логике ветвящегося времени Пнуели.

С учетом вышеприведенных особенностей, можно построить алгоритм синтеза конкретной логико-временной схемы управляющего алгоритма, реализующей необходимую семантику, в которой сводится к минимуму количество проверок логических условий в разные моменты включения (входы) УА РВ. По построенной логико-временной схеме и описаниям отдельных функциональных задач затем открывается возможность построения конкретной управляющей программы.

Описанные математические модели поддерживаются технологией визуального конструирования управляющих алгоритмов реального времени и реализующей ее инструментальной программной системой, функционирующей на платформе Windows 95/98/2000/XP (разработанной в основном средствами языка C/C++).

При этом на входе системы человек-проектировщик, исходя из материалов по логике управления, которую должен реализовывать алгоритм, осуществляет интерактивное «конструирование» визуального образа управляющего алгоритма в виде циклограммы.

Выходными данными инструментальной программной системы являются техническая документация на УА РВ, а также собственно текст управляющей программы на целевом языке программирования.

Таким образом, решается задача автоматического синтеза программы по спецификации ее характеристик.

Важными задачами при проведении отладки управляющих алгоритмов реального времени являются автоматизированное выявление всех возможных вариантов исполнения («маршрутов») алгоритма в зависимости от значений компонент вектора логических переменных, а также выявление всех имеющихся в алгоритме информационных и управляющих связей с другими управляющими программами.

Созданная система позволяет на базе внутренних структур данных, отражающих многовходовую модель УА РВ, решать эти задачи.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Управление космическими аппаратами зондирования Земли: Компьютерные технологии / Д. И. Козлов, Г. П. Анишаков, Я. А. Мостовой, А. В. Соллогуб. М.: Машиностроение, 1998. 328 с.
2. Логика и компьютер. Моделирование рассуждений и проверка правильности программ / А. М. Анисов, П. И. Быстров, В. А. Смирнов и др. М: Наука, 1990. 242 с.
3. Советов Б. Я., Яковлев С. А. Моделирование систем. М.: Высш. шк., 2001. 322 с.
4. Ben-Ari M., Manna Z., Pnueli A.: The Temporal Logic of Branching Time. Proc. 8th Annual Symposium on Principles of Programming Languages, 1981, ACM Press, Williamsburg. Springer-Verlag, 1992. P. 164-176.
5. Baeten J. C. M., Bergstra J. A. Real time Process Algebra. Formal Aspects of Computing, Vol. 3, 1991. P.142-188.
6. Касьянов В. Н., Евстигнеев В. А. Графы в программировании: обработка, визуализация и применение. СПб.: БХВ-Петербург, 2003. 262 с.
7. Семантика языков программирования / Сб. статей под ред. К. Г. Батаева. М.: Мир, 1979. 223 с.
8. Калентьев А. А. Автоматизированный синтез алгоритмов асинхронного управления техническими системами с множеством дискретных состояний / Сам. гос. аэрокосм. ун-т. Самара, 1998. 204 с.
9. Тюгашев А. А. Проблема неоднозначности при порождении логико-временной структуры управляющего алгоритма по многовходовой модели реального времени // Сб. трудов Третьей международной молодежной школы-семинара БИКАМП-01, СПб, 2001. С. 179.

Поступила 12.05.2005 г.