РОССИЙСКАЯ АКАДЕМИЯ НАУК

САНКТ-ПЕТЕРБУРГСКИЙ ИНСТИТУТ
ИНФОРМАТИКИ И АВТОМАТИЗАЦИИ РАН

С.В. МИКОНИ

ОБЩИЕ ДИАГНОСТИЧЕСКИЕ БАЗЫ ЗНАНИЙ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

САНКТ-ПЕТЕРБУРГ
1992

РОССИЙСКАЯ АКАДЕМИЯ НАУК

САНКТ-ПЕТЕРБУРГСКИЙ ИНСТИТУТ ИНФОРМАТИКИ И АВТОМАТИЗАЦИИ РАН

С.В. МИКОНИ

ОБЩИЕ ДИАГНОСТИЧЕСКИЕ БАЗЫ ЗНАНИЙ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

САНКТ-ПЕТЕРБУРГ
1992

УДК 681.1

Микони С. В. Общие диагностические базы знаний вычислительных систем, СПб.: СПИИРАН. 1992. 234 с.

В монографии рассматриваются основные составляющие общего диагностического обеспечения вычислительных систем – понятия, модели и методы. Излагается общий подход к их упорядочению и машинному представлению, основанный па использовании аксиоматического метода и теории формальных систем. Представлены системы понятий, общих диагностических моделей ВС и методов диагностирования. Приводятся примеры, интерпретирующие общие модели и методы. Применена технология баз знаний для представления всех видов диагностического обеспечения. Последнее представляется в ЭВМ в виде семейства интеллектуальных справочников. В качестве иллюстрации подхода подробно описывается интеллектуальный справочник по методам диагностирования вычислительных сетей и его программная реализация.

Монография представляет интерес для специалистов в области надёжности и диагностирования вычислительных систем, аспирантов, научных сотрудников и преподавателей соответствующих специальностей. Она может быть также полезна специалистам, интересующимся применением идей искусственного интеллекта в технических дисциплинах. Библиогр.134 назв. Ил. 44. Табл. 14.

Рецензенты:
доктор технических наук профессор В.В. Сапожников
доктор технических наук профессор В.В. Барашенков

150400000,

М ---------- , без объявления

042(02)-92

ISBN 5-201-10182-8

THE RUSSIAN ACADEMY OF SCIENCES

ST. PETERSBURG INSTITUTE FOR INFORMATICS
AND AUTOMATION OF THE RAS

S.V. MIKONI

GENERAL DIAGNOSTIC KNOWLEDGE BASE OF COMPUTING SYSTEMS

ST. PETERSBURG
1992

Mikoni S.V. General diagnostic knowledge base of computing systems. SPb. SPIIRAS. 1992. 234 p.

The monograph examines the main components of the general diagnostic support of computing systems – concepts, models and methods. A general approach to their ordering and computer representation based on the use of the axiomatic method and the theory of formal systems is presented. The system of concepts, general diagnostic models and diagnostic methods are presented. Examples are given which interpret general models and methods. The knowledge base technology has been applied to represent all types of diagnostic software. The latter is represented in a computer as a family of intellectual reference books. As an illustration of the approach, the intellectual reference book on methods for diagnosing computer networks and its software implementation are described in detail.

The monograph is of interest to specialists in the field of reliability and diagnosis of computing systems, graduate students, researchers and teachers of relevant specialties. It may also be useful to specialists interested in applying the ideas of artificial intelligence in technical disciplines. References 134. Figers 44. Table 14.

Reviewers:
Doctor of Technical Sciences, Professor V.V. Sapojnikov
Doctor of Technical Sciences, Professor V.V. Barashenkov

ВВЕДЕНИЕ

Задачами любой научной дисциплины являются упорядочение, хранение и передача знания. С развитием вычислительной техники и информатики появилась возможность автоматизировать эти процессы, что многократно усиливает их интенсивность. Автоматизации предшествуют два необходимых этапа исследования предметной области –систематизация и формализация объектов ей изучения.

В монографии рассматриваются все 3 этапа разработки баз знаний применительно к конкретной предметной области – технического диагностирования вычислительных систем (ВС). Специфика рассматриваемых баз знаний определяется выделенной актуальной подобластью – разработкой общего *диагностического обеспечения* (ДО) ВС.

Под ДО ВС понимается комплекс взаимоувязанных правил, методов, алгоритмов и средств, необходимых для осуществления диагностирования на всех стадиях жизненного цикле ВС. Признак «общее» выделяет инвариантную относительно различных стадий жизненного цикла ВС часть ДО. Она представляется системой понятий, общими диагностическими моделями и методами диагностирования и является метасистемой по отношению к конкретным моделям и методам.

Этапы разработки общего ДО, выполняемые при проектировании баз знаний, требуют глубокого проникновения в основы технической диагностики с целью создания полной и непротиворечивой системы знаний. Решению последней проблемы были посвящены работы К.Б. Карандеева, А.В. Мозгалевского, П.П. Пархоменко, В.А. Гуляева, В.П. Калинина и других исследователей. Однако по мере развития технической диагностики возникают новые противоречия, разрешение которых является стимулом по совершенствованию её системы знания.

Наиболее очевидным отношением, в котором находятся между собой понятия, диагностические модели и методы диагностирования ВС является отношение *толерантности*. Известен эмпирический подход установления отношения толерантности, основанный на нейронной парадигме. Однако получаемые отношения имеют вероятностный характер. Детерминированный подход к их определению основан на выявлении базовых элементов знания с последующим их комбинированием в производные элементы. Этот подход требует стратификации зданий по уровням общности, упорядочения их внутри уровней и между ними.

Он принят за основу формирования предметной области «техническое диагностирование ВС».

С позиций экономичности и полноты знаний целесообразно хранить в базах не сами знания, а их признаки. Это даёт возможность генерировать различные *варианты* знания и принимать те или иные варианты в зависимости от конкретных условий. Такой путь стал возможным благодаря существенному повышению производительности ЭВМ, в том числе персональных. Он позволяет реализовать машинное представление общего диагностического обеспечения ВС в виде семейства интеллектуальных справочников.

Монография состоит из восьми глав и заключения.

В первой главе даётся краткое описание составляющих общего ДО ВС – систем понятий, диагностических моделей и методов диагностирования с целью выявления их единства и противоречий. На основе этого обосновывается возможность и необходимость их систематизации с применением формальных моделей и методов. Глава рассчитана на читателя, знакомого с основами технического диагностирования ВС. Для ознакомления с ними даются ссылки на литературу.

Вторая глава посвящается изложения теории порождения понятий, которая используется в качестве методологической основы систематизации всех аспектов предметной области. В силу прикладного характера главы и последующей иллюстрации применения её положений строгие доказательства последних опущены.

В третьей главе приводится система понятий технического диагностирования ВС. Поскольку понятия предметной области и их определения (дефиниции) выражаются вербально, внимание читателя акцентируется на использовании естественно-ориентированного формализованного языка для установления отношения между понятиями по их определениям.

В четвертой гласе описывается система диагностических моделей ВС, базирующаяся на языке первого порядка математической логики. Теоретико-множественное представление моделей позволила применить теорию порождения понятия для нахождения отношений между различными применяемыми на практике моделями.

В пятой главе приводится система методов технического диагностирования ВС. Показано, как на основе функционального базиса

порождаются различные методы диагностирования. Приведена иерархическая модель системы методов.

В шестой главе описывается ряд оригинальных конструктивных моделей и методов ВС, интерпретирующих общие модели и методы.

Седьмая глава посвящена построению баз знаний процедурного типа, ориентированных на формальное порождение различных вариантов понятий, моделей и методов. По своему назначению они представляют собой универсальные базы данных, а по технологии представления данных относятся к классу баз знаний.

В восьмой главе подробно рассматривается процесс проектирования базы знания применительно к методам диагностирования вычислительных сетей.

В заключении обсуждаются проблемы, затронутые в монографии.

Монография является итогом многолетних исследований автора в области технического диагностирования ВС и его интеллектуализации. Благоприятной средой для этих исследований послужили многочисленные школы, семинары и совещания по технической диагностике, проводимые с 1969 года под научным руководством члена-корреспондента РАН П.П. Пархоменко. Личная переписка с последним позволила автору уточнить взгляда по отдельным вопросам технической диагностики. Автор выражает благодарность ему и коллегам по научным дискуссиям.

Работа обобщена и написана в Санкт-Петербургском институте информатики и автоматизации Российской Академии наук. Раздел 6.2 написан совместно с А.В. Дубровским. На взгляды автора повлияло общение с профессорами Н.Н. Ляшенко, А.О. Слисенко, В.А. Торгашёвым, В.И. Городецким, за что он выражает им свою признательность. Автор благодарен за участие и помощь в работе коллегам, в особенности к.т.н. В.П. Иванову, к.ф-м.н. В.М. Нестерову и И.В. Цареву.

Автор глубоко благодарен академику Санкт-Петербургской Инженерной Академии Р.М. Юсупову за неизменную помощь и поддержу в работе над монографией.

Улучшению качества книги в немалой степени способствовали замечания рецензентов – профессоров В.В. Сапожникова и В.В. Барашенкова, за что автор выражает им свою признательность.

Глава 1. ДИАГНОСТИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Общее диагностическое обеспечение вычислительных систем включает следующие составляющие: понятия, диагностические модели и методы диагностирования. С различной степенью подробности они рассматривались в [1-8].

Основное влияние на общее ДО современных ВС оказывают два фактора – развитие архитектуры ВС и порождаемые им новые возможности организации отказоустойчивых вычислений [9-15]. В качестве примера развития ВС сошлёмся на машину с динамической архитектурой (МДА), разработанную В.А. Торгашёвым и описанную в [16, 17]. Такие её свойства, как автономное функционирование вычислительных модулей (ВМ), возможность их взаимодействия по различным путям и в произвольные моменты времени, наличие сетевых команд и механизма их очередей в ВМ, наличие индикаторного поля, постоянно отслеживающего техническое состояние ВМ, возможность изоляции неисправных ВМ в процессе вычислений – значительно расширяют разнообразие методов и средств, применяемых для диагностирования ВС, что актуализирует разработку их баз данных и знаний.

Ниже кратко описываются составляющие общего ДО ВС как объекты последующей систематизации и машинного представления.

1.1. Система понятий технического диагностирования ВС

Система понятий образует основу научного знания. Она включает философскую, общенаучную и предметную составляющие, различающиеся степенью общности. Отношения между понятиями устанавливаются через определения (дефиниции) понятий. Последние, как правило, задаются вербально. В качестве универсального языка определения понятий используется естественный язык, ибо «именно к нему сводимы все искусственные языки, создаваемые людьми для себя и для компьютеров» [18]. Естественный язык отражает лингвистическую компоненту системы понятий, а связи между понятиями – логическую компоненту. Вместе они представляют терминологическую систему (терминосистему) области знания [19].

Любая система понятий должна удовлетворить требованиям полноты и непротиворечивости. Первое из них носит относительный характер,

а второе – абсолютный характер. Однако практически любой эвристически созданной системе понятий присущи *неполнота* и *противоречивость*. Источниками последней являются неправильные связи между некоторыми понятиями внутри системы и связи с понятиями более общих и смежных терминосистем. Некоторые термины терминосистемы являются *ложно ориентирующими*. В качестве примеров рассмотрим ряд понятий технической диагностики и надежности, чья неправильная трактовка в государственных терминологических стандартах влечет противоречия в системах понятий.

1. **Техническое состояние**. Определенное в [126] как «совокупность свойств объекта, характеризуемая … признаками, установленными технической документацией на этот объект» оно несёт в себе неявный смысл сопоставления свойств объекта с признаками, содержащимися в документации. Но результатом такого сопоставления является оцененное техническое состояние, т.е. его вид, например, исправное или неисправное состояние. Между тем, свойства объективно присущи техническому объекту независимо от того, охарактеризованы они извне или нет. Наличие документации лишь обеспечивает *проверяемость* этих свойств.

2. **Техническое диагностирование**. Оно определенно в [127] как «процесс определения технического состояния объекта диагностирования с определенной точностью». В примечании 1 к нему указывается, что «результатом диагностирования является заключение о техническом состоянии объекта с указанием, при необходимости, места, вида и причины дефекта». Определение и примечание находятся в логическом противоречии. Первое характеризует диагностирование как несамостоятельный процесс, являющийся *частью* процесса контроля технического состояния (согласно справочному приложению к [127]). А второе характеризует диагностирование как *самостоятельный* процесс, завершающийся нахождением дефекта, т.е. постановкой технического диагноза так же, как и контроль технического состояния.

3. **Неисправность и дефект**. Неисправность определяется в [128] как «состояние объекта, при котором он не соответствует хотя бы одному из требований НТД». Дефект определен в [129] как «каждое отдельное несоответствие свойств изделия требованиям НТД». Столь малое различие определений влечет весьма произвольные толкования связей

между этими понятиями.

4. **Тест диагностирования**. Он определен в [127] как «одно или несколько тестовых воздействий и последовательность их выполнения, обеспечивающие диагностирование». Это определение не имеет чётко выраженной цели, поскольку последняя выражается через неоднозначно неопределенное там же понятие «диагностирование».

Очевидная цель применения теста к объекту диагностирования (ОД) – постановка диагноза. Но последний не может быть поставлен без знания ожидаемых реакций объекта на тестовые воздействия. Именно, они и характеризуют состояние объекта – исправное или неисправное и посему должны включаться в тест.

5. **Функциональное диагностирование**. Термин образован в нарушение принципа деления понятий, установленного в формальной логике. Видовые понятия, образуемые на основе родового понятия «диагностирование» по основанию деления «вид воздействия» – *тестовое* или *рабочее* – должны различаться этими противоположными признаками как «тестовое» или «рабочее диагностирование». Однако второй термин неправомерно замещен термином «функциональное диагностирование», в котором признаку «функциональное» придан неоправданно узкий смысл функционирования объекта только по назначению. Между тем объект функционирует (реализует свои функции) и на тестовых воздействиях. Признак «функциональное» внёс путаницу этого термина с термином «функциональный контроль», ибо в [127] диагностирование рассматривается, как часть контроля. По этой логике функциональный контроль реализуется так же, как и функциональное диагностирование в процессе функционирования объекта по назначению с использованием дополнительных ресурсов. Однако в электронной технике термин «функциональный контроль» имеет смысл контроля *функций* в отличие от параметрического контроля, предназначенного для контроля *параметров* микросхем.

6. **Функциональный контроль и контроль функционирования**. Эти термины являются объектом постоянной путаницы. Между тем согласно правилам русской грамматики [75] первый термин характеризует контроль функций ОД как его специфических *свойств*, а второй термин контроль *процесса* реализации этих функций. Иными словами, функциональный контроль определяет *потенциальную*, а контроль

функционирования – *актуальную* готовность объекта к реализации его функций.

7. **Испытание**. Оно трактуется в [130] как более общее понятие по отношению к измерению и контролю, так как последние являются «способами проведения испытаний». Совершенно очевидна ситуативность этой каузативной (причинно-следственной) связи между названными понятиями, ибо известно также и обратное отношение – применение испытаний, как способа контроля надежности в производстве изделий.

Анализ приведённых понятий технической диагностики и надежности позволяет сделать вывод об эвристическом подходе к формированию понятий этих дисциплин [20]. При этом совершаются как логические, так и лингвистические ошибки.

Неправильные связи между понятиями отражают неправильные классификации, как правило, отсутствующие в явном виде. Между тем «классификации являются мощным средством познаваемости любой научной дисциплины, поскольку они решают такую проблему теории познания, как выявление сущностей и установление отношений между ними, а также выяснение первичности сущностей и сведение их к более фундаментальным» [21]. При составлении классификаций совершаются следующие ошибки: несоблюдение правил формальной логики, смешение реальных объектов с их моделями, неправильный выбор оснований деления. Эти ошибки выражаются через неправильные определения понятий.

К лингвистическим ошибкам относится игнорирование основного смысла слов естественного языка, используемых в качестве технических терминов и незнание роли слов в словосочетаниях. Они проявляются в неправильном терминотворчестве.

Система понятий технического диагностирования ВС, расширяющая систему понятий технической диагностики, наследует недостатки последней. Вместе с тем существует тенденция новые, более частные понятия, выражать получившими широкое распространение общими терминами, в результате чего первые утрачивают свой специфический смысл, выражаются огрублённо, а порой и искажённо.

Все отмеченные недостатки терминосистемы технического диагностирования ВС препятствуют развитию языка предметной области,

ухудшая не только её познавательный аспект, но и затрудняя разработку диагностического обеспечения ВС.

1.2. Диагностические модели вычислительных систем

По степени общности диагностические модели ВС делятся на конкретные (конструктивные) и абстрактные. Первые предназначаются непосредственно для проектирования диагностических процедур и тестов. По математической форме представления различают алгебраические, графовые и табличные (матричные) модели.

Нижний логический уровень ВС составляет логические схемы без обратных связей (комбинационные схем) и с обратными связями (последовательностные схемы).

Функциональное назначение комбинационной схемы отражает булева функция, выраженная в дизъюнктивной или конъюнктивной нормальной форме. Функцию и структуру комбинационной схемы отражают следующие диагностические модели:

1) скобочная форма булевой функции [22];
2) система булевых разностей (производных) [23, 24];
3) эквивалентная нормальная форма булевой функции (ЭНФ) [25, 26];
4) логическая сеть [27];
5) контактный двухполюсник [28, 29];
6) альтернативный граф [30];
7) матрица D-кубов [31];
8) матрица контролирующих цепочек [32].

Первые три вида моделей выражены в алгебраической форме, вторые три – в графовой, а два последних – в матричной. Все они естественные образом подходят для выражения константной неисправности, как логической модели дефекта аппаратуры, а при определенной модернизации – для других моделей дефектов [33-36].

Эти же модели берутся за основу для построения диагностических моделей последовательностных схем. С этой целью в них дополнительно отражаются обратные связи. В качестве примера приведем итеративную сеть [37]. Она представляет собой последовательное соединение нескольких копий логической сети, описывающей последовательностную

схему с разорванными обратными связями. Выходы обратных связей предшествующей копии соединяются с входами обратных связей последующей копии, причём число копий определяется числом внутренних состояний, необходимых для проявления неисправности на контролируемом выходе схемы.

При большом числе обратных связей диагностические модели, отражающие структуру последовательностной схемы, становятся громоздкими. Для устранения этого недостатка диагностической моделью отражают только функцию последовательностной схемы. Этому назначению соответствует модель конечного автомата [38] и вход-выходные комплексы [39].

Средний уровень описания ВС составляют микропрограммно-управляемые устройства. Для их моделирования и построения тестов применяются различные разновидности [40, 41] модели регистровых передач [42], отражающей управление потоками данных между регистрами микропроцессора.

Верхний уровень ВС представляют процессоры, вычислительные модули и построенные на их основе вычислительные сети.

Для описания систем с параллельно функционирующими и асинхронно взаимодействующими компонентами применяются сетевые модели, среди которых наибольшее распространение получили сети Петри [43]. Наличие у этой модели специального математического аппарата позволяет использовать её помимо моделирования функционирования сетей для общего и специального анализа их корректности. Первый заключается в проверке свойств *живости*, *ограниченности*, *устойчивости*, *детерминированности*, а второй – в проверке свойств *консервативности*, *последовательности*, *структурной ограниченности* и *активности*. Свойство *достижимости* между маркировками сети Петри позволяет использовать её в качестве диагностической модели для поиска отказа в системе.

Изложенные модели не исчерпывают всего многообразия диагностических моделей компонентов ВС. Существует большое количество их модификаций, различающихся между собой языком описания, составом отражаемых свойств, ограничениями на сложность модулируемого объекта, перечнем предполагаемых неисправностей и т.д. В этом плане разнообразие диагностических моделей безгранично,

и всегда можно найти такое сочетание перечисленных факторов, которое еще не являлось предметом исследования и может потребоваться на практике.

Наряду с моделированием различных уровней описания ВС важную роль для системного анализа играет моделирование *взаимодействия* уровней. Для его описания более приемлемы абстрактные модели, не отражающие излишних подробностей об уровнях. В их качестве используются многосортные теоретико-множественные модели. Сорт символа соответствует уровню системы. Такого рода модели, основанные на языке первого порядка математической логики, были применены в частности, для описания микропроцессорных БИС (МП БИС) [44, 45].

Согласно [45] многоуровневая «модель МП БИС представляет собой совокупность более детальных описаний, поэтому её языковые конструкции определяются включением: $L = L_{АУ}\,(L_{ФУ}(L_{ВУ}))$. Здесь АУ, ФУ, ВУ – соответственно *алгоритмический*, *функциональный* и *вентильный* уровни рассмотрения МП БИС. Для их описания предлагаются символы многосортного языка первого порядка

$$<X, A, Y, F_{\Delta}, F_{\wedge}, P>,$$

где X – множество входов МП БИС;

A – структурный универсум системы, вместо которого часто используется абстрактный универсум Q;

Y – множество выходов;

F_{Δ}, $F_{\wedge}$ – функции переходов и выходов МП БИС;

P – множество предикатных символов».

Сопоставляя эту модель с её модификацией и расшифровкой [44], а также с моделью конечного автомата [47], нетрудно сделать вывод о совпадении первых пяти символов с символами теоретико-множественной модели конечного автомата (КА). Различие заключается лишь в интерпретации символов X, Y и Q множествами предметных переменных. Такая модель ориентирована на *алгебраическую* форму представления функциональных отношений. Обычно символы КА интерпретируются *значениями* переменных, т.е. ориентируются на табличную форму представления [48]. Ей соответствует интерпретация символов X, Y и Q, как множеств входных, выходных и промежуточных *состояний*.

Дополнительным по отношению к модели КА в рассматриваемой

модели является символ *P*. В [44] он характеризуется как признак алгоритмического уровня. Между тем в математической логике [46] предикат характеризует истинность высказывания – истинную или ложную. Применительно к МП БИС предикаты можно интерпретировать как признаки, характеризующие состояния регистров. В частности, к арифметическим предикатам относятся признаки нуля, переноса и знака в сумматоре АЛУ. Они используются в качестве условий ветвления процесса обработки данных для управляющего органа. По этой причине модель КА с дополнительным символом *P* резонно назвать моделью *управляемого* автомата. При этом следует учитывать относительность этого названия, поскольку любой управляющий автомат в иерархической системе является также и управляемым вплоть до автомата самого верхнего уровня, управляемого человеком, даже с самыми вырожденными функциями управления – пуске и остановки системы.

Число сортов символов модели в [45] не поясняется, но судя по числу отражаемых ею уровней, оно равно трём, поскольку МП БИС используется для математической – логической, арифметической и групповой обработки данных. Этим сортам можно сопоставить языки исчисления предикатов *Log*, элементарной арифметики *Ar* и векторного пространства *Vect* [46]. Однако реальное деление ЯПП на сорта ещё тоньше, поскольку и логические, и арифметические переменные делятся на скаляры и векторы. В МП БИС значение логического скаляра (предиката) формируется на выходе логической схемы (комбинационной или последовательностной), а логического вектора – в регистре. С другой стороны, в регистре же размещается в неявном виде и арифметический скаляр *A*:

$$A = \sum_{i=0}^{l-1} c_i \cdot a^i,$$

вычисляемый через коэффициенты c_i *l*-разрядного двоичного вектора, где *a* – основание системы счисления, а $c_i = 0, 1, \ldots, a-1$.

Таким образом, в силу совмещения реализации в МП БИС модулей двух языков – *LogVect* и *ArScal*, соответствующие структуры МП БИС представляются общей моделью регистровых передач.

В отличие от элементарных операций языка *LogVect*, выполняемых

в один такт функционирования МП БИС, арифметические операции языка *ArScal* являются составными. В МП БИС они реализуются схемно или програмно с помощью *алгоритмов* над более элементарными операциями. Операции языка арифметического векторного пространства *Vect* также реализуются в МП БИС алгоритмическим путём с помощью специальных программ.

В каждой из приведённых языков – сортов многосортной модели МП БИС используются все символы этой модели с учётом соответствующего сорта. Это касается и языка самого нижнего уровня – исчисления высказываний (логических скаляров) *Log*, предметные переменные которого группируются для реализации функций переходов из F_{Δ}, функций выходов из $F_{\wedge}$ и условий ветвлений из P.

Между тем в [45] деление по уровням детальности реализации функций МП БИС выполняется не по сортам языков, составляющих многосортную модель, а по составу входящих в неё символов.

Логической функции сопоставляется модель $< X, Y, F_{\wedge}>$,

конечному автомату – $< X, Q, Y, F_{\Delta}, F_{\wedge}>$,

а алгоритму, реализуемому схемно или програмно, – полная модель $<X, Q, Y, F_{\Delta}, F_{\wedge}, P>$.

По составу символов эти модели линейно упорядочены в отношении общее-частное.

В [49] этим уровням, названным соответственно вентильным, функциональным и алгоритмическим, придана следующая предметная интерпретация. «Представителями уровней являются: комбинационный преобразователь, функциональный узел с памятью, контроллер или ЭВМ, работающий по фиксированному последовательному алгоритму». Дополнительно к ним пространственной групповой обработке (в языке *Vect*), не реализуемой однопроцессорной МП БИС, сопоставляется сетевой уровень ВС.

Предложенные интерпретации линейно упорядочены в [49] по «шкале сложности». Действительно, комбинационный преобразователь более прост по сравнению с функциональным узлом, а последний – проще контроллера. Этот порядок соответствует отношению *часть-целое* между приведёнными интерпретациями. Согласно ему комбинационный преобразователь в общем случае составляет часть функционального узла, а последний – часть контроллера. Контроллер представляет собой часть

вычислительной сети.

Исходя из вышеизложенного, нетрудно сделать вывод о несоответствии интерпретаций приведённым моделям ВС. Более того, ни модели, ни их интерпретации не упорядочены по детальности описания ВС, как утверждается в [49].

Согласно [45] формирование содержательных моделей (интерпретаций) предшествовало построению формальной модели системы. Именно этот подход и определил выбранное число уровней модели. Реально же при проектировании МП БИС рассматривается большее число уровней. В частности, важным для проектирования и диагностирования МП БИС является уровень микрокоманд [41], реализующий языки *LogVect* и *Ar*. Микрокоманды используются для реализации алгоритмов обработки информации, сопоставленных в [45] с алгоритмическим уровнем модели. Таким образом, число уровней модели МП БИС фактически больше трёх.

И в завершение анализа рассматриваемой модели нельзя не остановиться на применяемой авторами терминологии. Все наименования уровней заимствованы из различных семантических рядов [21]. Действительно, вентильный уровень поименован *предметным* термином, функциональный – *философским* (общенаучным) термином, а алгоритмический – *математическим* термином.

Резюмируя анализ системного подхода к моделированию взаимодействия уровней сложных ВС, следует отметить необходимость более строгого применения философского, логического и терминологического аспектов при построении теоретико-множественных моделей ВС. Это влияет как на корректность самих моделей, так и на познавательный аспект построенной системы моделей.

1.3. Методы технического диагностирования

Решение задачи обеспечения надёжного функционирования современных сложных ВС таких, как машина с динамической архитектурой, предусматривает *комплексное* использование различных методов технического диагностирования – тестового и рабочего.

В задачу тестового диагностирования входит определение технического состояния диагностирования с помощью специальной –

тестовой последовательности входных воздействий $\mathbf{x}_\text{Т}$. В тривиальном случае в последовательность входят все возможные воздействия на ОД, полученные упорядоченным или псевдослучайным перебором. Однако полный перебор воздействий не всегда возможен и допустим. Сокращение перебора воздействий относится к классу оптимизационных задач. При этом задача построения тестовой последовательности $\mathbf{x}_\text{Т}$ математически формулируется следующим образом:

$$\mathbf{x}_\text{Т} = \min_{L} \overset{L}{\underset{i=1}{\wedge}} \mathbf{x}_i$$

Здесь $\mathbf{x}_i$ – i-е входное воздействие;

L – максимальное количество воздействий (длина тривиальной тестовой последовательности воздействий).

Ограничениями задачи являются выбранная точка наблюдения и перечень предполагаемых неисправностей.

Если объектом минимизации является количество точек наблюдения при постоянном перечне неисправностей и тестовой последовательности, то задача формулируется следующим образом:

$$\mathbf{z}_\text{Т} = \min_{n} \overset{n}{\underset{j=1}{\wedge}} z_j$$

Здесь z_j – j-я точка наблюдения;

n – общее количество точек.

В ряде случаев требуется совместная минимизация длины тестовой последовательности и числа контрольных точек [50]:

$$\mathbf{z}_\text{Т}(\mathbf{x}_\text{Т}) = \min_{j} \min_{i} \overset{n}{\underset{j=1}{\wedge}} z_j (\overset{L}{\underset{i=1}{\wedge}} \mathbf{x}_i)$$

$$\mathbf{x}_\text{Т}(\mathbf{z}_\text{Т}) = \min_{i} \min_{j} \overset{L}{\underset{i=1}{\wedge}} \mathbf{x}_i (\overset{n}{\underset{j=1}{\wedge}}) z_j$$

Для сложных ОД задачи минимизации тестовой последовательности и числа контрольных точек решаются раздельно.

Задача минимизации тестовой последовательности решается двумя способами – либо *исключением* из исходной совокупности всех воздействий, неинформативных по отношению к заданной контрольной точке и перечню неисправностей, либо *включением* в тестовую

последовательность всех информативных в указанном смысле воздействий. Под информативностью здесь понимается ненулевое приращение диагностической информации при включении рассматриваемого воздействия в тестовую последовательность.

В качестве диагностической модели при использовании первого способа применяется таблица функций неисправностей (ТФН) [51]. Минимизации подвергаются совокупности *различающих* воздействий, определяемых несовпадением реакций исправного ОД и его *i*-й неисправной модификации. Для *i*-ой неисправности они представляются дизъюнктивным членом $\bigvee_{t=1}^{s} \mathbf{x}_t^i$. Подставляя его в выражение минимизации тестовой последовательности, получаем выражение:

$$\mathbf{x}_\mathrm{T} = \min_{L} \bigwedge_{i=1}^{L} \bigvee_{t=1}^{s} \mathbf{x}_t^i$$

сформулированное в [51]. В зависимости от формы представления этого выражения, применяются алгебраический [51] и матричный [52] методы его минимизации. Результирующее выражение:

$$\mathbf{x}_\mathrm{T} = \min_{t} \bigwedge_{h} \bigvee_{L} x_t^i$$

представляет h вариантов минимальной тестовой последовательности. Для нахождения приближённой тестовой последовательности применяют градиентные методы (методы локальной оптимизации) с применением информационного критерия и его аналогов [50, 52]. К ним относятся методы поиска (воздействий) с выделением заданной модификации ОД (в [53] только исправной) и с равномерным разбиением модификаций. Метод выделения исправной модификации, рассмотренный в [53], был расширен [54] на неисправные модификации ОД.

Различающее воздействие $\mathbf{x}_t^i$ является существенным по отношению к выявляемой с его помощью *i*-ой неисправности. В том случае, когда оно представляет собой вектор входных переменных, $\mathbf{x}_t^i = (x_1, \ldots, x_k, \ldots, x_m)$, *существенной* в нем является *k*-я переменная, изменение значения которой в присутствии *i*-й неисправности влияет на значение наблюдаемой функции ОД. Если функция f – булева, существенность *k*-ой переменной описывается выражением:

$$f(x_1, \ldots, x_{k-1}, 1, x_{k+1}, \ldots, x_m) \neq f(x_1, \ldots, x_{k-1}, 0, x_{k+1}, \ldots, x_m),$$

характеризующим булеву производную df/dx_k или разность $\Delta f/\Delta x_k$.

Нахождение булева вектора значений переменных, обеспечивающих существенность x_k по отношению к константной неисправности k-го входа элемента v с функцией f, называется *активизацией* этого входа. Обратная операция называется *деактивизацией*.

Например, активированным является первый вход трёхвходового элемента ИЛИ по отношению к одиночной константной неисправности $x_1||0$, если значения его входных переменных определяются вектором (100). Вектор (101) деактивирует первый вход, поскольку при $x_3=1$ изменение значения x_1 вследствие неисправности $x_1||0$ не влияет на значение функции $x_1 \vee x_2 \vee x_3$.

В ОД, состоящем из совокупности функциональных элементов, для проявления неисправности k-го входа элемента $\vee$, выход которого не является внешним, необходимо активировать входы всех элементов, через которые элемент $\vee$ соединен с внешним выходом. Активизация пути выполняется назначением таких значений входных переменных ОД, при которых устанавливается необходимое для обнаружения неисправности значение на неисправном входе элемента $\vee$ и значения промежуточных переменных, обеспечивающие влияние его на выход ОД.

В [55] свойства операций активизации и деактивизации обобщены следующим образом. Активизация пути p_{ab} в ОД – от элемента a до b обозначается булевой переменной $\mathbf{A}p_{ab}$, а деактивизация – переменной $\mathbf{D}p_{ab}$. Аналогично обозначаются активизация и деактивизация k-го входа элемента $\vee$ – $\mathbf{A}_{kv}$ и $\mathbf{D}_{kv}$.

Операции активизации и деактивизации обладают следующими свойствами:

1. $\overline{\mathbf{A}_{kv}} = \mathbf{D}_{kv}$
2. $\overline{\mathbf{A}p_{ab}} = \mathbf{D}\,p_{ab}$
3. $\mathbf{A}\,p_{ab} = \bigwedge\limits_{k=a}^{v=b} \mathbf{A}_{kv}$
4. $\mathbf{D}\,p_{ab} = \bigvee\limits_{k=a}^{v=b} \mathbf{D}_{kv}$
5. $\forall\, q \in p_{ab}\ (\mathbf{A}p_{ab}=\mathbf{A}p_{av} \wedge \mathbf{A}p_{vb})$
6. $\forall\, q \in p_{ab}\ (\mathbf{D}p_{ab}=\mathbf{D}p_{av} \vee \mathbf{D}p_{vb})$

Первые два выражения характеризуют противоположность операций активизации и деактивизации элемента (пути). Третье выражение описывает возможность активизации пути при условии активизации всех составляющих его элементов. Для деактивизации пути p_{ab} согласно выражению 4 достаточно деактивизации хотя бы одного входящего в него элемента. Выражение 5 гарантирует активизацию любого элемента v, лежащего на активируемом пути p_{ab}, а выражение 6 – деактивизацию хотя бы одного элемента в деактивируемом пути p_{ab}.

Выражения 1–6 играют роль аксиом, поскольку они справедливы для всех методов синтеза тестовых воздействий. В [55] на их основе выведены теоремы активизации путей в комбинационной логической схеме

с произвольной структурой.

Различия между методами синтеза воздействий определяются, прежде всего, используемой диагностической моделью ОД. Для случая комбинационных схем различают методы:

1) булевых производных [23];
2) цепей и сечений [28];
3) активизации эквивалентных путей (на базе модели ЭНФ) [25];
4) активизации D-кубов неисправностей [31];
5) активизация путей с коррекцией списков неисправностей [32];

Существуют многочисленные модификации этих методов, не исчерпывающие всё многообразие возможных вариантов.

При синтезе тестового воздействия используются методы, аналогичные методам выделения и равномерного разбиения модификаций, рассмотренные выше для модели ТНФ. Иначе их можно назвать методами *минимального* и *максимального* приращения диагностической информации. Количественно они оцениваются приращением неисправностей, обнаруживаемых воздействием.

Методы синтеза тестовых последовательностей для последовательностных схем либо являются расширением приведённых методов в направлении учёта внутренних состояний [56], либо основываются на установочном и диагностическом экспериментах над конечными автоматами [38].

Методы построения тестовой последовательности микрокоманд для микропроцессорных устройств также основываются на активизации цепочек микрокоманд в графе регистровых передач [40-42]. Они также реализуют принципы минимального и максимального приращения диагностической информации. В [57] методы, использующие эти принципы, названы соответственно *старт-малым* и *старт-большим*. Спецификой микропроцессоров является самотестирование. Поэтому в отличие от предыдущих (пассивных) ОД здесь решается проблема исправного ядра, реализующего тестирование остальных узлов ЭВМ.

Методы одного назначения различаются между собой степенью приспособленности (готовности) диагностической модели к получению требуемых результатов, способами обработки модели, определяемыми формой её представления и различным сочетанием применяемых операций. Различия методов, реализуемых на ЭВМ, порождаются принимаемыми структурами данных и процедурами их обработки.

Приспособленность диагностической модели к построению теста проиллюстрируем на примере двух алгебраических моделей комбинационной схемы – эквивалентной нормальной и скобочной форм булевой функции. Первая из них непосредственно приспособлена к активизации путей, которые помечены в модели специальными индексами при входных переменных. Скобочная форма лишь обладает необходимой информацией для порождения путей в процессе построения теста. Естественно при этом, что трудоемкость метода, использующего вторую модель, выше.

Различными являются и операции, применяемые для той или другой модели. Например, минимизация теста при алгебраической записи различающих совокупностей воздействий в виде ПΣ [51] отличается о минимизации ТФН, представленной в матричной форме [52]. Если при исключении поглощаемых членов в первом случае используются скалярная операция отбрасывания второго члена выражения $a \vee c \wedge b = a$, то во втором случае осуществляется поиск и исключение доминирующего по единицам вектора в матрице различий.

Анализ методов построения тестовых последовательностей показывает их общность по ряду общих признаков и различие в частных признаках, что позволяет устанавливать связь между методами и сопоставлять их между собой. Этот же принцип анализа применим и к методам рабочего

диагностирования, основанным либо на дублировании всех или части свойств ОД, либо на применении кодовых методов [8].

1.4. Принципы систематизации ДО ВС

Развитие архитектуры и увеличение совокупности свойств ВС повысило разнообразие применяемых в ВС методов диагностирования. Для обеспечения отказоустойчивости ВС в различных сочетаниях применяются методы внешнего и само тестирования, дублирования и мажорирования, а также кодовые методы. Ранее эти три группы методов развивались в рамках различных теорий – технической диагностики, надёжности и помехоустойчивого кодирования. В силу самостоятельного предыдущего развития этих теорий их системы понятий оказались недостаточно согласованными. Более того, сами системы понятий в процессе развития не избежали противоречий. Последние стали помехой на пути применения этих систем понятий в области технического диагностирования современных ВС. Как результат этого с одной стороны увеличилась пестрота применяемых терминов, а с другой стороны обнаружился их недостаток – общие понятия стали привлекаться для обозначения частных понятий.

Основной причиной возникновения противоречий в системе понятий является эмпирический подход к её построению. Во избежание этого следует применять элементы формального подхода. Тонким вопросом является определение полноты системы, ибо перечисление всех понятий, которые могут потребоваться на практике, является трудноразрешимой и непродуктивной задачей. В связи с этим следует разграничить базовые понятия системы от производных. Нахождению адекватного для выражения сути проблемы понятия должно способствовать автоматическое порождение понятий, не вошедших в систему.

Как показывает рассмотрение диагностических моделей вычислительных систем, им присуща различная степень общности. Все они находятся между собой в отношении толерантности относительно различных общих признаков – таких, как функция, структура, управление и др. Это свидетельствует о возможности построения системы моделей.

Существует два пути решения этой задачи.

Первый путь заключается в поиске гомологических (общих) частей в совокупностях признаков, характеризующих существующие модели. На основе совокупности выделенных общих признаков порождается каждая существующая модель путём дополнения совокупности признаками, присущими этой модели. Первый этап этого способа построения системы моделей реализует индуктивный метод, а второй – дедуктивный.

Второй путь заключается в поиске свойств объектов, отражаемых моделями, в более общих по отношению к предметной области областях знания – философской и общенаучной [58]. Затем выделяются специфические для данной предметной области свойства объектов. Комбинированием общих и специфических свойств достигается построение моделей конкретных вычислительных систем.

Аналогичными путями может быть построена система методов диагностирования ВС. Специфика построения определяется особенностями методов. Характеризующими их признаками являются используемые в них операции обработки информации. На совокупность операций метода влияют не только его назначение и свойства, но и используемая математическая модель ВС и форма её представления – алгебраическая, табличная или графовая. По сравнению с понятиями метазнанием для методов является не философия, а математическая логика, изучающая свойства операций в различных алгебраических системах [59].

Выражение моделей ВС и методов диагностирования через совокупности характеризующих их признаков позволяет применить к построению систем моделей и методов методологию построения системы понятий предметной области. В качестве унифицированного языка представления моделей и методов, необходимого для построения классификаций, естественно принять язык первого порядка математической логики. Построение классификаций моделей и методов должно подчиняться логическим законам, применяемым для построения систем понятий, что гарантирует их непротиворечивость.

Так же, как и система понятий, системы моделей и методов должны быть открытыми для расширения. Требование полноты должно предъявляться к совокупности признаков, характеризующих модели

и методы, а не к самим системам. Это позволяет на их основе генерировать различные варианты, в том числе ранее не рассматриваемые. Здесь уместно привести аналогию с таблицей химических элементов Менделеева, принципы построения которой были сформулированы до её полного заполнения. Применительно к системам моделей и методов этот принцип важен ещё по той причине, что изменение моделируемых свойств ОД и требований, предъявляемых к методам, могут потребовать новых вариантов моделей и методов, ранее не применявшихся, либо неизвестных.

Разработка принципов построения систем – понятий технического диагностирования ВС, моделей и методов диагностирования ВС позволяет сформулировать задачу построения соответствующих баз знаний. Их назначением является генерация различных вариантов понятий, моделей и методов. При этом пользователя может интересовать как каждый отдельный вариант, так и представитель некоторого класса эквивалентности. Первая цель достигается в АСУ и САПР информационно-*справочными*, а вторая – информационно-*советующими* системами. Основной их компонентой является база данных. В отличие от неё база знаний не хранит, а генерирует различные варианты знания из заранее подготовленных блоков – элементов знания. Поскольку информационно-советующая система призвана осуществлять *выбор* вариантов, необходимо снабдить её дополнительно к информационно-справочной системе механизмами выбора.

Перечисление всех вариантов специального знания представляет собой *NP*-полную переборную задачу. Известна высокая трудоемкость такого рода задач [60]. Тем не менее, современные вычислительные системы позволяют реализовать многие переборные задачи [61], а методы сокращения перебора дают возможность выполнять их в приемлемые сроки. При просмотре вариантов время генерации каждого может исчисляться секундами. Исходя из этого требования, число ярусов дихотомического дерева генерации вариантов на современных ЭВМ не должно превышать двадцати. Это число ограничивает совокупность признаков, на основе которых могут генерироваться варианты специального знания.

Глава 2. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПОРОЖДЕНИЯ ПОНЯТИЙ

2.1. Формализация отношений между понятиями

Понятие характеризуется содержанием (интенсионалом) и объёмом (экстенсионалом) [62, 63].

Содержанием понятия называется совокупность существенных признаков, по которым обобщаются и выделяются предметы в понятие.

Объёмом понятия называется совокупность обобщённых, отражённых в понятии предметов.

Характеристикам понятия – содержанию и объёму – согласно их определениям соответствует множественная интерпретация. Множество признаков, характеризующее содержание понятия, будем обозначать символом C, а множество предметов, отражённых в понятии, т.е. его объём – символом V. Характеризуемое ими понятие a будем помечать индексом при соответствующем символе, например, C_a и V_a.

В приведённых обозначениях понятие a представляется двумя множествами:

$$C_a = \{C_1, \ldots, C_j, \ldots, C_k\},$$
$$V_a = \{a_i \mid \forall C_j \in C_a\},$$

Содержание понятия a задается перечислением k существенных признаков C_j, которыми обладают предметы, входящие в понятие a.

Объём понятия V задается описанием условий, которым удовлетворяют предметы, входящие в понятие a. Этот способ описания позволяет избежать проблем, связанных с представлением большого количества предметов – в ряде случаев счётного множества. Объём понятия V_a в общем случае представляет собой класс, поскольку не всегда можно определить принадлежность элемента понятию a.

Теоретико-множественное представление понятий позволяет формализовать отношения между ними. Из содержательного и объёмного представления понятий для этой цели наиболее приемлемым является первое, поскольку множество существенных признаков перечислимо и имеет небольшую мощность.

Установим отношения между содержаниями понятий. Пусть заданы содержания двух понятия b и d:

$$C_b = \{C_{b1}, \ldots, C_{bi}, \ldots, C_{bk}\},$$
$$C_d = \{C_{d1}, \ldots, C_{dj}, \ldots, C_{dm}\},$$

где C_{bi}, $i = \overline{1,k}$, C_{dj}, $j = \overline{1,m}$, – признаки, характеризирующие понятия *b* и *d*, причем $\forall\ C_{bi} \in C_b$, $C_{dj} \in C_d$ ($C_{bi} \neq C_{dj}$).

Содержание понятия *a*, объединяющего содержания понятий *b* и *d*, должно включать все признаки C_{bi} и C_{dj}, характеризующие каждое понятие *b* и *d* в отдельности:

$$C_a = \{C_{b1},\ldots,C_{bi},\ldots, C_{bk}, C_{d1},\ldots,C_{dj},\ldots, C_{dm}\},$$

т.е. содержание понятия *a* шире, чем содержание понятий *b* и *d*.

Из сопоставления множеств C_b, C_d, C_a следует, что между содержаниями понятий *b* и *a*, *d* и *a* существуют отношения включения:

$$C_b \subset C_a,\ C_d \subset C_a, \tag{2.1}$$

Их можно выразить через операции объединения и пересечения следующим образом:

$$C_a = C_b \cup C_d; \tag{2.2}$$

$$C_b \cup C_a = C_a,\ C_d \cup C_a = C_a \tag{2.3}$$

$$C_b \cap C_a = C_b,\ C_d \cap C_a = C_d \tag{2.4}$$

$$C_b \cap C_d = \varnothing. \tag{2.5}$$

Определим отношения между объёмами понятий *a*, *b* и *d*. Естественно, что совокупностью признаков, характеризирующих понятие *a*, обладает меньшее число предметов, чем число предметов, обладающих совокупностями признаков понятий *b* и *d* в отдельности, так как каждый признак из совокупности вносит дополнительное ограничение на включение предмета в объём понятия *a*. Отсюда следует, что объём понятия *a* меньше объёмов каждого из составляющих его понятий *b* и *d*, т.е. понятие *a* у́же, чем *b* и *d*. Последние являются более общими. Таким образом, если для содержания понятий *a*, *b*, *d* справедливо отношение (2.1), то между их объёмами имеют место обратные отношения включения:

$$V_b \supset V_a,\ V_d \supset V_a \tag{2.6}$$

Выражение (2.6) можно представить следующим образом:

$$V_a = V_b \cap V_d \tag{2.7}$$

$$V_b \cup V_a = V_b,\ V_d \cup V_a = V_d \tag{2.8}$$

$$V_b \cap V_a = V_a,\ V_d \cap V_a = V_a \tag{2.9}$$

$$V_b \cup V_d = V_l, \text{ где } V_l \cap V_a = \varnothing \tag{2.10}$$

Отношения (2.1) и (2.6) и их следствия иллюстрируют закон двойственности содержания и объёма понятия [62]: чем обширнее набор признаков, составляющих содержание понятия, тем уже класс объектов, удовлетворяющих им, и, наоборот, чем у́же содержание понятия, тем шире его объём. Таким образом, если отношения между содержаниями понятий определять по формулам (2.1)-(2.5), то отношение между их объёмами будут определяться по двойственным формулам (2.6)-(2.10).

Отношения совместимости между понятиями *a* и *b* выражаются через их содержание следующим образом:

1) $C_a \cap C_b = \varnothing$ – понятия *a* и *b* несравнимы;

2) $C_a = C_b$ – понятия *a* и *b* – равнозначны (по содержанию);

3) $C_a \subset C_b$ – понятие *a* подчинённое, а *b* подчиняющее;

4) $C_a \cap C_b \neq \varnothing$ – пересекающиеся понятия.

2.2. Родо-видовая связь понятий

Формализация отношений между содержаниями (объёмами) понятий позволяет формализовать процесс получения частного понятия на основе отношения «общее-частное», называемого родо-видовым [19]. Оно, естественно, выражается через содержание понятий. Если исходное понятие a_ρ принять за родовое, а понятие b_i – за видовое отличие, то содержание получаемого на их основе видового понятия $a_{\rho+1,i}$ определяется на основе формулы (2.2):

$$C_{a_{\rho+1,\,i}} = C_{a_\rho} \cup C_{b_i} \qquad (2.11)$$

Здесь ρ – ранг понятия. Величина ρ определяется количеством видовых признаков, которые привлекаются для образования требуемого вида понятия. Ранг рассматриваемого видового понятия, определяемый по отношению к начальному (исходному) родовому понятию, назовём *глобальным*, а ранг, определяемый по отношению к некоторому промежуточному понятию, – *локальным*.

Ранг ρ обладает свойствами метрики, позволяя количественно оценивать степень родства понятий.

Объём видового понятия a определяется на основе формулы (2.7), двойственной формуле (2.2):

$$V_{a_{\rho+1,\,i}} = V_{a_\rho} \cap V_{b_i} \qquad (2.12)$$

Согласно формуле (2.12) объём видового понятия $a_{\rho+1,i}$ меньше объёма родового понятия a_ρ (а его содержимое больше).

Примером вербального видового понятия, полученного в соответствии с формулой (2.11), является «производственный контроль». Его ранг по отношению к родовому понятию «контроль», принятому за исходное, равен единице.

В силу рекурсивного характера формула (2.11) применима для образования видового понятия любого ранга. Так, например, видовое понятие 1-го ранга «производственный контроль» является родовым по отношению к видовому понятию 2-го ранга «выходной производственный контроль». Последнее может быть использовано в качестве родового для образования видового понятия 3-го ранга и т.д.

Утверждение 2.1. Понятия, находящиеся в родо-видовой связи, образуют решетку.

Согласно [64] частично упорядоченное множество образует решетку, если выполняются следующие условия:

1) для всякой пары элементов $a, b \in L$ в L существует такой элемент $c = a \cap b$, что $c \le a$, $c \le b$, причём если некоторый элемент c' обладает свойствами $c' \le a$, $c' \le b$, то $c' \le c$;
2) для всякой пары элементов $a, b \in L$ в L существует такой элемент $d = a \cup b$, что $d \ge a$, $d \ge b$, причём если некоторый элемент d' обладает свойствами $d' \ge a$, $d' \ge b$, то $d' \ge d$.

Этим условиям отвечают все понятия, содержание которых определяется по формуле (2.11), и для которых возможно установить степень родства с помощью ранга.

Проиллюстрируем утверждение 2.1 с помощью вышеприведенного примера. При этом положим, что все существенные признаки, характеризующие содержание понятия, выражаются соответствующими словами в его термине. Тогда пересечением родственных понятий «контроль» и «выходной производственный контроль» является понятие

«контроль» (условие 1), а объединением этих понятий – «выходной производственный контроль» (условие 2).

Назовем решётку, характеризирующую родо-видовую связь понятий, родо-видовой. Элементарная родо-видовая решетка, представляющая формулу (2.11) в виде графа, изображена на рис. 2.1.

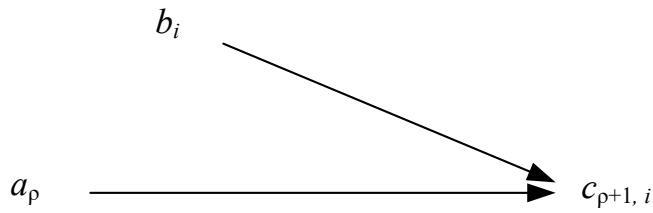

Рис. 2.1. Элементарная родо-видовая решетка

Дуги, заходящие в вершину, соответствующую видовому понятию $a_{\rho+1,i}$, соединяют её с вершинами, соответствующими родовому понятию a_ρ (горизонтальная дуга) и видовому признаку b_i (наклонная дуга).

Рекурсивный процесс образования видового понятия ρ-го ранга описывается последовательным соединением элементарных родо-видовых решеток ρ-1-го, ρ-2-го, …, 1-го ранга. Последовательная родо-видовая решётка, характеризирующая образование видового понятия 2-го ранга «выходной производственный контроль», изображена на рис. 2.2.

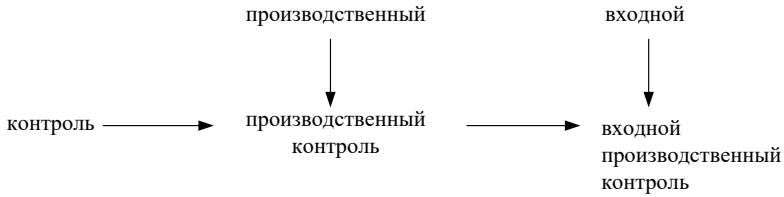

Рис. 2.2. Последовательная родо-видовая решетка

Утверждение 2.2. Последовательная родо-видовая решетка изоморфна цепи видовых понятий.

Согласно формуле (2.11) $C_{a_\rho} \subset C_{a_{\rho+1}}$, $C_{a_{\rho+1}} \subset C_{a_{\rho+2}}$, …, $C_{a_{\rho+n-1}} \subset C_{a_{\rho+n}}$.

Поскольку здесь любая пара видовых понятий сравнима, они образуют по отношению включения линейное упорядоченное множество или цепь [64]:

$$C_{a_\rho} \subset C_{a_{\rho+1}} \subset C_{a_{\rho+2}}, \ldots, \subset C_{a_{\rho+n}} \quad (2.13)$$

Таким образом, за счет исключения видового признака и перехода от операции $\cup$ к отношению включения $\subset$ осуществляется преобразование родо-видовой решётки в цепь видовых понятий. Поскольку содержание понятия $a_{\rho+1,i}$ включает видовой признак b_i, и, следовательно, возможен обратный переход от цепи к родо-видовой решетке, данное отображение является изоморфным.

В силу двойственности содержания и объёма понятий отношение включения в цепи видовых понятий, выраженных в объёмах, является обратным отношению (2.13):

$$V_{a_\rho} \supset V_{a_{\rho+1}} \supset V_{a_{\rho+2}}, \ldots, \supset V_{a_{\rho+n}} \quad (2.14)$$

Согласно выражениям (2.13) и (2.14) содержание видовых понятий в цепи возрастает, а их объём убывает.

2.3. Деление понятий

Сущность деления (наследования признаков [65]) состоит в том, что предметы, входящие в объём делимого понятия, распределяются по группам. Делимое понятие рассматривается при этом как родовое, и его объём разделяется на соподчиненные виды. Основанием деления является признак, значения которого образуют видовые понятия, входящие в объём делимого понятия [66].

Образование i-го видового понятия $a_{\rho+1,i}$ с использованием видового признака b_i, полученного по основанию деления $æ_s$, представляется на базе формулы (2.11). С учётом используемого основания деления она записывается следующим образом:

$$C_{a_{\rho+1,\,i}}(æ_s) = C_{a_\rho} \cup C_{b_i}(æ_s) \quad (2.15)$$

В ней существенной является зависимость видового признака b_i от основания деления $æ_s$. Объём члена деления выражается через объём родового понятия с помощью двойственной формулы, аналогичной (2.12):

$$\mathrm{V}_{a_{\rho+1,\,i}}(æ_s) = \mathrm{V}_{a_\rho} \cap \mathrm{V}_{b_i}(æ_s) \tag{2.16}$$

В соответствии с правилами деления понятий [66] члены деления $a_{\rho+1,i}$ и $a_{\rho+1,j}$, $i, j = \overline{1,k}$, $i \neq j$, представляют собой несовместимые понятия, объёмы которых не совпадают:

$$\mathrm{V}_{a_{\rho+1,\,i}}(æ_s) \cap \mathrm{V}_{a_{\rho+1,\,j}}(æ_s) = \varnothing. \tag{2.17}$$

Например, если за основание деления понятия «контроль» принять способ размещения средства контроля относительно объекта контроля (вне или внутри последнего), то объёмы полученных видовых понятий «внешний контроль» и «внутренний контроль» не совпадают (пересечение их является пустым). Данный пример иллюстрирует также полное деление понятия «контроль» по выбранному основанию деления, так как последнее не порождает других членов деления, кроме приведённых выше.

Полнота деления понятия a_ρ по основанию $æ_s$ определяется формулой:

$$\bigcup_{i=1}^{k} \mathrm{V}_{a_{\rho+1,i}}(\aleph_s) = \mathrm{V}_{a_\rho}. \tag{2.18}$$

Согласно ей, объединение объёмов членов деления равно объём делимого по основанию $æ_s$ понятия a_ρ (правило соразмерности деления [66]). Через содержания понятий данная формула выражается следующим образом:

$$\bigcap_{i\,=\,1}^{k} \mathrm{C}_{a_{\rho\,+\,1,\,i}}(\aleph_s) = \mathrm{C}_{a_\rho}. \tag{2.19}$$

Если имеются всего два члена деления понятий a_s по основанию $æ_s$ (дихотомическое деление), то они находятся между собой в отношении противоречия [66]: $a_{\rho+1,1}(æ) = a_{\rho+1,2}(æ)$, т.е. если одно из этих понятий является истинным, то второе – ложным, и наоборот. При этом для их объёмов и содержаний справедливы соответствующие формулы:

$$\mathrm{V}_{a_\rho} \setminus \mathrm{V}_{a_{\rho+1,\,1}}(æ_s) = \mathrm{V}_{a_{\rho+1,\,2}}(æ_s) \tag{2.20}$$

$$\mathrm{C}_{l}(æ_s) \setminus \mathrm{C}_{b_1}(æ_s) = \mathrm{C}_{b_2}(æ_s), \tag{2.21}$$

где $C_l(æ_s)$ – полное множество видовых признаков, соответствующих основанию деления $æ_s$.

Примером неполного деления понятия «контроль» по основанию деления «стадия жизненного цикла» являются видовые понятия «производственный контроль» и «эксплуатационный контроль». Действительно, помимо упомянутых стадий, контроль может применяться при *проектировании* изделий, их *хранении* и *транспортировке*.

2.4. Фасетная и иерархическая классификации

Они характеризуют многократным делением исходного понятия [66]. В зависимости от способа деления различают параллельную или многоаспектную классификацию, называемую *фасетной*, и последовательную квалификацию, называемую *иерархической* [66].

В фасетной классификации переменным параметром формулы (2.15) является $s = \overline{1, m_\rho}$, а в иерархической – параметр $\rho = \overline{1, n_{\max}}$, Здесь m_ρ – количество взаимно независимых оснований деления, а $n_{\max}$ – максимальное число степеней деления.

Требование взаимной независимости оснований деления в фасетной классификации означает, что ни одно из них не может быть выражено через другое, т.е. $æ_s \neq \varphi(æ_t)$, $æ_s, æ_t \in K_\rho$, $s \neq t$, где K_ρ – множество оснований деления понятия *a* на ρ-ом уровне деления. Другими словами, ни одно из оснований деления не детализирует другое.

Легко проверяется теорема сверки фасет, широко применяемая на интуитивном уровне при классификации понятий.

Утверждение 2.3. Если основания деления $æ_1$ и $æ_2$ порождают два одинаковых фасета видовых понятий, то им соответствует обобщённое основание деления $æ = æ_1 \vee æ_2$.

Это утверждение иллюстрируется рисунком 2.3 для случая деления понятия *a* на два видовых понятия *b* и с по двум основаниям деления $æ_1$ и $æ_2$, ($æ_1 \neq æ_2$). Очевидно, что справедлива и обратная операция – раскрытие основания деления æ на составляющие $æ_1$ и $æ_2$.

Отличительным признаком иерархической классификации является

многоступенчатость деления исходного понятия. При этом имеет место преемственность между основаниями деления разного уровня:

$$æ_{ρ+1} ≠ φ\ (æ_ρ).$$

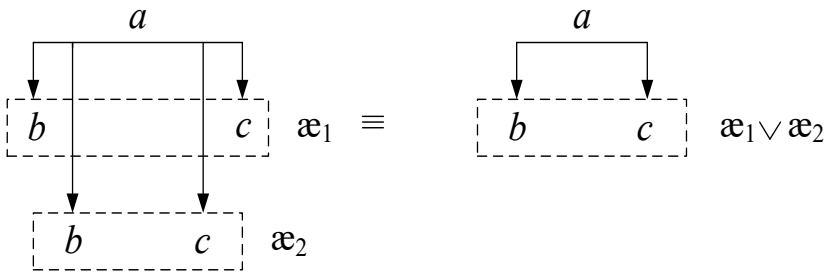

Рис. 2.3. Свёртка фасет

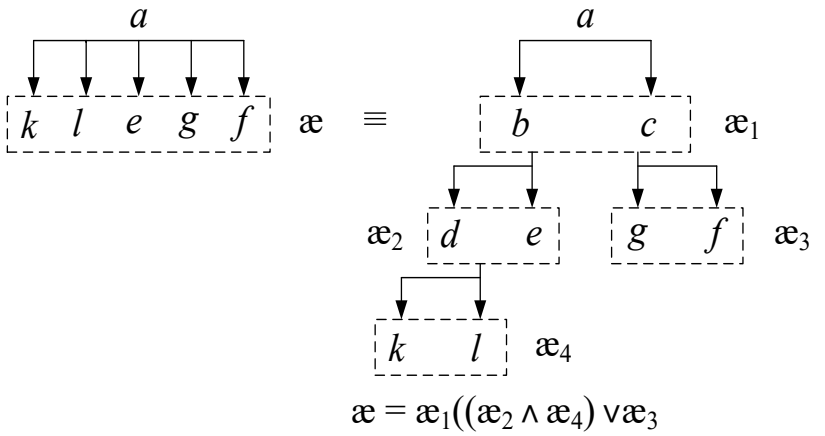

Рис. 2.4. Раскрытие фасеты

Каждое последующее основание детализирует предыдущее. Например, полученное по основанию деления «стадия жизненного цикла изделия, на которой применяется контроль», понятие «производственный контроль» разбивается на *входной*, *операционный* и *выходной*, если стадии производства рассматриваются поэтапно.

Наибольшее распространение получили дихотомические иерархические классификации. Для их получения следует применять теорему раскрытия фасет.

Утверждение 2.4. Если фасет, полученный по основанию деления æ, состоит из $N>2$ видовых понятий, для дихотомического деления предшествующего понятия необходимо привлечь $n =]\log_2 N[$ [1] оснований деления.

Это утверждение иллюстрируется рисунком 2.4. Согласно условию иерархической классификации общность оснований деления $æ_1$, $æ_2$, $æ_3$, $æ_4$ убывает от уровня к уровню, т.е. по содержанию они образуют следующую цепочку: $æ_4 \supset (æ_2, æ_3) \supset æ_1$.

Реальные классификации понятий являются, как правило, смешанными. Они включают фасетные и иерархические классификации в качестве составных частей.

Утверждение 2.5. Любая классификация представима разветвленной родо-видовой решеткой.

Действительно, каждое видовое понятие и в фасетной и в иерархической квалификации образуется в соответствии с формулой (2.11), характеризующей родо-видовую связь понятий. При этом разветвленность родо-видовой решетки обуславливается делением понятий в классификациях. Помимо разветвления, иерархическая классификация выражается последовательным соединением базовых родо-видовых решёток и может рассматриваться как совокупность расходящихся цепей видовых понятий.

Отличительной особенностью разветвленной родо-видовой решетки по отношению к классификации является присутствие в явном виде всех видовых признаков и их связей с видовыми понятиями. Они представляются самостоятельными вершинами графа, соединенными

[1] Символы] [означают ближайшее большее целое число.

с дугами с видовыми понятиями, и являются внешними по отношению к видовым понятиям, входящим в классификацию.

Справедливость утверждения 2.5 для любой классификации объясняется единственностью родо-видовой связи при классификации понятий. Существующие помимо неё собирательная и последовательная связи [132] в соответствии со своим назначением не могут быть использованы для деления понятий. Партитивная связь (целое – часть) не образует классификации, поскольку признаки понятия целого не распространяются на понятие части, и не существует основания деления целого на части помимо самой необходимости членения понятия [132].

Изоморфизм классификации разветвленной родо-видовой решетке доказывается таким же образом, как и для утверждения 2.2.

Видовые понятия, получаемые по различным взаимно независимым основаниям деления, называют *координатными* [132]. Такое название объясняется ортогональностью данных понятий, т.е. их не выводимостью одного через другое. Координатные понятия ρ-го ранга задают базис пространства понятий ρ-го уровня.

Существуют два способа образования понятий на базе координатных:

1) основания деления $æ_s$ и $æ_t$ порождающих координатных понятий различны ($s \neq t$);
2) основание деления порождающих координатных понятий общее.

2.5. Порождение межвидовых понятий

Рассмотрим первый способ образования понятий на основе координатных понятий одного ранга, полученных по различным основаниям деления.

Объединение содержаний любой пары понятий $a_{\rho+1,i}$ и $a_{\rho+1,j}$, относящихся к различным основаниям деления $æ_s$ и $æ_t$, образуется новое (межвидовое) понятие $a_{\rho+2,\,ij}$ со следующим содержанием и объёмом:

$$C_{a_{\rho+2,\,ij}} = C_{a_{\rho+1,\,i}}(æ_s) \cup C_{a_{\rho+1,\,j}}(æ_t) \qquad (2.22)$$

$$V_{a_{\rho+2,\,ij}} = V_{a_{\rho+1,\,i}}(æ_s) \cap V_{a_{\rho+1,\,j}}(æ_t) \qquad (2.23)$$

Утверждение 2.6. В параллельной классификации каждое межвидовое понятие, порождаемое из любых двух координатных понятий, полученных по *различным* основаниям деления, имеет непустой объём.

Согласно выражению (2.17) $V_{\alpha_{\rho+2,ij}} = \varnothing$ только в случае $æ_s = æ_t$. Однако условиям утверждения 2.6 соответствует $æ_s \neq æ_t$ и $æ_s \neq \varphi(æ_t)$, что доказывает его справедливость, т.е.

$$V_{a_{\rho+2,\,ij}} = V_{a_{\rho+1,\,i}}(æ_s) \cap V_{a_{\rho+1,\,j}}(æ_t) \neq \varnothing.$$

Непустой объём межвидового понятия, полученного на основе параллельной классификации, является необходимым, но недостаточным условием его практического применения. К достаточному условию относится практическая целесообразность порождённого межвидового понятия.

Утверждение 2.7. Межвидовое понятие $a_{\rho+2,ij}$, порождаемое координатными понятиями $a_{\rho+1,i}(æ_s)$ и $a_{\rho+1,j}(æ_t)$ при $æ_s \neq æ_t$ и $æ_s \neq \varphi(æ_t)$, представимо последовательной родо-видовой решеткой.

Выразим формулу (2.22), определяющую содержание порождаемого понятия $a_{\rho+2,ij}$ через родовое понятие a_ρ и видовые признаки, с помощью формулы (2.15):

$$C_{a_{\rho+2,\,ij}} = C_{a_{\rho+1,i}}(æ_s) \cup C_{a_{\rho+1,j}}(æ_t) = (C_{a_\rho} \cup C_{b_i}(æ_s)) \cup (C_{a_\rho} \cup C_{b_j}(æ_t)) =$$
$$= (C_{a_\rho} \cup C_{b_i}(æ_s)) \cup C_{b_j}(æ_t).$$

Аналогичным образом на основании формулы (2.23) с привлечением

формулы (2.16) получим выражение для объёма порождаемого понятия

$$V_{\rho+2,\,ij} = (V_{a_\rho} \cap V_{b_j}(æ_s)) \cap V_{b_j}(æ_t).$$

Из утверждения 2.6 следует, что $V_{a_{\rho+2,ij}} \neq \varnothing$.

Полученные выражения характеризуют последовательный процесс порождения понятия $a_{\rho+2,ij}$ на основе родового понятия a_ρ и видовых признаков b_i и b_j, причем очерёдность привлечения последних безразлична. Отсюда следует, что порождение межвидового понятия $a_{\rho+2,ij}$ можно выразить с помощью последовательной родо-видовой решетки.

Поскольку в формулах (2.11) и (2.12), характеризирующих родо-видовую связь понятий, видовые признаки не являются функциями основания деления, они отсутствуют в родо-видовой решетке, отображающей порождение межвидового понятия $a_{\rho+2,ij}$ в параллельной классификации. При этом последнему в родо-видовой решетке соответствует видовое понятие того же ранга.

Отсутствие оснований деления делает невозможным обратный переход от родо-видовой решетки к фасетной классификации, что соответствует гомоморфному отношению между ними.

2.6. Порождение собирательных понятий и понятий-частей

Рассмотрим второй способ образования понятий на базе координатных, относящихся к *одному* основанию деления ($s=t$). Пусть понятие $a_{\rho+2,ij}$ образуется на базе координатных понятий $a_{\rho+1,i}$ и $a_{\rho+1,j}$, полученных по основанию деления на $æ_s$. В соответствии с формулой (2.22) содержание понятия $a_{\rho+2,ij}$ равно:

$$C_{a_{\rho+2,ij}} = C_{a_{\rho+1,i}}(æ_s) \cup C_{a_{\rho+1,j}}(æ_t). \qquad (2.24)$$

Раскроем содержание объединяемых понятий с использованием формулы (2.15):

$$C_{a_{\rho+2,ij}} = (C_{a_\rho} \cup C_{b_i}(æ_s)) \cup (C_{a_\rho} \cup C_{b_j}(æ_s)) = (C_{a_\rho} \cup C_{b_i}(æ_s)) \cup C_{b_j}(æ_s).$$

Отсюда следует, что содержание понятия $a_{\rho+2,ij}$ включает содержание родового понятия и видовых признаков координатных понятий.

Поскольку согласно формуле (2.17) объёмы координатных, относящихся к одному основанию деления, не пересекаются (их пересечение является пустым), объём понятий $a_{\rho+2,ij}$ определяется формулой:

$$V_{a_{\rho+2,\,ij}} = V_{a_{\rho+1,\,i}}(æ_s) \cap V_{a_{\rho+1,\,j}}(æ_s). \qquad (2.25)$$

Из формул (2.24) и (2.25), определяющих образование понятия $a_{\rho+2,ij}$, вторая не приводима к формуле (2.12), характеризующей родо-видовую связь понятий. Следовательно, понятие $a_{\rho+2,ij}$ не может считаться ни видовым, ни межвидовым. Формулам (2.24) и (2.25) соответствует собирательная связь понятий, при которой отдельные понятия объединяются в агрегат – *собирательное* или смешанное понятие.

Утверждение 2.8. Собирательное понятие порождается либо двумя видовыми понятиями, либо любой парой понятий последовательной классификации.

Справедливость утверждения 2.8 по отношению к одному основанию деления параллельной классификации подтверждается формулой (2.25).

Для последовательной классификации характерна зависимость каждого последующего основания деления от предыдущего. Это означает, что каждый фасет последовательной классификации порождается на основе видового понятия предыдущего уровня, что соответствует последовательной детализации исходного понятия. При этом объём последнего делится на непересекающиеся части. Объединение любых двух частей независимо от их положения в классификации и соответствует порождению собирательного понятия, характеризуемого формулами (2.24) и (2.25).

Объём собирательного понятия $a_{\rho+2,ij}$ равен объёму исходного понятия классификации только в случае $\rho=1$ (одного этапа разбиения понятия) и наличия только двух видовых понятий (i-го и j-го) в фасете.

Утверждение 2.9. Партитивная связь (связь между целыми и частью) является обратной собирательной и выражается формулами, аналогичными (2.24) и (2.25):

$$C_{a_{\rho,ij}} = C_{a_{\rho+1,i}}(æ_s) \cup C_{a_{\rho+1,j}}(æ_s) \qquad (2.26)$$

$$V_{a_{\rho,ij}} = V_{a_{\rho+1,i}}(æ_s) \cup V_{a_{\rho+1,\,j}}(æ_s). \qquad (2.27)$$

Действительно, если понятие-агрегат принять за *целое*, то отдельные понятия, из которых оно образуется, являются по отношению к нему *частями*. Членение понятия, принятого за целое, на части (наследование частей [65]), является обратной операцией по отношению к объединению в собирательное понятие. На этом основании партитивная связь между понятиями подчиняется тем же формулам, что и собирательная. Меняются только направленность связей (дуг в графе понятий).

В левых частях формул (2.26) и (2.27) понятие-целое представлено в свернутом виде, включающем два понятия-части. Выразим понятие-целое через объединение его частей, увеличив число последних:

$$C_{a_{\rho,i}} \cup C_{a_{\rho,j}} \cup ,\ldots,C_{a_{\rho,k}} \cup C_{a_{\rho,l}} = (C_{a_{\rho+1,i}}(æ_s) \cup C_{\alpha_{\rho+1,j}}(æ_s)) \cup,\ldots,$$
$$\cup (C_{a_{\rho+1,k}}(æ_t) \cup C_{a_{\rho+1,l}}(æ_t)) \qquad (2.28)$$

$$V_{a_{\rho,\,i}} \cup V_{a_{\rho,\,j}} \cup ,\ldots,V_{a_{\rho,k}} \cup V_{a_{\rho,l}} = (V_{a_{\rho+1,i}}(æ_s) \cup V_{a_{\rho+1,j}}(æ_s)) \cup,\ldots,$$
$$\cup (V_{a_{\rho+1,k}}(æ_t) \cup V_{a_{\rho+1,l}}(æ_t)) \qquad (2.29)$$

Формулы (2.28) и (2.29) описывают промежуточный этап членения целого на группы, включающие по две части. Признаки $æ_s$, $æ_t$ агрегирования частей в группы противоположны по смыслу основаниям деления. Формулы (2.28) и (2.29) иллюстрируют сочетательный закон. С учетом идемпотентности содержаний и объёмов понятий он может быть выражен в следующем виде:

$$C_{a_{\rho,i}} \cup,\ldots,C_{a_{\rho,k}} \cup C_{a_{\rho,l}} \cup C_{a_{\rho,l}} = (C_{a_{\rho+1,i}}(æ_s) \cup C_{a_{\rho+1,k}}(æ_s)) \cup,\ldots,$$
$$\cup (C_{a_{\rho+1,i}}(æ_t) \cup C_{a_{\rho+1,l}}(æ_t)) \qquad (2.30)$$

$$V_{a_{\rho,i}} \cup ,\ldots,V_{a_{\rho,k}} \cup V_{a_{\rho,l}} \cup V_{a_{\rho,l}} = (V_{a_{\rho+1,i}}(æ_s) \cup V_{a_{\rho+1,k}}(æ_s)) \cup,\ldots,$$
$$\cup (V_{a_{\rho+1,i}}(æ_t) \cup V_{a_{\rho+1,l}}(æ_t)) \qquad (2.31)$$

Отсюда следует, что *i*-я часть понятия-целого может группироваться с другими частями более одного раза.

Утверждение 2.10. Собирательная и партитивная связи между понятиями не представимы родо-видовой решеткой.

Справедливость данного утверждения следует из различия формул (2.25) и (2.12), определяющих отношения между объёмами понятий для собирательной (или партитивной) и родо-видовой связей. Согласно формуле (2.25) отношение между порождаемыми и порождающими понятиями при собирательной (или партитивной) связи является следующим:

$$V_{a_{\rho+2,ij}} \supset V_{a_{\rho+1,i}}(æ_s),\ V_{a_{\rho+2,ij}} \supset V_{a_{\rho+1,j}}(æ_s) \qquad (2.32)$$

Они обратны отношениям между объёмами соответствующих понятий, находящихся в родо-видовой связи (см. отношение (2.14) между объёмами смежных понятий в цепи).

Решётки понятий, основанные на собирательных связях, по аналогии с родо-видовыми можно назвать *собирательными*. В отличие от родо-видовых в собирательных решетках так же, как и при порождении межвидовых понятий, обе связи, соединяющие порождающие понятия с порождаемым, равноценны.

2.7. Установление отношений между понятиями

Пусть известна совокупность понятий *M* предметной области. Требуется объединить в подсистемы понятия, между которыми существуют родо-видовая и собирательная виды связей. Согласно ранее изложенному, к таким понятиям относятся родовое понятие подсистемы, видовые, межвидовые и собирательные понятия.

Условием использования формального подхода к решению задачи является наличие сопоставимых определений у сопоставимых понятий. Все определения должны быть одного вида – либо интенсиональными (описательными), либо экстенсиональными (перечислительными). В том случае, если все термины подсистемы понятий являются системными[2], можно для сопоставления понятий воспользоваться непосредственно ими.

[2] Системный термин включает слова, отражающие все существенные признаки понятия, начиная с родового.

Для решения задачи применим модель <M, Σ>. Её сигнатура Σ включает следующие имена отношений: Σ = {⊂, =, ≠, ~}. Отношение включения является теоретико-множественным выражением отношений «общее-частное» и «часть-агрегат». Равенство характеризует тождественность понятия как самому себе (отношение рефлексии), так и понятиям, имеющим термины-синонимы. Неравенство означает принадлежность понятий к разным подсистемам. Символ ~ означает отношение *слабого* родства понятий. Если общим для двух понятий является *родовой* признак, то это свидетельствует о принадлежности понятий одной подсистеме. Если общим является один из *видовых* признаков, это характеризует *сходство* видового отличия понятий, принадлежащих различным подсистемам.

Сопоставление понятий по их содержанию позволяет выявлять кроме вида отношения его направленность (⊂ или ⊃) и количество связей у каждого понятия. Это позволяет, в свою очередь, различать родовое и видовые понятия от межвидовых и собирательных. Структурно родовое понятие характеризуется отсутствием заходящих дуг в графе подсистемы.

Видовые понятия характеризуются наличием единственной заходящей дуги, а межвидовые и собирательные понятия – наличием более одной заходящей дуги. Между собой последние два вида понятий ни структурно, ни по содержаниям различить невозможно, так как формулы нахождения их содержаний (2.22) и (2.24) соответственно совпадают. Для их различения учитываются основания деления порождающих понятий. Если последние принадлежат различным фасетам, ими порождается межвидовое понятие. Если порождающие понятия принадлежат одному фасету (или порождаются из него), то порождаемое ими понятие является собирательным.

Для установления отношения между всеми понятиями предметной области, входящими в совокупность M, рассматривается декартово произведение $M \times M$. Его удобно представить в виде таблицы, по вертикали и горизонтали которой расположены понятия из M. На пересечении i-ой строки и j-го столбца, $i, j = \overline{1, n}$, помечается отношение r_{ij} из сигнатуры Σ, найденное между понятиями a_i и a_j. Общий алгоритм установления отношений между понятиями предметной области приведён на рис. 2.5.

В таблице 2.1. приведены результаты применения алгоритма к следующим понятиям:

- контроль технического состояния – К;
- функциональны – Ф;
- параметрический – П;
- статический – С;
- динамически – Д;
- внутренний – ВН;
- внешний – ВШ;
- текстовой – Т;
- рабочий – Р.

Их аббревиатуры, расположенные по горизонтали и вертикали таблицы 2.1, образуют основу декартова произведения. Для упрощения обозначений у всех понятий опущен символ содержания С. В матрице отношений представлены имена отношений, найденные для каждой пары понятий.

Приведём пример установления вида отношения для пары понятий «контроль»-«функциональный контроль» по самим терминам:

2. $C_U = C(К) \cap C(Ф * К) = \{К\} \cap \{Ф, К\} = \{К\}$

5. $C_E = C(К) \cup C(Ф * К) = \{К\} \cup \{Ф, К\} = \{К\}$

6. $C_{\overline{U}} = C_E \setminus C_U = \langle Ф \rangle$

12. Так как $C_E = C(Ф * К)$, $C(К) \subset C(Ф * К)$

Номера операторов соответствуют номерам блоков алгоритма на рис. 2.5.

Как видно их таблицы 2.1 подматрица отношений понятий ранга 9×9 симметрична относительно диагонали. В частном случае, когда понятия упорядочены по количеству существенных признаков, как это имеет место в таблице 2.1, матрица отношений делится относительно главной диагонали на две треугольные матрицы с отношениями включения только одной направленности в каждой (либо ‘$\subset$’ – в верхней, либо ‘$\supset$’ – в нижней). Таблица 2.1 является также примером единой подсистемы понятий, так как в ней отсутствует отношение ‘$\neq$’ между понятиями.

На основании таблицы отношений можно построить иерархическую

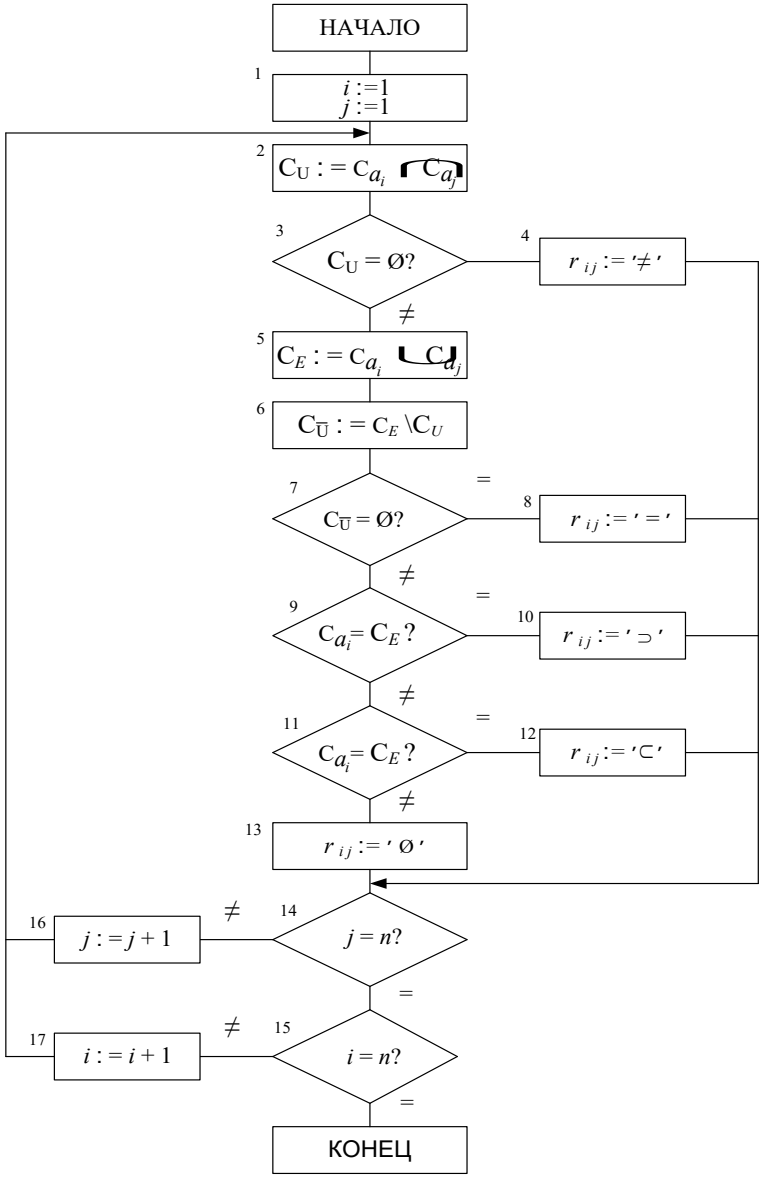

Рис. 2.5. Алгоритм установления отношения между понятиями

Таблица 2.1.

Отношения между содержаниями понятий

М×М	К	Ф К	П К	С К	Д К	ВН К	ВШ К	Т К	Р К
К	=	⊂	⊂	⊂	⊂	⊂	⊂	⊂	⊂
Ф К	⊃	=	~	~	~	~	~	~	~
П К	⊃	~	=	~	~	~	~	~	~
С К	⊃	~	~	=	~	~	~	~	~
Д К	⊃	~	~	~	=	~	~	~	~
ВН К	⊃	~	~	~	~	=	~	~	~
ВШ К	⊃	~	~	~	~	~	=	~	~
Т К	⊃	~	~	~	~	~	~	=	~
Р К	⊃	~	~	~	~	~	~	~	=
ВШ Ф К	⊃	⊃	~	~	~	~	⊃	~	~
Т ВШ Ф К	⊃	⊃	~	~	~	~	⊃	⊃	~

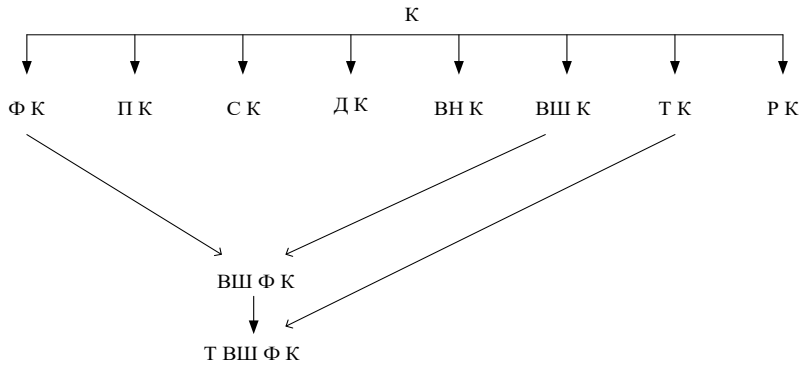

Рис. 2.6. Структура понятий таблицы 2.1.

структуру понятий. Для этого достаточно разделить понятия на порождающие и порождаемые (по содержанию) и построить цепи понятий.

Алгоритм построения структуры основывается на следующих утверждениях, проверяемых по таблице 2.1.

Утверждение 2.12. Понятие a_i является родовым понятием подсистемы, если *i*-ая строка матрицы оказывается *наибольшей* покрывающей строкой (по символу '⊂') по отношению к другим строкам матрицы.

Подмножеству покрываемых ею строк соответствуют понятия, порождаемые родовым понятием подсистемы.

Утверждение 2.13. Понятие a_j порождается понятием a_{j-s}, если *j*-й столбец матрицы является *покрывающим* по отношению к *j–s*-му столбцу (по символу '⊂') и различие в символах '⊂' между ними является *минимальным* по сравнению с любым другим покрываемым им столбцом.

Структура, построенная в соответствии с этими предложениям, на основе таблицы 2.1, изображена на рис. 2.6.

Как было отмечено при постановке задачи, модель подсистемы не обладает большей информацией для различения межвидовых понятий от собирательных. Для этого необходимо знать, относятся порождающие понятия к *разным* фасетам или к одной, т.е. знать основания деления видовых понятий.

2.8. Сетевая и реляционная классификации

Под сетевой классификацией будем понимать сеть, состоящую из родового, видовых, межвидовых и собирательных понятий. Все понятия, входящие в сеть, относятся к одной категории, определяемой родовым понятием. Несколько родовых понятий порождают мультисеть. Непротиворечивость сети гарантируется правилами порождения понятий. Полнота сети определяется заданной степенью детальности представления понятий и отражением ими системы-оригинала. Сетевую классификацию можно считать частным случаем семантической сети [63], в которой отношения между понятиями ограничены родо-видовыми и собирательными. К достоинствам сетевой классификации относятся наглядность связей между понятиями и отражение процесса порождения последних, а к недостаткам – необходимость отражения промежуточных

понятий и, как следствие, – громоздкость сети, отражающей сложные системы понятий.

Этого недостатка лишена реляционная классификация, представляющая собой отношение R между классифицируемыми понятиями A и понятиями признаками B: $R \subset A \times B$. Наиболее наглядным представлением реляционной классификации является таблица, в которой элементы множества A образуют первый столбец, элементы множества B – первую строку, а пересечения строк и столбцов отмечены символами отношения R.

Реляционная классификация может отражать в своей специфической форме любую из рассмотренных классификаций – фасетную, иерархическую, сетевую и их композиции, при этом опуская промежуточные понятия.

Классификация сложных объектов в общем случае является составной, причем каждый компонент её может относиться к любому из рассмотренных видов классификаций.

2.9. Аксиоматическая теория порождения понятий

Представим полученные результаты в рамках формальной теории. В [67] теорией Th алгебраической системы $\mathcal{A}$ называется множество формул $\{\Phi| \mathcal{A} |= \Phi\}$ сигнатуры $\Sigma=<R, F, \mu>$. Здесь R – множество отношений (предикатов) системы, F – множество функций (операций), и μ – функции интерпретации символов R и F. Этому определению теории соответствует совокупность формул (2.1)-(2.32) сигнатуры $\Sigma=\{\subset, \supset, \cup, \cap, \setminus\}$. Сопоставляя эти формулы с аксиомами булевой решётки [68], нетрудно показать, что рассмотренная выше система понятий $\mathcal{A}$ сигнатуры Σ изоморфна булевой решётке той же сигнатуры.

Носитель $\mathcal{M}$ системы понятий $\mathcal{A}_{\text{П}}$ представляет собой *диагональ* или двухместное отношение $U=\{(C_a, V_a)| C_a \in \mathcal{C}_A, V_a \in \mathcal{V}_A\}$, где $a \in A$ – элемент множества понятий, а $\mathcal{C}_A$ и $\mathcal{V}_A$ – множества содержаний и объёмов, характеризующие понятия из A, $|\mathcal{C}_A - \mathcal{V}_A| = 0$. Если понятие представляется только через содержание (объём), то диагональ заменяется одномерным

множеством $\mathcal{C}$ ($\mathcal{U}$).

Приведённая выше запись теории Th предполагает перечисление входящих в неё формул, что зачастую может оказаться неприемлемым для формального описания достаточно сложных реальных систем. Более экономной и регулярной является аксиоматическая теория Th_a совокупность формул которой Ф порождается на основе исходных формул – аксиом Ψ с применением набора В правил вывода [69]:

$Th_a = <\Sigma, \Psi, B>$.

В качестве аксиом теории генерации понятий $Th_П$ примем элементарные формулы – существенные признаки. В качестве правил вывода будем использовать выражения, полученные для порождения понятий разного вида:

(2.15)-(2.19) – *деления* понятий (наследования свойств),

(2.22)-(2.23) – *синтеза* межвидовых понятий,

(2.24)-(2.25) – *синтеза* собирательных понятий,

(2.28)-(2.31) – *членения* понятия-целого (наследования частей).

Помимо них будем использовать общее правила исчисления высказываний [68]:

$$\frac{A,\ A \supset B}{B}$$ (modus ponens) и

$$\frac{C}{C\,(Q_1,\ldots,Q_m||B_1,\ldots,B_m)}$$ (правило подстановки).

В записи аксиоматической теории Th_a отсутствует носитель $\mathcal{M}$. Наряду с сигнатурой Σ он представляет модель $<\mathcal{M}, \Sigma_M>$ теории, если $\Sigma_M = \Sigma$ и после интерпретации сигнатур теории и модели каждая аксиома теории становится истинным высказыванием [67].

Утверждение 2.14. Носитель $\mathcal{M}$ может быть интерпретирован любой теоретико-множественной моделью, если при этом удовлетворяются условия существования модели $<\mathcal{M}, \Sigma_M>$ в теории Th_a.

Это утверждение справедливо в силу того, что при выводе формул (2.1)-(2.32) использовалось *множественное* представление понятий без каких-либо ограничений в интерпретации.

Утверждение 2.14 позволяет применить теорию порождения понятий как к синтезу собственно структуры понятий технического диагностирования, так и моделей и методов диагностирования, представленных в теоретико-множественной форме.

Применительно к конкретному носителю $\mathcal{M}$ аксиоматическую теорию можно представить формальной системой Ф=<*T*, *P*, *A*, *B*> [63,70]. Её символы интерпретируются, соответственно, как множества базовых элементов, синтаксических правил, аксиом и правил вывода (семантических правил). Базовые элементы и синтаксические правила задают язык описания существенных признаков. Если последние интерпретируется понятиями, то за основу принимается естественный язык, а при интерпретации математическими моделями – язык математических символов. Аксиомы в формальной системе задают термы – исходные существенные признаки (родовые и видовые), а правила вывода *B* из аксиом *A*, совпадающих с Ψ, порождают в виде правильно построенных формул производные понятия различного вида.

Утверждение 2.15. Формальные системы, порождающие понятия, модели и методы диагностирования ВС, обладают *конструктивностью* и *разрешимостью* [63].

Конструктивность системы определяется наличием процедур, позволяющих синтаксически различить базовые элементы между собой и правильную запись совокупности базовых элементов. Эти процедуры следуют из возможности формализации языка описания существенных признаков.

Разрешимость системы определяется наличием процедуры сопоставления порождаемых формул с известными образцами понятий, моделей и методов диагностирования ВС и разделения их на семантически правильные и неправильные. С этой целью заранее известные ограничения вариантов генерации задаются запрещёнными формулами, либо их признаками. Соответствующие им формулы признаются неправильными и исключаются из рассмотрения в процессе порождения. Результатом генерации являются семантически правильные формулы, образующие перечислимое множество допустимых вариантов

Формальные системы, генерирующие варианты различного назначения (понятия, модели, методы), представляют собой метасистемы по отношению к системам более узкого назначения. Если формулы метасистемы, в свою очередь, выражать внутренними формальными системами, становится возможным порождать элементы иерархических систем, используя правило подстановки.

Глава 3. СИСТЕМА ПОНЯТИЙ ДИАГНОСТИРОВАНИЯ ВС

3.1. Формирование понятий предметной области

Понятие предметной области (ПрО) порождается на основе следующих исходных знаний – эмпирического (опытного), общенаучного (философского) и абстрактного (рис. 3.1).

В первом случае понятие формируется на основе наблюдения реального объекта и выделения характеризующих его существенных признаков. Полученное таким образом понятие называют эмпирическим [71]. При его формировании используется метод *индукции*. Помимо интенсионального определения (через существенные признаки) эмпирическое понятие во многих случаях может быть определенно остенсиально (указанием на реальный объект).

Во втором случае понятие порождается *дедуктивным* путем на основе более общего – философского или общенаучного понятия [58], существенные признаки которого отбираются и конкретизируются применительно к рассматриваемому объекту ПрО.

В третьем случае понятие ПрО формируется путем *интерпретации* абстрактного понятия (такого, например, как множество, отношение и т.д.) с помощью общенаучных или специальных понятий-признаков (функциональное назначение объекта, принцип его функционирования, статика и динамика работы и т.д.).

Перечисленные источники порождения понятий не исключают, а дополняют друг друга. Первый источник обеспечивает связь понятия с реальным объектом, второй – связь с родственными понятиями смежных областей знания через общенаучные и философские понятия, а третий – обеспечивает построение теоретической модели понятия.

При формировании и анализе системы понятий предметной области важнейшей задачей является установление связей между рассматриваемыми понятиями. В главе 2 приведены формулы, характеризующие различные виды связей. Они выражаются через теоретико-множественные операции над содержанием и объёмом связуемых понятий. Содержание и объём определяемого понятия представляются на естественном языке [72] соответствующими лингвистическими единицами – словами и словосочетаниями.

Общепринятым средством представления содержания и объёма понятия является его определение (дефиниция). Объём определяемого понятия раскрывается в экстенсиональном, а содержание – в интенсиональном определениях. В первом непосредственно перечисляются все предметы, обобщаемые определённым понятием, а во втором указываются существенные признаки этих предметов.

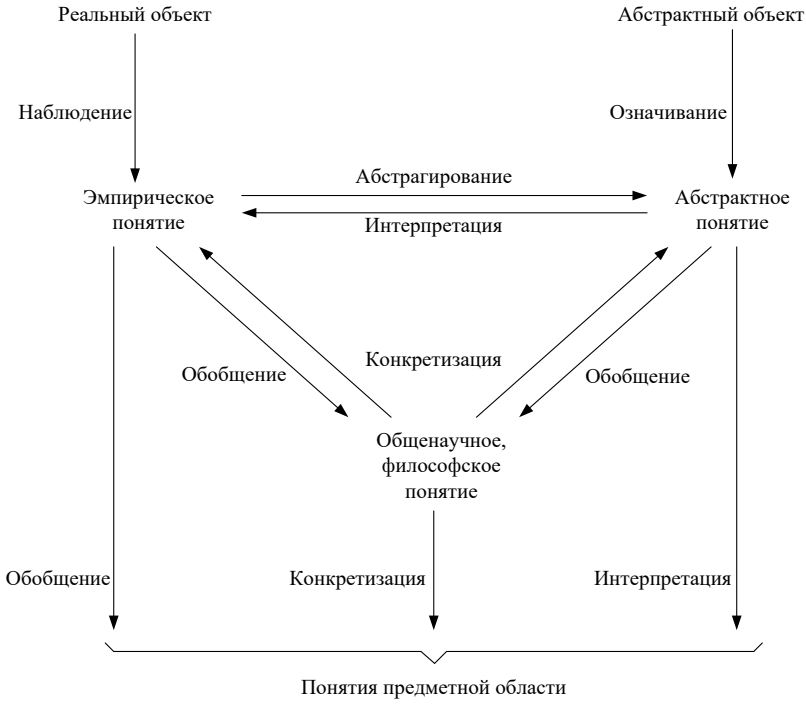

Рис. 3.1. Порождение понятий предметной области

Для представления содержания и объёма понятия в виде совокупности лингвистических единиц необходимо выделить из его определения все существенные признаки. Эта процедура может быть формализована, если определения понятий представлены на специальном проблемно-ориентированном языке. Последний должен обладать *семантикой* естественного языка, допускать *выделение* существенных признаков, отражать *логические* связи между понятиями, быть *универсальным* по отношению к любым определениям и *однозначным* в пределах каждой конкретной предметной области. Определение понятия, выраженное на языке этого типа, представляет собой логико-лингвистическую модель понятия. Она применима как для логико-семантического анализа понятий, так и для установления связей между ними.

3.2. Формализованный язык для определения понятий

3.2.1. Отношения между словами в словосочетаниях

Поскольку в естественном языке одно и то же слово может играть различную роль в словосочетании и быть представленным в разных словоформах, для сопоставления словосочетаний необходимо выразить отношения между словами в явном виде. Чаще всего слова в словосочетании связанны отношением конкретизации [73]. В соответствии с ним любое понятие выражается с помощью *идентификатора* и словарных *конкретизаторов*.

В зависимости от способа выражения отношения конкретизации выделяются различные его виды. Первый вид отношений характеризуется отсутствием глагола. Это, прежде всего, модификация идентификатора справа и слева [74]. Модификация справа осуществляется с помощью дополнения, например, контроль параметра. Модификация слева осуществляется с помощью прилагательного, числительного или причастия, например, параметрический контроль. В обоих примерах слово контроль является идентификатором определяемого понятия, а слова параметр и параметрический – конкретизаторами.

Частным случаем модификации справа является неделимая двухместная конкретизация. Она применяется для идентификаторов, выражающих неразрывную связь двух понятий (сопоставление одного понятия с другим, переходом от одного к другому, влияние одного на другое).

Например, в словосочетании *сопоставление измеренного значения параметра с заданным значением* слово *сопоставление* играет роль идентификатора, связующего словосочетания *измеренное значение параметра* и *заданное значение*. Оба они играют роль конкретизаторов слова *сопоставление*, причем это слово однозначно только в присутствии обоих конкретизаторов. Отметим сокращённый вид второго конкретизатора, иллюстрирующий *семантическую конденсацию* словосочетаний [74].

Другие примеры словосочетаний, реализующих отношение неделимой двухместной конкретизации, – *перевод устройства из произвольного в начальное состояние* и *воздействие внешней среды на изделие*. В первом из них слова *из произвольного* могут опускаться. Этот пример иллюстрирует также одновременное применение модификации справа (*перевод устройства*) и двухместной неделимой конкретизации (*перевод…из произвольного в начальное состояние*). Во втором примере в отсутствие второго конкретизатора *изделие* неделимая двухместная конкретизация сводится к модификации справа (*воздействие внешней среды*).

Другой тип конкретизации характеризуется использованием связующего глагола, применяемого в атрибутной форме [75] – в виде причастия, например, *воздействия, применяемые к изделию*. Здесь слова *воздействия* и *изделие* являются, соответственно, идентификатором и конкретизатором, а причастие *применяемые* выражает отношение конкретизации. Часто в результате семантической конденсации глагол связывается идентификатором более чем с одним конкретизатором. Например, в определении *совокупность свойств объекта, характеризуемых в определенный момент времени признаками, задаваемыми в технической документации,* идентификатор *совокупность свойств объекта* связан причастием *характеризуемый* с двумя конкретизаторами: *определенный момент времени* и *признаки, задаваемые в технической документации*. Последний сам выражен словосочетанием с глагольным отношением конкретизации (причастие *задаваемые*).

Этот пример также иллюстрирует вложение отношений конкретизации и смешанное использование глагольных и безглагольных отношений в реальных определениях понятий. Примером смешанного использования глагольного отношения и неделимой двухместной конкретизации является определение *сигнал, переводящий устройство из произвольного в начальное состояние*. Здесь слияние отношений вызвано переходом идентификатора *перевод* в глагольную форму.

Помимо отношений конкретизации в определениях понятий широко используются логические связки И и ИЛИ. В экстенсиональных определениях специального вида, так называемых *процессуальных* определениях, понятие-процесс определяется через совокупность определяющих его процессов. Если при этом порядок следования определяющих процессов существен, представляющие их словосочетания находятся между собой в отношении временного следования. Например, *под контролем параметра* (*A*) *понимается измерение значения контролируемой величины* (*B*) *и последующее сопоставление полученного значения со значением параметра, принятым за норму* (*C*).

3.2.2. Описание языка для определения понятий

Представим перечисленные выше отношения между понятиями в явном виде. Согласно [63], язык для определения понятий ЯОП [77] принадлежит классу реляционных языков, но, в отличие от языков, описанных в [63], имеет некоторые ограничения, обусловленные спецификой определений понятий. Она заключается в следующем:

- понятия, определяемые в терминологических стандартах, выражаются существительными, а, следовательно, существительным являются и их идентификаторы, входящие в определения;
- определение понятия представлено повествовательным предложением в третьем лице;
- в определении используются только глаголы в изъявительном наклонении, в предикативной или атрибутивной форме [75].

Язык *T* состоит из трёх множеств:

$$T = T_1 \cup T_2 \cup T_3 \qquad (3.1)$$

Множество лингвистических единиц T_1={*A*, *B*, *C*, . . . } представляет собой совокупность терминов конкретной предметной области и слов естественного языка, употребляемых в определениях понятий.

В определениях используются следующие категории [63]: **предмет, свойство, состояние, процесс, событие, оценка, модификатор, квантификатор, модальность.** Первые пять категорий выражаются существительными, например, соответственно: *микросхема, надёжность, техническое состояние, контроль, переключение.* Оценка может выражаться как существительным (*исправность*), так и наречием в значении компаратива (*больше*). Модификатор выражается прилагательными, причастиями и числительными, например, *быстродействующий.* Квантификаторы и модальность выражаются наречиями, например, *часто* и *возможно.*

Элементы множества T_1 должны быть однозначно интерпретируемы (синонимия и омонимия в пределах рассматриваемой предметной области должны быть сняты) и семантически нагружены.

Множество T_2 включает все виды отношений конкретизации:

$$T_2 = \{r_{\text{п}}^{(1)}, r_{\text{п}}^{(2)}, r_{\text{л}}, r_{\text{г}}\}, \qquad (3.2)$$

где $r_{\text{п}}^{(1)}$ – одноместная модификация справа; $r_{\text{п}}^{(2)}$ – двухместная модификация справа; $r_{\text{л}}$ – модификация слева; $r_{\text{г}}$ – глагольная конкретизация.

Множество T_3 включает все логические связки и отношения:

$$T_3 = \{\vee, \wedge, \succ, (,)\} \qquad (3.3)$$

Где $\vee$, $\wedge$ – логические связки ИЛИ, И; $\succ$ – отношение временного следования (слева направо); (,) – круглые скобки; используемые для объединения элементов из T_1.

Символы $\supset$ и $\neg$, обычно используемые в реляционных языках [63], не включены в множество T_3 по следующим причинам:

определения понятий не содержат отношения логического следствия, выражаемого символом импликации $\supset$;

отрицание в наделённом семантикой ЯОП выражается с помощью частицы НЕ.

Формулы (предложения) на ЯОП строятся с помощью общих синтаксических правил [63].

Поскольку все виды отношения конкретизации из T_2 имеют направленность от конкретизатора C к идентификатору B, будем обозначать её там, где это необходимо, стрелкой:

$$B \overleftarrow{r} C, \; C \overrightarrow{r} B \qquad (3.4)$$

При использовании семантически нагруженного алфавита T_1 будем обозначать безглагольные отношения конкретизации следующими символами, отражающими их направленность:

$$B\ r_{\text{п}}^{(1)}\ C = B(C),$$
$$B\ r_{\text{п}}^{(2)}\ (C, D) = B(C, D), \qquad (3.5)$$
$$C\ r_{\text{л}}\ B = C^{*}B.$$

3.2.3. Выражение определений на ЯОП

Наиболее распространенным видом интенсиональных определений является глагольное родо-видовое определение понятия: видовое понятие *A* определяется через родовое понятие *B* и видовой признак *C* с помощью отношения глагольной конкретизации:

$$A = B\ \overleftarrow{r}_{\text{г}}\ C \qquad (3.6)$$

Этой формуле соответствует, например, следующее определение: *постоянное запоминающее устройство* (*A*) *есть запоминающее устройство* (*B*), *предназначенное для* $(\overleftarrow{r}_{\text{г}})$ *хранения неизменяемой информации* (*C*).

Безглагольное родо-видовое определение имеет вид:

$$A = B\ r_{\text{п}}\ C \qquad (3.7)$$

Например, функциональный контроль (*A*) – это распознавание (*B*) вида функционального состояния изделия (*C*).

К простейшим, или вырожденным, безглагольным родо-видовым определениям можно отнести составные термины, обозначающие видовые понятия:

$$A = B(C),\ \ A = C^{*}B. \qquad (3.8)$$

Этими формулами могут выражаться, например, такие понятия как *контроль параметра* и *параметрический контроль*, соответственно.

Особый вид интенсионального определения понятий – сопоставительное определение, реализуемое с помощью неделимого двухместного отношения конкретизации:

$$A = B(C, D) \qquad (3.9)$$

Его можно считать частным случаем родо-видового определения, в котором родовое понятие *B* доопределяется сразу двумя видовыми признаками *C* и *D*, которые не могут применяться в отрыве друг от друга. К родовым понятиям, доопределяемым с помощью двух признаков,

относятся *сопоставление*, *сравнение*, *соответствие*, *отношение* и т.д. Например, *коэффициент усиления транзистора по току* (*A*) – *это отношение* (*B*) *величины выходного тока транзистора* (*C*) к величине входного тока (*D*).

Процессуальное определение описывается формулой:

$$A = B \succ C / D \tag{3.10}$$

Приведённые формулы служат базой для представления реальных определений со сложной структурой отношений. К таким определениям можно отнести даже сравнительно простые определения, иллюстрировавшие эти формулы. Их структура заметно усложнится, если перейти от отношений между словосочетаниями к отношениям между словами. Это достигается с помощью операции подстановки. Будем обозначать её символом ||.

Простейшей подстановкой является замена термина или не термина на соответствующий ему синоним:

$$B\ r\ C\ (C||D) \rightarrow B\ r\ D,$$
$$B\ r\ C\ (B\ ||D) \rightarrow D\ r\ C. \tag{3.11}$$

В общем случае – при подстановке словосочетания – число отношений конкретизации в родо-видовом определении понятия увеличиваются:

$$B\ r_i\ C\ (C||D\ r_j\ E) \rightarrow B\ r_i\ D\ r_j\ E,$$
$$B\ (C,\ D)\ (D||E\ r_j\ F) \rightarrow B\ (C,\ E\ r_j\ F). \tag{3.12}$$

Это же относится к процессуальным определениям:

$$B \succ C\ (C||D\ r_j\ E) \rightarrow B \succ D\ r_j\ E \tag{3.13}$$

Процессуальное определение может быть также разделено на части:

$$B \succ C\ (C||D \succ E) \rightarrow B \succ D \succ E \tag{3.14}$$

Кроме отношений конкретизации и следования, в родо-видовое и процессуальное определение понятий могут вводиться путём подстановки логические связки $\vee$ и $\wedge$:

$$B\ r\ C\ (C||D \vee E) \rightarrow B\ r\ (D \vee E),$$
$$B \succ C\ (C||D \vee E) \rightarrow B \succ (D \vee E). \tag{3.15}$$

Ограничения на подстановки обуславливаются семантической правильностью результирующей формулы определения и наличием определений для замещаемых слов. Последнее, в частности, может касаться модификаторов слева, поскольку они выражаются прилагательными, причастием или числительным, определения которых в терминологических стандартах обычно отсутствуют.

Проиллюстрируем применение подстановок на первом примере, приведенном в этом разделе, заменив словосочетания *A*, *B*, *C* на входящие в них слова с учётом отношений между ними:

$$П*З*У = З*У \overleftarrow{r_Г} X(Н*И) \qquad (3.16)$$

В этой формуле, полученной из формулы (3.6), символы обозначают начальные буквы слов, входящих в определение.

Установление родо-видовой и партитивной связи между понятиями требует приведения их определения к виду, пригодному для составления. Это достигается благодаря эквивалентным преобразованиям определений понятий. Преобразования определений понятий на ЯОП базируется на следующих свойствах отношений этого языка.

I. Распределительный (дистрибутивный) закон[3]

1) модификация справа

$$A(B) \wedge C(B) = (A \wedge C)(B); \qquad (3.17)$$
$$A(B) \wedge A(C) = A(B \wedge C);$$

2) модификация слева

$$B^* A \wedge B^* C = B^*(A \wedge C);\ldots. \qquad (3.18)$$
$$B^* A \wedge C^* A = (B \wedge C)^* A;$$

3) двухместная модификация справа

$$A(B,D) \wedge C(B,D) = (A \wedge C)(B, D); \qquad (3.19)$$
$$A(B \wedge C, D) = A(B,D) \wedge A(C, D);$$

4) одно- и двухместная модификация справа

$$A(D) \wedge A(B, C) = A((D) \wedge (B, C)); \qquad (3.20)$$

5) глагольная конкретизация:

а) с одним глагольным отношением

$$A \overleftarrow{r_Г} B \wedge C \overleftarrow{r_Г} B = (A \wedge C) \overleftarrow{r_Г} B; \qquad (3.21)$$
$$A \overleftarrow{r_Г} B \wedge A \overleftarrow{r_Г} C = \text{A} \overleftarrow{r_Г} (B \wedge C);$$

б) с двумя глагольными отношениями ($r_{Гi} \neq r_{Гj}$)

$$A \overleftarrow{r_{Гi}} B \wedge C \overleftarrow{r_{Гj}} B = (A \overleftarrow{r_{Гi}} \wedge C) \overleftarrow{r_{Гj}} B; \qquad (3.22)$$
$$A \overleftarrow{r_{Гi}} B \wedge A \overleftarrow{r_{Гj}} C = A(\overleftarrow{r_{Гi}} B \wedge \overleftarrow{r_{Гj}} C);$$

[3] Все свойства, демонстрируемые относительно связки $\wedge$, справедливы также для связки $\vee$.

I. Правило сцепления синтагм (операция конкатенации ○)

$$(A\ r_i\ B)\circ B\ r_j\ C = A\ r_i\ B\ r_j\ C; \qquad (3.23)$$

II. Свойства отношения следования:

а) рефлексивность $A \succ A = A$

б) несимметричность $\neg (B \succ A) \supset A \succ B$

в) транзитивность $A \succ B,\ B \succ C \supset A \succ C.$

3.2.4. Семантическая нагруженность ЯОП

Для немашинного представления и анализа определений понятий существенна семантическая нагруженность символов ЯОП. Она реализуется путем непосредственного применения терминов и слов естественного языка для выражения рассматриваемых понятий. Для краткости входящие в определения словоформы представлены аббревиатурами, образованными по правилам сокращения русских и словосочетаний [76]. Исходной словоформе, представленной в словаре предметной области, ставится в соответствие кратчайшая аббревиатура. Количество букв в ней определяется необходимостью различия терминов и слов предметной области. Для различения понятий и глагольных отношений конкретизации их аббревиатуры записываются, соответственно, прописными буквами. Аббревиатуры производных слов дополняются представлениями слово изменяющихся морфов – префиксов и суффиксов. Например, *функция* – ФНК, *функционирование* – *ФНКВ*, *соответствие* – *СТВ*, а *несоответствие* – *НСТВ*.

Представление собирательных и разделительных понятий в виде аббревиатур различается введением модификатора во втором случае. Например, изделие – *ИЗД*, каждое изделие – *КЖД*ИЗД*.

Аббревиатура причастия, выражающего глагольное отношение конкретизации, помимо букв – представителей корня и слово изменяющих морфов, должна содержать признаки залога и времени. Например, *вплщ* – выполняющая, *вплш* – выполнявшая, *вплшс* – выполнявшаяся, *вплм* – выполняемая, *вплн* – выполненная.

Если глагольное отношение уточняется наречием в значении квантификатора, аббревиатура последнего помещается перед причастием. Она, как и аббревиатура причастия, записывается строчными буквами и отделяется интервалом. Например, протекающий быстро – *быстр пркщ*.

3.2.5. Запись и чтение определений понятий

Для записи на ЯОП все отношения между словами и словосочетаниями в определении понятий необходимо привести к виду, удобному для их формализации. Для этого:

- восстанавливаются глагольные формы, опущенные при семантической конденсации (например, очистка фильтрованием → очистка, выполняемая фильтрованием);
- глаголы в предикативной форме переводятся в атрибутивную (представляются в виде причастий);
- наречия-квантификаторы помещаются перед причастиями;
- наречия-компаративы (больше, меньше и т.д.) преобразуются в прилагательные, имеющие статус глагольных отношений, а база сравнения в роли конкретизатора размещается после соответствующего прилагательного (величина больше двух → величина, большая двух);
- восстанавливаются опущенные при семантической конденсации конкретизаторы из двухместных неделимых отношений.

Кроме того, для упрощения определений спускается неинформативные слова (например, слово *процесс* в процессуальных определениях) и сложные словосочетания заменяются более простыми. С другой стороны, сопоставимые словосочетания приводятся к одинаковой структуре отношений.

Чтение определения, выраженного на ЯОП, начинается с главного идентификатора, представляющего собой существительное в именительном падеже. Его род восстанавливается по словарю, а число определяется в зависимости от наличия и значения модификатора слева (род и падеж существительного и этого модификатора совпадают). Модификатор, расположенный справа, имеет родительный падеж. Падеж конкретизатора, связанного глагольным отношением с существительным, определяется его семантикой (место, время, направленность действия и т.д.) и формой управления глагола (слабое, сильное управление).

Приведем примеры записи определений понятий на ЯОП.

1. Вспомогательная память – *это запоминающее устройство, являющееся дополнением или расширением основной памяти и имеющее по сравнению с основной памятью значительно большую ёмкость и большее время обращения*.

*ВСПМ*ПМ = ЗПМЩ*УСТР* (*являс* (ДПЛ ∨ РСШ) (*ОСН*ПМ*) ∧
имщ ((*ЁМК знчт блш ЁМК* (*ОСН*ПМ*)) ∧ *ВРМ*ОБР* (*ОСН*ПМ*))).

Это определение является родо-видовым. Видовое понятие *ВСПМ*ПМ* определяется через родовое *ЗПМЩ*УСТР* и совокупность видовых признаков (в терминах грамматики – через идентификатор и совокупность словарных конкретизаторов).

2. *Установочный сигнал есть сигнал, переводящий устройство в начальное состояние*.

С учетом восстановления опущенного конкретизатора *из произвольного состояния* запись определения на ЯОП имеет вид:

*УСТ*СГН = СГН првщ УСТР* (*ПРВ*СТ*, *НЧ*СТ*).

3. *Распознавание есть процесс, заключающийся в получении информации об объекте и последующем сопоставлении её с имеющейся об объекте информацией.*

Это процессуальное определение приведем к виду, удобному для записи на ЯОП:

Распознавание – это получение информации, представляющей объект, и сопоставление информации, *представляющей объект, с известной информацией, представляющей объект*:

РСП = ПЛЧ (ИНФ *пртщ ОБ*) →
СОП (*ИНФ пртщ ОБ*, *ИЗВ*ИНФ пртщ ОБ*).

Для сокращения записи словосочетание пртщ ОБ может быть опущено.

3.2.6. Определение вида связи между понятиями, представленными на ЯОП

Для установления вида связи между двумя понятиями их определения должны отвечать следующим требованиям:

- характеризовать оба понятия с одной стороны;
- выражаться в одних терминах;
- иметь одинаковую синтаксическую структуру.

Необходимость выполнения перечисленных условий можно сравнить с необходимостью приведения дробей к общему знаменателю перед их алгебраическим сложением.

Вид и ранг [78] связи между сопоставляемыми понятиями определяется путем анализа различающих их элементов. Если различающее понятие входит в объём одного из сопоставляемых понятий, то последние находятся между собой в партитивно-собирательной связи.

Если различающее понятие является элементом содержания одного из понятий и выполняет роль конкретизатора, то между понятиями существует родо-видовая связь; если оно выполняет роль идентификатора, причём выражает понятие-агрегат по отношению к конкретизатору, то между сопоставляемыми понятиями существует партитивно-собирательная связь.

Для формирования содержания (объёма) понятия из интенсионального (экстенсионального) определения последнее расчленяется на элементарные синтагмы [63] – тройки вида *ArB*, где *A* и *B* – слова, выражающие различные понятия, а *r* – отношение конкретизации между ними. Если формула определения понятия имеет группирующие скобки[4], они раскрываются через свойства отношений ЯОП. После этого опускаются логические связки и отношения следования. Полученные фрагменты формулы определения членятся на синтагмы. Они и представляют собой элементы содержания (объёма) понятия.

В соответствии с [78], содержание и объём понятия будет обозначать через *P* и *V*, а в скобках будем приводить термин или определение понятия. Процесс получения множества *P* (*V*) будем описывать последовательным переходом от термина к определению и затем к самому содержанию (объёму) понятия.

В качестве примера определим вид связи между понятиями *установочный сигнал* и *сигнал гашения*. Последнее определим как сигнал, переводящий устройство в нулевое начальное состояние. Формула этого определения, сопоставимая с формулой определения установочного сигнала, имеет вид:

СГН (*ГШ*) = *СГН првщ УСТР* (*ПРЗН*СТ*, *НЛВ*НЧ*СТ*).

Определения обоих понятий являются интенсиональными и сопоставимыми. Найдем содержание этих понятий, выраженное через совокупность синтагм:

P(*УСТ*СГН*) = *P*(*СГН првщ УСТР* (*ПРЗН*СТ*, *НЧ*СТ*)) = (*СГН првщ УСТР*, *првщ* (*СТ*, *СТ*), *ПРЗН*С*, *НЧН*С*);

P(*СГН* (*ГШ*)) = *P* (*СГН првщ УСТР* (*ПРЗН*СТ*, *НЛВ*НЧ*СТ*)) = (*СТ првщ УСТР*, *првщ* (*СТ*, *СТ*), *ПРЗН*СТ*, *НЧ*С*, *НЛВ*СТ*).

Здесь синтагма *првщ* (*СТ*, *СТ*) имеет вид *r*(*A*, *B*) эквивалентный *ArB*.

[4] К ним не относятся скобки, обозначающие модификацию справа.

Вычислим пересечение и объединение содержания понятий как совокупностей синтагм, отражающих существенные признаки:

$$P(U) = P(УСТ*СГН) \cap P(СГН\ (ГШ)) = P(УСТ*СГН),$$

$$P(E) = P\ (УСТ*СГН) \cup P(СГН\ (ГШ)) = P(СГН*ГШ).$$

Поскольку $P(U) \neq P(E)$, содержания этих понятий различны. Они различаются признаком:

$P(E) \setminus P(U) = НЛВ*СТ$.

Слово *состояние* (*СТ*) входит в содержание обоих понятий. Следовательно, различающим является нулевое (НЛВ). Оно выполняет в синтагме роль конкретизатора. Поэтому между рассматриваемыми понятиями существует родо-видовая связь, причем понятие *сигнал гашения* является видовым по отношению к понятию *установочный сигнал*.

Для иллюстрации партитивно-собирательной связи сопоставим понятия *совокупность возможных состояний* и *возможное состояние*. Содержание этих понятий имеет вид:

(*СВП* (*СТ*), *ВЗМ*СТ*) и (*ВЗМ*СТ*).

Они различаются синтагмой *СВП* (*СТ*), причём различающим словом является *совокупность*, которое выполняет в синтагме роль идентификатора и выражает понятие-агрегат по отношению к понятию *состояние*. На этом основании делается вывод о наличии партитивно-собирательной связи между сопоставляемыми понятиями.

3.3. Метод построения системы понятий предметной области

На начальном этапе анализируется предмет исследования, его свойства, процессы взаимодействия его с другими объектами, свойства процессов, их характеристики. Результатом анализа является набор основных понятий предметной области, связанных каузативными отношениями. Эти понятия являются ядрами подсистем понятий предметной области. Они конкретизируются с помощью видовых отличий.

При выборе очерёдности включения понятий в рассматриваемую ПрО целесообразно руководствоваться прагматическими соображениями. Утилитарный подход к изучению любого объекта концентрирует внимание на его изменении (в пространстве, во времени) или на выполняемых над ним действиях. Поэтому первоочередное понятие ПрО

должно относиться к категории действия. Математически оно представляется *n*-местным предикатом, а на естественном языке – отглагольным существительным. К этой категории понятий относятся, например, *проектирование*, *изготовление*, *транспортировка*, *хранение, эксплуатация*, *оценивание*, *распознавание* и т.д. Каждое из них применимо к объекту любой природы (первые два – только к техногенным объектам). В свою очередь, один и тот же объект может быть предметом исследования различных ПрО. Например, *техническое состояние* является предметом исследования технической диагностики, надежности, прогнозирования. Остальные понятия ПрО порождаются методом «раскрутки от ядра» путем нахождения ответов на вопросы: «каким образом?», «с помощью чего?», «чем?», «когда?», «куда?» и т.п. При этом отношения между порождаемыми понятиями характеризуются последовательной связью.

Графически система понятий, основанная на последовательных связях, представляется семантической сетью, вершинам которой соответствуют генерируемые понятия, а дугам – двухместные предикаты (типа «быть методом», «быть средством» и т.д.), связывающие пары понятий. Базовый перечень понятий определяется спецификой ПрО. Например, в него помимо процесса (действия) могут входить понятия – *объект*, его *свойства*, *состояния*, их *оценки*, *метод* и *средство* реализации процесса, их *характеристики*, *оценки*. Понятия организационной области должны, в свою очередь, отражать *субъект действия*, *место* и *время* действия, их *характеристики* и *оценки*.

Изложенные выше принципы реализуются следующей методикой построения структуры понятий ПрО [79, 80].

1. Выделяется наиболее общий процесс ПрО и формализуется с помощью *n*-местного предиката $P(x_1, \ldots, x_n)$.
2. Формируется содержание понятия $P(x_1, \ldots, x_n)$ на основе нахождения общего с философским или общенаучным понятием и особенного, характеризующего ПрО.
3. Определяется состав аргументов $P(x_1, \ldots, x_n)$. Формируется содержание каждого аргумента таким же образом, как и в п.2.
4. Если возможно, предикат $P(x_1, \ldots, x_n)$ разделяется на части, т.е. на составляющие его процессы:

$$P(x_1, \ldots, x_n) = <P(x_1, x_2)\ldots..., P(x_1, x_2), P(x_{n-1}, x_n)>.$$

5. Формируется содержание предикатов $P\ (x_1, x_2)$….., $P\ (x_1, x_2)$, $P\ (x_{n-1}, x_n)$ так же, как в п.2.
6. Определятся состав и содержание аргументов этих предикатов так же, как в п.3.
7. Последовательно находятся ответы на следующие вопросы к переменным – аргументам предикатов: «каким образом?», «с помощью чего?», «чем характеризуется?», «к чему относится?» и т.д. Ответом на вопрос считается либо найденное семантическое значение переменной, т.е. понятие, либо новая переменная. Во втором случае процесс раскрутки продолжается до установления содержания всех базовых понятий ПрО.
8. Осуществляется классифицирование и/или членение всех базовых понятий ПрО. Здесь основной проблемой является нахождение оснований делений понятий и видовых признаков (частей).
9. Проверяется возможность генерации производных понятий – межвидовых и собирательных и устанавливаются разрешенные (запрещённые) понятия на основе их сопоставления с реальными объектами.
10. Построение структуры понятий завершается после отражения понятиями всех особенностей ПрО.

При пересмотре существующей структуры понятий ПрО выполняются следующие процедуры:

- при необходимости переопределяются известные понятия (обобщаются или уточняются);
- изменяются связи переопределенных понятий;
- при необходимости вводятся новые основания деления и видовые признаки;
- добавляются новые понятия и связи.

Анализ структуры понятий на непротиворечивость, порождение понятий, проверка и установление связей между ними выполняются с помощью изложенного выше формального языка определения понятий с учетом того, что «важны в познавательной деятельности те возможности математического аппарата, которые позволяют высветлить противоречие, сделать его «невыносимым» для простого рассудка, для ограниченно-метафизического мышления» [67]. Названные операции используются в пункте 2 вышеприведенного алгоритма.

3.4. Логико-семантический анализ основных понятий технической диагностики

Основной задачей технической диагностики является *распознавание* состояния объектов технической природы. Оно осуществляется путём отождествления диагностируемого объекта с одним из его *образцов*: правильным или неправильным. Образцы технического объекта обычно задаются в нормативно-техническом документе (НТД) на изделие, например в технических условиях. Обычно в НТД задаётся правильный образец изделия, причём не в единственном экземпляре, а как представитель класса, определяемого допустимыми диапазонами изменения свойств изделия.

Важной особенностью образцов является их вневременной характер, поскольку они применяются к изделию на всех стадиях его жизненного цикла. В противоположность образцам свойствам изделия присуща *изменчивость*. Для разрешения противоречия между изменчивостью свойств изделия и стабильностью его возможных образцов используется понятие *техническое состояние*.

2.4.1. Техническое состояние

Являясь частным по отношение к общенаучному понятию *состояние*, оно должно охватывать существенные признаки последнего. Примем за основу ещё более общее, чем общенаучное, – философское понятие *состояние*. В [81] оно определено как «философская категория, отражающая специфическую форму реализации бытия, фиксирующая момент устойчивости в изменении, развитии, движении материальных объектов в некоторый данный момент времени при определенных условиях».

В этом определении нашли отражение первые два из трёх законов связи состояний:

- одного объекта во *времени* или временной связи состояний объекта;
- одного объекта в данный момент времени с состояниями *других* объектов в тот же момент времени;
- *внутреннего* и *внешнего* состояний объекта.

Проинтерпретируем существенные признаки состояния из приведённого определения применительно к рассматриваемой предметной

области. «Специфическая форма реализации бытия» сводится к утилитарным свойствам объекта диагностирования. Устойчивость свойств технического объекта проявляется в фиксации их в определенный момент времени. «Определенные условия» фиксации свойств характеризуются значениями параметров внешней по отношению к техническому объекту среды, при которых определяются его свойства. Существенность данного признака следует аз очевидной зависимости свойств технического объекта от условий внешней среды.

Сопоставим с рассмотренными признаками определение технического состояния, введённое в [126]. В нём оно определено как «совокупность подверженных изменению в процессе производства или эксплуатации свойств объекта, характеризуемая в определённый момент времени признаками, установленными технической документацией на этот объект». Логико-лингвистическая модель этого понятия на языке ЯОП с применением соответствующих аббревиатур имеет следующий вид [82]:

$$ТХН\ С = СВП(СВ\ (измн\ (ПРВ \vee ЭКСП) \wedge ОБ)$$
$$харм(ПРЗ\ устн\ ТХН\ ДКМ \wedge ОМВ)). \qquad (3.24)$$

Путём раскрытия скобок извлечём из формулы определения *ТХН С* все характеризующие его признаки, выраженные через синтагмы:

$$СВП(СВ(ОБ)),$$
$$СВП(СВ\ измн(ПРВ \vee ЭКСП)),$$
$$СВП(СВ\ харм(ПРЗ\ устн\ ТХН\ ДКМ)),$$
$$СВП(СВ\ харм\ ОМВ).$$

Первая синтагма нуждается в уточнении понятия *объект*. Его необходимо дополнить признаком технический:

$$СВП(СВ(ТХН\ ОБ)).$$

Вторая синтагма не является полной. В неё не включены такие стадии жизненного цикла изделия как *разработка*, *транспортировка* и *хранение*, в течение которых изделие также может изменять свои свойства. Поэтому следует либо дополнить ими определение, либо вообще исключить из него признак изменчивости как очевидный.

Для анализа третьей синтагмы уточним смысл двух используемых в нём понятий – свойства и признака. Свойство – это то, что необходимо принадлежит предмету, выражает его внутреннюю природу. Признак – это любая черта, сторона, состояние предмета, которая выделяет его среди других.

Основываясь на смысле данных понятий, приходим к выводу, что третья синтагма носит оценочный характер, постольку характеризация признаками означает выделение состояния объекта среди других его возможных состояний. Этот вывод подтверждается практикой использования определения технического состояния, в соответствии с которой под техническим состоянием обычно понимают оцененное (определённое) состояние. Между тем, оценивание состояния требует выполнения определённых действий таких, как сбор информации о фактических свойствах изделия и сопоставление их с признаками, зафиксированными в НТД. Для разделения неоцененного технического состояния от оцененного следует исключить третью синтагму из определения понятия.

Четвёртая синтагма отвечает второму существенному признаку понятия *состояние*. Её целесообразно уточнить как совокупность свойств, фиксируемых в определённый момент времени (ОМВ):

СВП(*СВ* фикс *ОМВ*).

Следует указать на относительность понятия *момент времени* в данной синтагме. Определение (оценка) технического состояния сложного объекта требует заметных затрат времени. Если это время неизмеримо мало по сравнению со временем заметного изменения свойств объекта, то принятая идеализация времени как момента является вполне допустимой.

Таким образом, из четырёх проанализированных синтагм существенные признаки понятия *состояние* содержат первая и четвертая синтагмы. Упущенным оказался третий признак, характеризующий взаимосвязь внутреннего и внешнего состояний. Его можно сформулировать как «совокупность свойств объекта, фиксируемых при определенных условиях внешней среды»:

СВП (*СВ фикм ОПР УСЛ* (*ВНШ СРД*)).

Объединив три принятые синтагмы, получим следующую логико-лингвистическую модель понятия *техническое состояние*:

ТХН С = *СВП*(*СВ* (*ТХН ОБ* ∧ *фикм* (*ОМВ* ∧ *ОПР* УСЛ (*ВН СРД*)))).

Она читается как «совокупность свойств технического объекта, фиксируемых в определённый момент времени при определенных условиях внешней среды». Возможны и другие редакции определения, сохраняющие тот же смысл.

2.4.2. Анализ понятий техническое диагностирование, поиск дефекта, контроль технического состояния, регламентированных в терминологическом стандарте

В главе 1 была указана неоднозначность формулировки этого понятия в пределах одного терминологического стандарта. Проанализируем его с помощью ЯОП. Оно определенно в [127] как «процесс определения технического состояния объекта диагностирования с определенной точностью». Поскольку точность диагностирования в [127] не определена, исключим её из определения. Для сокращения записи опустим также слова процесс и объект. С учётом замечаний формула определения на языке ЯОП имеет вид [83]:

$$Д = ОПР\ (ТХН\ С). \qquad (3.25)$$

Согласно этому определению некоторые исследователи трактуют техническое диагностирование как некое обобщённое измерение технического состояния.

Второй смысл диагностирования заключён в примечаниях 1 к определению [127]: «результатом диагностирования является заключение о техническом состоянии объекта с указанием при необходимости, места, вида и причины дефекта». Очевидно, что заключение о техническом состоянии объекта можно сделать только по результату его сопоставления с заранее известным техническим состоянием (образцом). Оно может быть как правильным, используемым при контроле технического состояния объекта, так и неправильным, применяемым при поиске дефекта.

Вследствие того, что заключение о техническом состоянии объекта основано на его оценке, оно может быть получено только путём определения технического состояния, заключающегося в нахождении значений характеризующих его параметров и функций, и последующего сопоставления *определённого* технического состояния с одним из его известных дефектных (неправильных) образцов. Этому смыслу технического диагностирования соответствует следующее его процессуальное определение на языке ЯОП:

$$Д = ОПР\ (ТХН\ С) \succ СПСТ\ (ОПРН\ ТХН\ С, ИЗВН\ ДФН\ ТХН\ С). \qquad (3.26)$$

Формула (3.26) включается в качестве первого члена в формулу (3.25), что свидетельствует о неоднозначности определения технического

диагностирования. Неоднозначность проявляется также при установлении в [127] отношений между техническим диагностированием, контролем технического состояния и поиска дефекта. По отношению к первому из них оно трактуется в [127] как «часть процесса при контроле технического состояния», что отвечает формуле (3.25). Другой частью процесса контроля технического состояния остаётся сопоставление определённого технического состояния с правильным – заданным в НТД. Это отношение выражается следующей формулой контроля технического состояния:

$$К(ТХН\ С)=Д \succ СПСТ(ОПРН\ ТХН\ С, ИЗВН\ ПРВН\ ТХН\ С). \quad (3.27)$$

Из этой формулы следует, что техническое диагностирование является незаконченным процессом, не завершающимся постановкой технического диагноза.

При определении понятия *поиск дефекта* в [127] техническое диагностирование трактуется как законченный процесс. Действительно, определение понятия поиск дефекта как «диагностирования, целью которого является определения места и, при необходимости, причины и вида дефекта» согласуется с [127, примечание 1] и, следовательно, соответствует формуле (3.26):

$$П(ДФ)=\text{ОПР}(ТХН\ С) \succ СПСТ(ОПРН\ ТХН\ С, ИЗВН\ ДФН\ ТХН\ С). \quad (3.28)$$

Идентичность формул (3.26) и (3.28) характеризует техническое диагностирование и поиск дефекта как синонимы, что и нашло подтверждение в публикациях по технической диагностике.

Для уточнения содержания этих понятий рассмотрим наиболее близкое к ним общенаучное понятие. В качестве такового резонно принять распознавание (узнавание) заранее известного объекта, поскольку при техническом диагностировании, контроле технического состояния и поиске дефектов образцы изделия (правильные и неправильные) остаются неизменными. Логико-лингвистическая модель распознавания описывается следующей процессуальной формулой, характеризующей процесс сбора информации об объекте и сопоставлении её с образцом:

$$РП= СБР(ИНФ) \succ СПСТ(ИНФ, ОБР). \quad (3.29)$$

2.4.3. Сбор информации

Сбору информации о техническом объекте соответствует определение его технического состояния, включающее в общем случае измерение

параметров объекта и определение значений функций: параметров объекта и определение значений функций:

$$ОПР(ТХН\ С)=ИЗМР(ПРМ) \wedge ОПР(ЗНЧ\ (ФНК)). \quad (3.30)$$

Измерение включает в себя следующие процессы:

- установку требуемых параметров внешней среды, либо их учёт в силу невозможности установки;
- сопоставление измеряемой величины с единицей измерения;
- подсчёт числа единиц измерения, соответствующего измеряемой величине.

Логико-лингвистическая модель его имеет следующий вид:

$$ИЗМР(ПРМ) = (УСТ \vee УЧЁТ)\ \ (ЗНЧ(ПРМ(ВНШ\ СРД) \succ$$

$$СПСТ(ВЛЧН, ЕДНЦ)\ \succ\ ПСЧ\ (ЧСЛ(ЕДНЦ)). \quad (3.31)$$

Последний член формулы характеризует назначение измерения, заключающееся в количественной оценке физической величины. Однако измерение не исчерпывается подсчётом числа единицы и должно, подобно распознаванию, включать в себя подготовительные стадии.

Измерение, в свою очередь, является составной частью контроля параметра, включающего дополнительно сопоставление измеренного значения параметра с образцом:

$$К\ (ПРМ) = ИЗМР\ (ПРМ)\ \succ\ СПСТ(ИЗМР\ (ПРМ), ОБР). \quad (3.32)$$

Измеренное значение параметра соответствует значению функции от параметров, характеризующих как входное воздействие на объект, так и влияние внешней среды.

Во многих случаях для оценивания технического состояния объекта помимо точечного значения функции требуется определение её значений для некоторых диапазонов значений параметров-аргументов. Логико-лингвистическая модель процесса определения значений функции также, как и процесс измерения параметра, включает первый член формулы (3.31). Помимо него модель отражает процессы генерации выходных воздействий на объект и фиксацию реакций на них:

$$ОПР(ЗНЧ(ФНК) = (УСТ \vee УЧЁТ)\ \ (ЗНЧ(ПРМ(ВНШ\ СРД) \succ$$

$$ГНР(ВЗД)\ \succ\ ФКС(РКЦ) \quad (3.33)$$

Вместо привлечения двух новых понятий для определения понятия *ОПР*(*ЗНЧ*(*ФНК*) последнее можно определить через совокупность измерений параметров:

$$ОПР(ЗНЧ(ФНК)= \text{СВКП(ИЗМР(ЛРМ))}. \quad (3.34)$$

ИЗМР(*ПРМ*) и *ОПР*(*ЗНЧ*(*ФНК*).

Первое является родовым, т.е. более общим. Формула (3.34) отражает родо-видовую связь между понятиями общим, по отношению ко второму. Если же рассматривать эти понятия как процессы, то первое их них представляет собой *часть* второго.

Эта неоднозначность характеризует относительность связей между понятиями. С учётом её следует оговаривать, относительно чего установлена связь между понятиями – относительно содержания, либо объёма.

2.4.4. Техническое диагностирование, поиск дефекта, контроль технического состояния

В соответствии с вышеизложенным понятие *техническое диагностирование*, завершающееся техническим диагнозом, логично определить как видовое по отношению к понятию распознавания. Важной особенностью последнего является его инвариантность по отношению к применяемым для сопоставления образцам объекта – правильным, либо неправильным. С учётом её логико-лингвистическая модель технического диагностирования, основанная на формуле (3.26), имеет следующий вид:

$$Д{=}ОПР(ТХН\ С) \succ СПСТ(ОПРН\ ТХН\ С, ИЗВН\ ТХН\ С). \quad (3.35)$$

Она отличается отсутствием признака *дефектное* у понятия *известное техническое состояние*. При этом условии понятия *контроль технического состояния* и *поиск дефекта* могут считаться видовыми по отношению к понятию диагностирования относительно основания деления *вид образца* (правильный или дефектный):

$$К(ТХН\ С) = ОПР(ТХН\ С) \succ СПСТ(ОПРН\ ТХН\ С, ИЗВН\ ПРВ\ ТХН\ С), \quad (3.36)$$

$$(ДФ) = ОПР(ТХН\ С) \succ СПСТ(ОПРН\ ТХН\ С, ИЗВН\ ДФН\ ТХН\ С). \quad (3.37)$$

Из сравнения формул (3.31), (3.32) и (3.35) следует, что понятие *измерение* является частью понятия *техническое диагностирование*.

2.4.5. Тест, испытание

Термин *тест* происходит от английского термина test. Последний определён в американском терминологическом стандарте по контролю и диагностике как «процедура или действие, предпринимаемые с целью определения при реальных или имитируемых условиях возможностей, ограничений, характеристик, эффективности, надежности или пригодности материала, устройства, системы и метода»[5].

[5] перевод В.Б. Ныркова.

Приведенному определению в отечественной литературе соответствует термин *испытание*.

Для установления связи понятия *испытание*, относящегося к теории надёжности, с понятиями технической диагностики необходимо констатировать тот факт, что достижение перечисленных в определении целей испытания осуществляется через распознавание технического состояния объекта. По сравнению с техническим диагностированием, выполняемым в номинальном режиме функционирования объекта, испытание характеризуется разнообразием режимов, имитирующих различные состояния внешней по отношению к объекту среды. Это достигается путем вариации значений параметров, характеризующих среду. Отнесём к параметрам внешней среды и параметры входных воздействий на объект. Тогда понятие *испытание* помимо двух процессов, характеризующих техническое диагностирование в формуле (3.35), должно включать процессы установки или изменения (вариации) значений параметров внешней среды и фиксации допустимых значений варьируемых параметров [84]:

$$ИСП = (УСТ \vee ИЗМН)\ \ (ПРМ(ВНШ(СРД)) \succ$$
$$ОПР(ТХН\ С) \succ СПСТ(ОПРН\ ТХН\ С, ИЗВН\ ТХН\ С) \succ$$
$$ФКС(ДПМ(ЗНЧ(ВНШ(ПРМ))). \qquad (3.38)$$

Если в качестве известного технического состояния объекта в третьем члене формулы принять его работоспособное состояние, то целью испытания является определение области работоспособности объекта относительно варьируемого параметра. Аналогичным образом конкретизируются и другие цели испытания объекта.

Из сопоставления формул (3.26) и (3.38) следует, что понятие *диагностирование* относительно объёмов понятий является *частью* понятия *испытание*, а относительно содержания этих понятий – более *общим*, чем испытание.

Русскоязычный термин *тест* обозначает не процесс, как test, а диагностическую информацию, используемую при выполнении процесса. При этом существует два различных определения понятия тест. Первое, заимствованное из работы [51], зафиксировано в терминологическом стандарте [127]: «одно или несколько тестовых воздействий и последовательность их выполнения, обеспечивающие диагностирование».

Второе, согласованное с практикой тестирования БИС и ЭВМ, сформулировано в [133] как «последовательность входных воздействий и определенная для неё последовательность выходных реакций, предназначенные для установления соответствия между техническим состоянием изделия и техническим состоянием, определённым на основе технической документации».

Очевидно, что основное различие между приведёнными определениями понятия *тест* заключается во включении в него во втором определении «последовательности выходных реакций». Однако не зная ожидаемых выходных реакций объекта (значений функции g), характеризующих некоторое множество M его состояний, невозможно идентифицировать диагностируемый объект с представителем $M_k \in M$. Мощность множества M $|M|=N+1$, где N – количество предполагаемых неисправностей. При этом не обязательно иметь заранее вычисленные значения функции g. Они могут вычисляться параллельно с функцией f диагностируемого объекта, как это реализуется методом *контроля по образцу*.

В отличие от теста контроля (проверяющего теста) тест поиска дефекта (диагностический тест) согласно [51] различается на N пар функций (f, g_k), $k = \overline{1, N}$, а все пары при функции $(f_i, f_j) \in \mathcal{M}$. Их число равно C^2_{N+1}. Таким образом, тест поиска дефекта представляет собой последовательность элементарных тестов, различающих пары функций из множества $\mathcal{M}$.

Приведенный анализ понятия *тест* позволяет признать его определение, сформулированное в [133], более соответствующим действительности.

3.4.6. Дефект, неисправность

В силу несущественного различия определений этих терминов, приведенных в разделе 1.1, они часто используются как синонимы. Для выявления различия между этими понятиями следует отметить, что термин *неисправность* в терминологическом стандарте [128] определён как краткая форма термина *неисправное состояние*. Последнее, выраженное через измеренные значения параметров, является отображением некоторой аномалии свойств объекта в область значений параметров.

Таким образом, неисправность вторична по отношению к источнику её возникновения. Последний же и резонно назвать *дефектом*, определив его как в [128]. Как следует из изложенного, эти понятия находятся в каузальном (причинно-следственном) отношении. Относительно содержания понятие *дефект* является более общим, поскольку через него может быть определенно понятие неисправность как «дефект, выраженный через контролируемые параметры изделия».

3.5. Подсистемы понятий технического диагностирования ВС

3.5.1. Техническое диагностирование ВС

По степени детальности разбиения множества технических состояний (на два или более классов) оно делиться на *контроль технического состояния* и на *поиск дефектов* (ошибок). Относительно условий применения (в рабочем или тестовом режиме) оно делится на *рабочее* и *тестовое*. Относительно распознаваемых свойств объекта оно делится на *функциональное*, *физическое* (или *параметрическое*) и диагностирование *внешнего вида*.

Как процесс, диагностирование может быть охарактеризовано следующими видовыми отличиями: *последовательное* и *параллельное*, *внешнее* и *внутреннее*, *индивидуальное* и *групповое*, *периодическое* и *постоянное*, *статическое* и *динамическое*, *синхронное* и *асинхронное*. Однако не все из них применимы к диагностированию.

Периодическое и постоянное диагностирование заменены в технической диагностике близкими, но более точными по смыслу понятиями *тестовое* и *рабочее* диагностирование. Понятия *последовательное* и *параллельное*, *синхронное* и *асинхронное* диагностирование также не используются, поскольку диагностирование не является основным процессом функционирования устройства.

Наиболее употребляемые видовые понятия технического диагностирования сведены в табл. 3.1. Они задают пространство всевозможных межвидовых и собирательных понятий – процессов технического диагностирования, причём часть из них может не отражать реально используемые процессы. В качестве примера приведём понятия *статический* и *динамический* контроль, получившие широкое употребление при производстве БИС. Однако по отношению к родственным понятиям – соподчиненному *поиск дефектов* и родовому – *диагностирование* видовые отличия *статический* и *динамический* не применяются. Это объясняется тем, что поиск дефектов не отождествляется

с поиском характеризующих их признаков.

Приведённые примеры показывают необходимость анализа получаемых формальным путем понятий на адекватное отражение ими реальной действительности. Этот анализ не требуется для собирательных понятий, предки которых находятся в одном фасете, поскольку любые

Таблица 3.1.

Техническое диагностирование

	Основание деления	Видовой признак
1	Степень детальности разбиения множества технических состояний	Контроль технического состояния
		Поиск дефекта
2	Вид используемых воздействий	Тестовое
		Рабочее
3	Диагностический признак (функция, параметр)	Функциональное
		Параметрическое
4	Учёт переходных процессов (статика, динамика)	Статическое
		Динамическое
5	Размещение средства диагностирования относительно объекта	Внешнее
		Внутреннее
6	Средство реализации процедуры диагнострования	Аппаратурное
		Программное
7	Количество одновременно диагностируемых объектов	Индивидуальное
		Групповое
8	Полнота охвата диагностических признаков	Полное
		Сокращённое (экспресс)

противопоставляемые видовые отличия подлежат интеграции. Например, программно-аппаратное диагностирование характеризует совмещение программных и аппаратных средств, объединяемых для проведения диагностирования. Более того, на практике трудно добиться, либо обосновать экономически реализацию одноаспектных понятий. Они всегда имеют противоположные «примеси» подобно химическим элементам, не встречающимся в природе в чистом виде.

3.5.2. Дефект, неисправность

Объектом технического диагностирования являются искажения свойств изделия, характеризуемые через измеряемые параметры множеством *технических состояний*. Последние в зависимости от роли искажённых свойств в функционировании изделия делятся на состояния *неправильного* функционирования (искажение функций), *неработоспособные* (искажение обеспечивающих функцию свойств) и *неисправные* (отклонения в конструктивном оформлении и внешнем виде, например в маркировке). Все они сопоставляются с различного рода неисправностями. По отношению к их обнаружению контроль технического состояния делится соответственно на контроль *правильности функционирования*, *работоспособности* и *исправности*.

Основные видовые отличия первичных искажений – дефектов сведены в табл. 3.2. В табл. 3.2а детализируется видовое понятие *кратный дефект* на доминирующий и недоминирующий. Поскольку неисправность представляет собой проявление дефекта через внешние характеристики объекта, приведенные видовые отличия применимы и к ней.

3.5.3. Средства технического диагностирования

Отвечая на вопрос «с помощью чего диагностируется объект?», разобьём средства диагностирования на *аппаратные*, *программные* и *информационные*. Относительно степени участия человека в процессе диагностирования средства делятся на *ручные*, *автоматизированные* и *автоматические*. Дальнейшая детализация понятий, характеризующих средства диагностирования, выполняется путём использования видовых

Таблица 3.2.

Дефект

	Основание деления	Видовой признак
1	Момент возникновения	Существующий
		Предполагаемый
2	Характер проявления во времени	Устойчивый
		Неустойчивый
3	Количество дефектов, одновременно присутствующих в изделии	Одиночный
		Кратный
4	Возможность различения	Различимый
		Неразличимый
5	Форма проявления	Функциональный
		Параметрический

Таблица 3.2а

Кратный дефект

1	Отношение доминирования между составляющими кратного дефекта	Доминирующий
		Не доминирующий

понятий технического диагностирования. В зависимости от применяемого метода технического диагностирования – тестового или рабочего аппаратура различается на *тестер* и *контрольное устройство*, а программы и информация – на *тестовую* и *контрольную*.

3.5.4. Диагностическая информация

К результирующей диагностической информации относятся заявленные в процессе диагностирования отклонения наблюдаемых признаков. К первичной диагностической информации относятся диагностические модели, используемые для построения процедур диагностирования, воздействия, реакции и базы сравнения, используемые в процессе диагностирования. Видовые отличия диагностической информации сведены в табл. 3.3. Часть из них детализируется в таблицах 3.3а, 3.3б, 3.3в. В табл. 3.3а межвидовое понятие *вспомогательная выходная информация* (база сравнения) детализируется на *прямую* и *дополняющую*. Прямой результат вычислений, используемый в качестве базы сравнения, совпадает с основным, а дополняющий – дополняет основной результат до некоторого модуля исчисления.

В табл. 3.3б детализируется преобразование базы сравнения относительно содержания, либо кода информации. А в табл. 3.3 в код информации подразделяется на *разделимый* и *неразделимый*. Эти таблицы отражают иерархическую классификацию межвидового понятия *преобразуемая выходная информация*. Последнее получило название *сигнатуры*. Так же, как и для ранее рассмотренных понятий, видовые отличия диагностической информации задают пространство признаков для порождения межвидовых и собирательных понятий подсистемы, которые делятся на допустимые и недопустимые.

3.5.5. Диагностируемость

Это понятие характеризуется приспособленностью объекта к выполнению процедуры его диагностирования.

Относительно процессов диагностирования – контроля технического состояния и поиска дефектов разделим свойства ОД, характеризующие приспособленность последнего к их выполнению на *контролепригодность* и *поископригодность*. Эти понятия связаны тем же видом связей, что и породившие их процессы. В зависимости от вида диагностирования – тестового или рабочего – диагностируемость можно разделить на

Таблица 3.3.

Диагностическая информация

	Основание деления	Видовой признак
1	Направленность потока информации	Входная
		Выходная
2	Источник получения информации	Экспериментальная
		Заданная
3	Наличие преобразования	Непреобразованная
		Преобразованная
4	Наличие ограничений	Разрешённая
		Запрещённая
5	Назначение	Рабочая
		Тестовая
6	Время получения	Предварительная
		Текущая
7	Алгоритм вычисления	Основная
		Вспомогательная

Таблица 3.3а.

Вспомогательная выходная информация

1	Соотношение результатов вычислений – основного и вспомогательного	Прямая
		Дополняющая

Таблица 3.3б.

Преобразуемая выходная информация

2	Преобразуемый признак	Содержание информации
		Код информации

Таблица 3.3в.

Кодированная информация

3	Связность основной и контрольной информации	Разделимая
		Неразделимая

тестопригодность (тестируемость) и *отказочувствительность*. Диагностируемость, обеспечиваемую за счёт внешних (внутренних) средств, резонно назвать соответственно *внешней* (*внутренней*).

Очевидно, что широко известное понятие *отказоустойчивость* является потомком двух понятий смежных дисциплин – *восстанавливаемости* (надёжность) и *отказочувствительности* (техническая диагностика).

3.5.6. Тест диагностирования

Поскольку тест реализует процедуру тестового диагностирования, к нему применимы все видовые отличия этой процедуры. Например, тест подразделяется на тест *контроля* и *поиска дефектов*, *функциональный* и *параметрический*, *статический* и *динамический*. Это разбиение при необходимости можно продолжить.

Приведённые подсистемы понятий технического диагностирования ВС не претендуют на полноту. Они иллюстрируют метод порождения понятий, который можно использовать при развитии системы – синтезе новых понятий, анализе и пересмотре существующих [85-87].

Разработанная система понятий *Техническое диагностирование* положена в основу Отраслевого и Государственного терминологических стандартов [133, 131].

Глава 4. СИСТЕМА ДИАГНОСТИЧЕСКИХ МОДЕЛЕЙ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

4.1. Свойства ВС

Свойства объекта диагностирования проявляются при его взаимодействии с другим объектом либо субъектом действия. В [88] на самом общем, философском, уровне выделяются следующие свойства (аспекты) системы [67] – *функциональные*, *структурные* и *субстациональные* (вещественные). Первые характеризуют роль и место системы среди других объектов в окружающем мире, вторые – связи между составляющими её элементами, а третьи – вещественный состав (природу) системы[6]. Очевидно, что эта триада не исчерпывает основных свойств реальных объектов. Расширим выделенную совокупность свойств, представляя их диадами, реализующими метод противопоставления (оппозиции). Приведём их без какого-либо обоснования, поскольку они являются объектом рассмотрения философских и системологических работ [58, 89-91]. К наиболее общим взаимно независимым относятся следующие свойства:

пространство – время;
функция – структура;
вещество – энергия;
энергия – информация;
внутреннее – внешнее.

Перечисленные философские категории представляют собой первичные свойства системы. Они последовательно уточняются с применением общенаучных и специальных понятий.

Пространственный аспект детализируется следующими парами противоположных признаков: *двух* и *n-мерное* ($n>2$) пространство, в прямоугольных и не прямоугольных координатах, *разомкнутое* и *замкнутое* пространство и т.д.

Временной аспект детализируется парами: *периодический – постоянный*, *статический – динамический*, *синхронный – асинхронный*.

Основные функции систем – *преобразование* и *хранение*, *передача* и *коммутация*, *управление* и *исполнение*. Они инвариантны по отношению

[6] По преимущественному изучению искажений перечисленных свойств объекта диагностика делится на функциональную, структурную и физическую. Эти её разделы преобладают соответственно в распознавании образов, диагностировании ВС и неразрушающем контроле.

к массе, энергии, информации. Дальнейшая детализация функций предполагает учёт специфики объекта. Например, преобразование информации в ЭВМ делится на *арифметическое* и *логическое*.

Структура объекта характеризуется следующими парами признаков: последовательная – параллельная, внутренняя – внешняя, с расходящимся и сходящимся путями и т.д.

Свойство управления проявляется во взаимодействии субъекта и объекта управления. Первый из них играет активную роль в управлении, а второй – пассивную.

Вещественный состав объекта может меняться в широком диапазоне, имеющем свою иерархию разбиения.

Таким образом, свойства системы – оригинала, находятся в отношении частичного порядка. Он устанавливается путем философско-методологического анализа основных свойств объекта и образует некоторое множество семантических сетей, характеризующих связи между ними, причём верхние части сетей включают свойства, общие для объектов различного назначения, а нижние – специфические свойства этих объектов.

Специфические предметные свойства ЭВМ, как объекта диагностирования, детализируют её архитектуру, способы управления и обработки информации, аппаратную и программную составляющие и т.д. В частности, аппаратура ЭВМ характеризуется элементной базой и конструктивным оформлением.

Наряду с аналитическим подходом к ЭВМ, заключающимся в анализе отдельных её свойств, при проектировании и диагностировании ЭВМ получил широкое распространение синтетический подход [92]. Он заключается в комплексном рассмотрении свойств ЭВМ на определённом уровне детальности описания, присущей как аппаратной, так и программной составляющей ЭВМ.

В порядке убывания детальности описания различают следующие уровни логического описания ЭВМ сетевой архитектуры:

- вентильный (логических элементов);
- функциональных узлов (ячеек);
- регистровый (совокупность однородных ячеек);
- функциональных устройств (МПУУ, АЛУ, ОЗУ, ПЗУ и др.);
- процессорный;

- многопроцессорный;
- сетевой.

Программное обеспечение МДА различается на следующие уровни детальности описания вычислительных процессов:

- микрокомандный (программно-управляемых сигналов);
- командный (макропрограммный);
- программный на языке Ассемблера;
- программный на языке высокого уровня;
- сетевой на языке низкого уровня;
- сетевой на языке высокого уровня;
- системный.

Естественно, что переход к менее детальному описанию позволяет без существенного изменения объёма информации представлять больший объём аппаратного и программного обеспечения ЭВМ. Это свойство моделей используется в технологии проектирования ЭВМ «сверху-вниз».

К специфическим свойствами объекта диагностирования относятся *искажение* заданного свойства и *порядок* искажений. Искажение, безусловно, является, нежелательным свойством ОД, но при определён-ных условиях неизбежным. К последним относится влияние внутренних факторов ОД и факторов внешней среды. Порядок искажений ОД может определяться как вероятностью их возникновения, так и маскированием одного искажения другим. Интерпретация искажения определяется особенностью ОД. Применительно к аппаратуре оно называется *неисправностью*, по отношению к программе – *ошибкой*, по отношению к результату обработки информации – *ошибкой вычислений*. Искажения свойств ЭВМ отражаются в их диагностических моделях.

4.2. Отражение свойств в формальных моделях

Примем следующее определение модели [87]: моделью некоторой системы-оригинала называется система, обладающая каким-то одинаковыми на одном фиксированном уровне детальности свойствами по сравнению с системой-оригиналом и являющаяся более простой, чем последняя. Это определение справедливо как для содержательных

(предметных), так и для формальных моделей. Центральным понятием в определении являются свойства системы. В зависимости от количества одновременно отражаемых свойств модели называются одно-, двух- и трех-аспектными. Модели реальных систем, как правило, являются многоаспектными, поскольку наряду с перечисленными отражают специфические свойства объекта.

К специфическим свойствам модели относятся *полнота*, *детальность* и *форма* отражения свойств системы-оригинала.

Математические модели отражают свойства предметных моделей с помощью переменных и отношений между ними. По форме представлений они различаются на *аналитические*, *табличные* и *графовые*. Связь между переменными математической модели и свойствами предметной модели устанавливаются с помощью функций интерпретации.

Одной из наиболее общих формальных моделей, пригодных для отражения свойств вычислительного процесса и операционной среды на различных уровнях представления ЭВМ, является алгебраическая структура (59) $\mathcal{M} = <A, C, F, P>$. Она относится к классу теоретико-множественных моделей, поскольку четвёрка её символов представляет собой множества:

A – предметных переменных или носитель;

C – констант;

F – функций;

P – предикатов.

Алгебраическая структура является моделью языка первого порядка Ω. Её элементы – суть результаты интерпретации этого языка с применением функций D, $Cnst$, Fn, Pr:

D: $\pi \to A_\pi$, $\pi \in Srt$;

$Cnst$: $cnst \to c$;

Fn: $\to f$;

Pr: $\to p$.

Элемент π множества Srt называется сортом объекта. Для каждого сорта фиксируется набор предметных переменных $a^\pi_1,...,a^\pi_n \in A_\pi$ и констант $c^\pi_1,...,c^\pi_k \in C_\pi$. Каждому функциональному символу fn сопоставляется n-местная функция f: $A_1 \times ... \times A_n \to A$, а каждому предикатному символу pr – $n+1$-местный предикат p: $A_1 \times ... \times A_{n+1} \to B$,

B={0,1}. Для порождения моделей с заданными свойствами примем следующую нотацию алгебраической структуры:

$$\mathcal{M} = <A, F, R>,$$

именуемую в [59] алгебраической системой. В ней опущен несущественный для отражения свойств символ констант C, а символ предикатов заменён на символ отношений R. Эта замена правомерна с силу тождественности k-местного отношения $R_i \subseteq A_{i1} \times ... \times A_{ik}$ и k-местного предиката $P_i : A_{i1} \times ... \times A_{ik} \to B$, где B={0,1}.

В [59] алгебраическая система членится на алгебру $\mathcal{A}$=<A, F> и модель или реляционную систему $\mathcal{B}$=<A, R>, что соответствует аксиоме членения (2.30). Примем эти части алгебраической системы за исходные – родовые модели, которые будем использовать для порождения теоретико-множественных моделей с заданными содержательными свойствами. Порождаемые – видовые модели наследуют свойства родовых моделей

$$\mathcal{A} = <\mathcal{A}, \mathcal{F}> \text{ и } \mathcal{B} = <\mathcal{A}, \mathcal{R}>.$$

Согласно [65] каждое свойство системы-оригинала выражается унарным отношением. Это означает, что содержательные свойства системы-оригинала должны отражаться сигнатурами F и R моделей $\mathcal{A}$ и $\mathcal{B}$. В соответствии с этим одноаспектные модели системы-оригинала порождаются путём интерпретации сигнатур F и R одним свойством, а многоаспектные – их интерпретацией несколькими свойствами. При последовательной интерпретации с добавлением дополнительного свойства каждая последующая модель может считаться видовой по отношению к предыдущей, поскольку помимо свойств предыдущей модели она обладает дополнительным видом отличием. В теоретико-множественной модели ему соответствует специальный символ. Согласно утверждению 2.9 он может быть найден с использованием аксиом и правил вывода теории $\mathrm{Th_a}$.

Содержательные свойства могут придаваться как элементам носителя, так и сигнатуры. Для этого в модели должен присутствовать наряду с символом унарного отношения r_k, отражающим k-е свойство объекта, символ принадлежности k-го свойства $r_{\in k} \in R_{\in}$. Он интерпретируется двухместным отношением $r_{\in k} \subseteq A_{i1} \times r_k$, отражающим k-е свойство носителя или $r_{\in k} \subseteq r_i \times r_k$, отражающим i-е свойство символа сигнатуры,

причём символ r_i сам может интерпретироваться многоместным отношением, т.е. представлять собой многомерное множество.

Синтез моделей, как и программ, возможен «снизу-вверх» и «сверху-вниз». По отношению к понятиям это соответствует различному направлению обхода триады «единичное-особенное-всеобщее». Обход её «слева-направо» означает движение от частного к общему, а «справа-налево» – наоборот.

Проиллюстрируем обход триады «слева-направо» следующим примером. Распространённым объектом моделирования являются управляющие входы блоков и узлов ЭВМ. При построении модели могут возникать вопросы типа «является ли рассматриваемая связь входом блока?» или «является ли вход управляющим?». Очевидно, что решить этот вопрос в рамках конкретной модели без привлечения дополнительных понятий не представляется возможным. Это вполне согласуется с теоремой Гёделя о полноте, гласящей в одном из изложений, что в рамках любой теории всегда найдутся утверждения, истинность которых невозможно ни доказать, ни опровергнуть средствами данной теории.

Для уточнения понятия «управляющий вход» необходимо рассмотреть более общие понятия «вход» и «управление». В свою очередь, первое из них относится к понятию «структура», а второе – к понятию «функция». Являясь наиболее общими, последние понятия сами являются предметом философского анализа путём сопоставления их с опытом.

Другой подход заключается в последовательном расширении состава рассматриваемых свойств объекта моделирования. Естественной последовательностью является переход от функции к структуре, а затем к их видам – управлению и входу.

Естественно предположить, что, как и в программировании, наиболее эффективно сочетание обеих подходов. Подход от общего к частному позволяет установить иерархию свойств модели и её связь с другими моделями. Подход от частного к общему более конкретен и конструктивен.

4.3. Теоретико-множественные модели ОД

4.3.1. Функциональная модель (*Ф-модель*)

Она формируется на основе алгебры $\mathcal{A}=<A, F>$, путём деления носителя A на два множества X и Y, интерпретируемых соответственно значениями входных и выходных переменных: $A = X \cup Y$.

В частном случае $A = X = Y$. Функция $f_i \in F$ отображает элементы множества X в элементы множества Y f_i: $X \rightarrow Y$. В другой форме это отображение записывается в виде функциональной зависимости $Y= f_i (X)$, которая интерпретируется моделью «чёрного ящика». С учётом введённых символов и их интерпретаций функциональная модель ОД описывается тройкой

$$\mathcal{M}_{\text{ф}}=< X, Y, F>.$$

Примером содержательной интерпретации Ф-модели может являться логический элемент, внутреннее устройство которого на уровне логической схемы не представляет интерес. Таким образом, Ф-модель характеризует минимальный уровень детальности представления рассматриваемого объекта. Следовательно, в зависимости от степени детальности рассмотрения аппаратуры и программ ЭВМ Ф-модель может быть использована для описания любого из ранее перечисленных уровней представления аппаратуры и программ. Каждому уровню соответствует определенный сорт π Ф-модели. Вместе они описываются многосортной Ф-моделью.

Мощность множеств X и Y определяет количество физических входов и выходов объекта. В частном случае, например на уровне функциональных устройств ЭВМ (регистровом уровне), мощность этих множеств совпадает: $X = Y = B^n$, где $B=\{0,1\}$. Здесь n – число разрядов в разрядной сетке (регистрах ЭВМ). Количество физических входов n и выходов m в общем случае определяются по формулам $n = log_2| X |$ и $m = log_2| Y |$. Выделению физических входов и выходов Ф-модели соответствует отражение внешнего структурного аспекта «чёрного ящика». Отражение внутреннего структурного аспекта связано с раскрытием «чёрного ящика».

4.3.2. Структурная модель (***С-модель***)

Она формируется на основе модели $\mathcal{B}=<A,\ R>$. В ней символ A интерпретируется элементами системы $A_э$, а R – связями между ними – $R_и$. Здесь $R_и$ – множество различных видов двухместных отношений инциденции вида $R_{иi} \subseteq A \times A$, $R_{иi} \in R_и$. Интерпретированная таким образом модель $\mathcal{B}$ представляет собой С-модель:

$$\mathcal{M}_с=< A_э, R_и >.$$

Если, например, в качестве элементов системы рассматриваются функциональные устройства или узлы ЭВМ, то множество $R_и$ включает отношения, характеризующие одиночные и групповые, одно и двунаправленные связи между ними. Если в качестве элементов системы рассматриваются программные элементы динамической автоматной сети МДА, то множество $R_и$ включает отношения, характеризующие двунаправленные связи и ссылки между ПЭ. Таким образом, С-модель отражает пространственную структуру объектов, как материальных (схем вычислителя), так и идеальных (схем решения задачи). Индивидуально элементы в С-модели не интерпретируются.

Очевидно, что теоретико-множественная С-модель изоморфна модели графа $G=<V,\ E>$, в которой множества V и E интерпретируются соответственно вершинами и дугами некоего графического образа.

4.3.3. Функционально – структурная модель (ФС-модель)

Её можно назвать также моделью функционирования, поскольку она отражает структурно-временной аспект или поведение объекта во времени. С этой целью носитель A алгебры $\mathcal{A}=<A,\ F>$ разбивается на три подмножества: $A = X \cup Y \cup Q$, интерпретируемых состояниями объекта – *входными*, *выходными* и *внутренними* соответственно. Символ F разделяется на два: $F_п$ и $F_в$, $F = F_п \cup F_в$, интерпретируемых соответственно множествами функций переходов и выходов.

Таким образом, ФС-модель представляет собой следующую пятерку символов:

$$\mathcal{M}_{фс}=< X, Y, Q, F_п, F_в >.$$

Для случая дискретных функций $f_п \in F_п$ и $f_в \in F_в$ эта пятерка описывает поведение конечного автомата (КА) [47].

$f_в: X \times Q \to Q;$

$f_п: X \times Q \to Y$ *(автомат Мили);*

$f_п: Q \to Y$ *(автомат Мура).*

Последовательность тактов времени в формулах языка КА отражается дополнительными символами автоматного времени t:

$$q(t)=f_п(q(t–1), x(t–1));$$

$$y(t)=f_в(q(t), x(t));$$

$$y(t)=f_в(q(t));$$

В том случае, когда множества $F_п$ и $F_в$ не одноэлементны, ФС – модель описывает многопрограммный КА.

С учётом символа констант Θ, интерпретируемого вектором параметров (коэффициентов линейного дифференциального уравнения) и при условии непрерывности и дифференцируемости функций $f_п \in F_п$ и $f_в \in F_в$ модель $\mathcal{M}_{фс}$ описывает функционирование непрерывного динамического объекта. Так же, как и для конечного автомата, эта модель детализируется на уравнения состояния системы и её выходов:

$$Q = f_п(X, Q, \Theta, t), \quad Y = f_в(X, Q, \Theta, t).$$

4.3.4. Структурно-функциональная модель (СФ-модель)

СФ-модель получается путём деления символа R модели $\mathcal{B}=<A, R>$ на три символа: $R = R_и \cup R_f \cup R_\in$, интерпретируемых соответственно двухместным отношением инциденции, унарным отношением функционального базиса и двухместным отношением принадлежности. Последние два символа являются дополнительными по отношению к С-модели. Носитель A интерпретируется множеством элементов модели $A_Э$.

Таким образом, СФ-модель описывается следующей четверкой символов: $\mathcal{M}_{сф}=< A_Э, R_и, R_f, R_\in >$.

Отношения $r_{\in i}$ и $r_{\in j}$, $r_{\in i}, r_{\in j} \in R_\in$ характеризуют совокупности элементов $A_{Эi} \in A_Э$ и связей $R_{иj} \in R_и$, реализующих соответственно i-ю и j-ю функции из функционального базиса R_f: $f_i, f_j \in R_f$:

$$r_{\in i} \subseteq A_Э \times f_i, \qquad r_{\in j} \subseteq R_и \times f_j.$$

Примером содержательной интерпретации СФ-модели является сеть функциональных элементов любой степени сложности (блок-схема, логическая сеть, схема решения задачи и т.д.).

Если функциональный базис однороден, то СФ-модель является *гомогенной*, а в противном случае – *гетерогенной* [88]. Неоднородный функциональный базис делится на группы однородных элементов. Свойство однородности является предметом соглашения. Например, если входные переменные функциональной схемы принять за вырожденные функции, то её модель следует считать гомогенной. В противном случае СФ-модель гетерогенна, поскольку переменные и функции, сопоставляемые элементам С-модели, принадлежат разным группам из R_f.

СФ-модель отражает статические отношения между функциональными элементами, ибо она не описывает их взаимодействие во времени.

4.3.5. Функциональная модель с управлением (*ФУ-модель*)

Свойство управления проявляется во взаимодействии субъекта и объекта управления. Первый из них играет активную роль в управлении, а второй – пассивную.

ФУ-модель управляющей системы формируется на основе Ф-модели путём интерпретации символов X и Y – значениями условий управления X_p и управляющих воздействий Y_c соответственно, а символа F – множеством функций управления F_c. С учётом этой интерпретации ФУ-модель управляющей системы описывается тройкой:

$$\mathcal{M}_{\text{фу,а}} = < X_c,\ Y_c,\ F_c >.$$

ФУ-модель управляемой (исполнительной) системы формируется на основе Ф-модели путём следующего деления символов:

$$X = X_d \cup X_p,\ \ Y = Y_d \cup Y_p,\ \ F = F_d \cup P,$$

Введённые символы интерпретируется следующим образом:

X_d – значения входных данных;

X_p – значения управляющих воздействий;

Y_d – значения выходных данных;

Y_p – значения признаков данных;

F_d – функции обработки данных;

P – признаки данных.

Признак данных представляет собой функцию вида p: $B^n \to B$, $B^n = X_d$, $B^n = \{0, 1\}$, т.е. предикат. Например, множество арифметических предикатов $P_A = \{=, \neq, <, >, \leq, \geq\}$.

Таким образом, ФУ-модель управляемой системы имеет вид:

$$\mathcal{M}_{\text{фу,п}} = < X_d, X_c, Y_d, Y_p, F_d, P >.$$

ФУ – модель, описывающая управляющую и управляемую системы во взаимосвязи, естественно, объединяет модели $\mathcal{M}_{\text{фу,а}}$ и $\mathcal{M}_{\text{фу,п}}$ с помощью пар символов $X_p = Y_p$, $X_c = Y_c$:

$$\mathcal{M}_{\text{фу}} = << X_c, Y_c, F_c > < X_d, X_c, Y_d, Y_p, F_d, P >>.$$

Эта модель иллюстрируется следующей схемой (рис. 4.1):

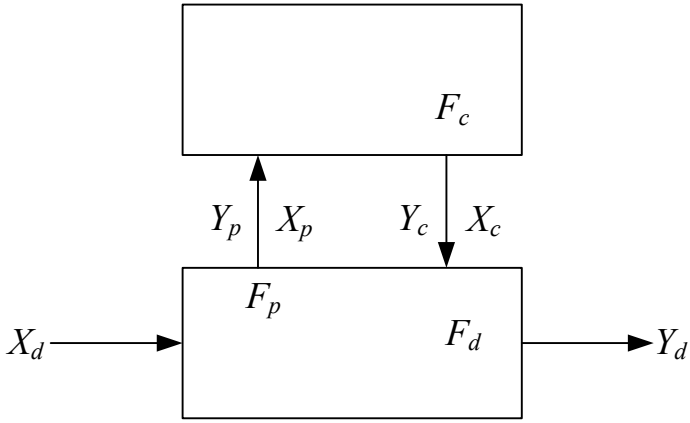

Рис. 4.1. Графическая модель объединенной управляющей и управляемой систем.

4.3.6. Функционально-структурная модель с управлением (ФСУ – модель)

Активная и пассивная ФСУ-модели образуются путём объединения свойств ФС-модели со свойствами $\text{ФУ}_\text{а}$-модели и $\text{ФУ}_\text{п}$-модели:

$$\mathcal{M}_{\text{фсу,а}} = < X_p, Y_c, Q_c, F_{\text{п},c}, F_{\text{в},c} >.$$

$$\mathcal{M}_{\text{фсу,п}} = < X_d, X_c, Y_d, Y_p, Q_d, Q_p, F_\text{п}, F_\text{в}, P >.$$

Объединение свойств выполняется не механическим объединением символов исходных моделей, а с применением интерпретации и операций деления и объединения теории Th_a.

4.3.7. Структурная модель с управлением (СУ – модель)

СУ-модель получается путем деления символа R модели $\mathcal{B}=<A, R>$ на четыре: $R = R_и \cup R_{cd} \cup R_{ap} \cup R_{\in}$, интерпретируемых соответственно двухместным отношением инциденции, двумя унарными отношениями – управляющей информации и данных, и активности-пассивности и двух (трёх)-местным отношением принадлежности. С помощью отношений принадлежности $R_{c,i}$, $R_{d,j}$, $R_{ca,k}$, $R_{cp,l} \in R_{\in}$ свойства R_{cd} и R_{ap} присваиваются связям между элементами – элементам двухместного множества $R_и$:

$$R_{c,i} \subseteq R_u \times R_{cd},$$
$$R_{d,j} \subseteq R_u \times R_{cd},$$
$$R_{ca,k} \subseteq R_u \times R_{cd} \times R_{ap},$$
$$R_{cp,l} \subseteq R_u \times R_{cd} \times R_{ap}.$$

Два двухместных отношения $R_{c,i}$, $R_{d,j}$, включают соответственно управляющие и информационные связи между элементами сети, а два трёхместных отношения $R_{ca,k}$, $R_{cp,l}$ разделяют управляющие связи на активные и пассивные.

Таким образом, $\mathcal{M}_{су} = < A_э, R_и, R_{cd}, R_{ap}, R_{\in} >$.

4.3.8. Структурно-функциональная модель с управлением (СФУ - модель)

СФ-модель образуется путём деления символа R модели $\mathcal{B}=<A, R>$ на пять: $R = R_и \cup R_f \cup R_{cd} \cup R_{ap} \cup R_{\in}$, Дополнительные по отношению к сигнатуре СФ-модели $\mathcal{M}_{сф}=<A_э, R_и, R_f, R_{\in}>$ символы R_{cd} и R_{ap} отражают свойства управляющей информации и данных, и активности-пассивности в управлении. Как и в С–модели с помощью отношений принадлежности $R_{c,i}$, $R_{d,j}$, $R_{ca,k}$, $R_{cp,l} \in R_{\in}$ эти свойства присваиваются элементам

двухместного множества $R_{и}$:

$$R_{c,i} \subseteq R_{и} \times R_{cd},$$
$$R_{d,j} \subseteq R_{и} \times R_{cd},$$
$$R_{ca,k} \subseteq R_{и} \times R_{cd} \times R_{ap},$$
$$R_{cp,l} \subseteq R_{и} \times R_{cd} \times R_{ap}.$$

Дополнительно с помощью отношений $R_{ca,s}$ и $R_{ca,t}$ свойство активности-пассивности может быть присвоено элементам одноместного множества $A_{э}$ наряду с присущими этим элементам функциями из функционального базиса R_f:

$$R_{ca,s} \subseteq A_{э} \times R_f \times R_{ap},$$
$$R_{ca,t} \subseteq A_{э} \times R_f \times R_{ap}.$$

Таким образом, $\mathcal{M}_{сфу} = < A_{э}, R_f, R_{и}, R_{\in}, R_{cd}, R_{ap} >$.

Содержательно отношения $R_{ca,s}$ и $R_{ca,t}$ интерпретируются как множества функциональных элементов (элементов с известными функциями по сравнению с СУ-моделью), обладающих свойством активности (пассивности) в управлении, а $R_{c,i}$, $R_{d,j}$, $R_{ca,k}$, $R_{cp,l}$ – как множества связей между элементами, передающих управляющую (активную и пассивную) и обрабатываемую информацию.

4.4. Связь между моделями

Рассмотренные модели из большого разнообразия свойств отражают в различном составе следующие основные свойства вычислительных систем – функцию, пространственную и временную структуру и управление [93]. Каждое свойство в теоретико-множественной модели выражается соответствующим символом. Следовательно, увеличение числа отраженных свойств сопровождается увеличением числа символов в модели [94]. Это соответствует увеличению числа признаков, характеризующих содержание понятие. В процессе синтеза модели с заданными свойствами базовое (родовое) свойство, присущее классу моделей, дополняется видовыми отличиями, характеризующими синтезируемую модель.

При порождении многоаспектных моделей (видовых понятий) могут использоваться два принципа. Первый из них основывается на возможности последовательного выделения требуемых признаков из исходного неопределенного множества признаков (принцип внутренних возможностей). Другой принцип заключается в присовокуплении к сигнатуре исходной модели внешних признаков (принцип внешней дополнительности). Первый принцип был использован при порождении многоаспектных моделей на основе алгебры и реляционной системы, а второй при порождении видовых понятий (в главе 2). Оба они дают одинаковые результаты, поскольку источник поступления признаков – внутренний неявный перечень или внешняя среда, несущественен.

Принципиальным при порождении модели является то, что вся раскрытая совокупность свойств оказывается содержанием полученной модели, в силу чего в соответствии с аксиомами Th_a она относится к видовому или межвидовому понятию. В том случае, когда свойства распределяются по разным моделям, это соответствует членению исходной модели и образованию моделей-частей.

На рис. 4.2 представлены связи между рассмотренными моделями. На нём пунктирными прямоугольниками сеть моделей разделена на три части. В верхний прямоугольник заключена партитивная полурешётка, состоящая из моделей $\mathcal{M}$, $\mathcal{A}$, $\mathcal{B}$. Последние две из них является частями первой, которая по содержанию признаков представляет собой минимальный элемент решётки. Модели, заключенные в два других прямоугольника, представляют собой две *родо-видовых решётки*. Обе они имеют максимальный и минимальный по числу символов элементы.

Приведённая сеть моделей обладает способностью к расширению при синтезе моделей, отражающих большее число свойств.

Утверждение 3.1. Минимальный базис моделей, представляющих объект диагностирования без очередей на входах его элементов, включает ФС и СФ-модели, отражающие принятый уровень детальности описания.

Действительно для построения диагностических процедур необходимо знать с одной стороны закон функционирования, а с другой стороны структуру ОД. Первый описывается соответствием между входными и выходными состояниями ОД и последовательностью их смены, а именно ФС-моделью. Вторая характеризует ОД как совокупность взаимосвязанных его частей, каждая из которых и является

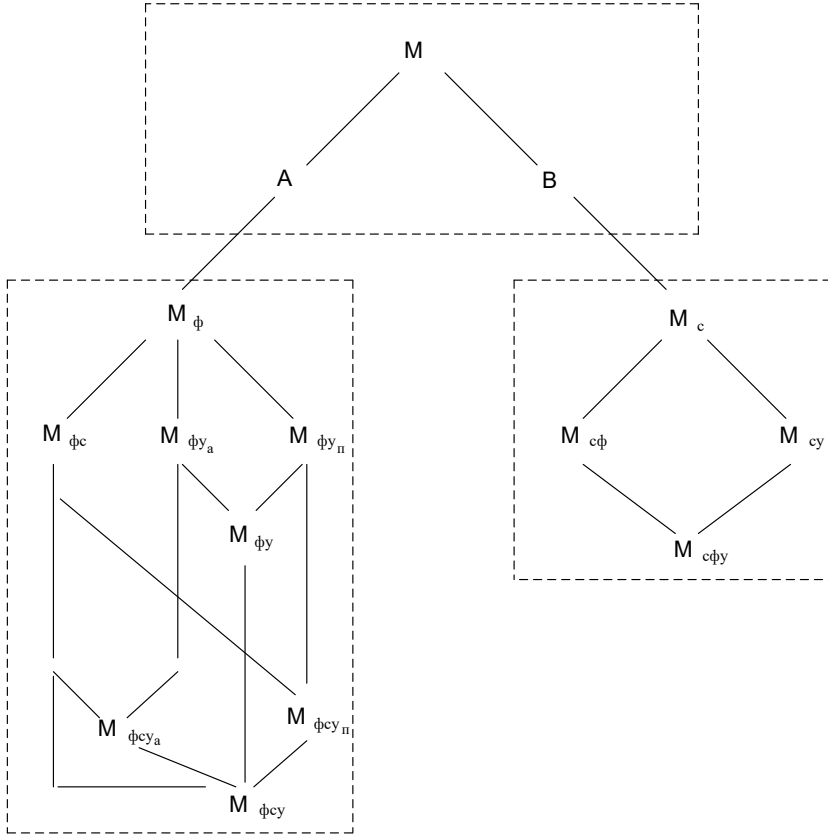

Рис. 4.2. Диаграмма связей моделей ОД

конечной целью диагностирования. Структура ОД описывается СФ-моделью. Таким образом, ФС и СФ-модели необходимы и достаточны для описания ОД с заданным ограничением, что и требовалось доказать.

Очереди на входах элементов ОД требуют дополнительной информации для отражения их в модели. В качестве её используются маркирования и связь между ними в сетях Петри. Из этого следует большая общность указанной модели.

Помимо свойств, отражаемых этими моделями, построение диагностических процедур может потребовать отражения дополнительных свойств, например «управления-исполнения». Модели, отражающие эти свойства, представляют собой совокупность видовых моделей по отношению к ФС- и СФ-моделям.

4.5. Общие диагностические модели ВС

Диагностическая модель описывается каркасом [67]: $\mathcal{D}=<\mathcal{M}, \Sigma_u, \Phi>$. Здесь $\mathcal{M}$ – теоретико-множественная модель ОД, Σ_u – сигнатура диагностической модели ОД, Ф – множество аксиом, предопределяющих использование сигнатуры Σ_u по отношению к модели $\mathcal{M}$.

Моделью $\mathcal{M}$ может являться любая видовая многоаспектная модель, построенная на основе родовых моделей $\mathcal{A}$ и $\mathcal{B}$. Сигнатура $\Sigma_u=<R_u, R_{ord}>$, включает два символа R_u и R_{ord}. характеризующие специфические свойства диагностической модели ОД – *искажения* и *порядок* искажений.

Искажение представляет собой замену одного элемента рассматриваемого множества теоретико-множественной модели ОД на другой. Применительно к носителю и сигнатуре выделяется две модели искажений ОД:

$$r_{aj} \subseteq A_э \times a_j, \qquad r_{aj} = a_i \,||\, a_j, \qquad a_i, a_j \in A_э,$$
$$r_{\sigma j} \subseteq \Sigma \times \sigma_j, \qquad r_{aj} = \sigma_i \,||\, \sigma_j, \qquad \sigma_i, \sigma_j \in \Sigma.$$

Символ, стоящий слева от знака $||$, означает *заменяемый* элемент носителя или его свойство, а стоящий справа – *заменяющий*.

Если *j*-е искажение присуще всем элементам или свойствам модели, то его модель сводится к декартовому произведению:

$$A_э \times a_j, \quad \Sigma \times \sigma_j.$$

Ограничения искажений элементов и свойств обуславливаются

либо имеющимся опытом их регистрации, либо их физической осуществимостью.

Обычно предполагается, что множества A и Σ *замкнуты* относительно искажения $r_{u,k} \in R_u$. Это означает, что заменяемый и заменяющий элементы принадлежат одному множеству. Однако это допущение не всегда справедливо и его необходимо обосновывать в каждом конкретном случае. Обычно также рассматривают одиночные искажения любого вида. Однако в реальных объектах нередко имеют место кратные (множественные) искажения.

Отношение R_{ord} устанавливает порядок следования (нумерацию [29]) искажений объекта, $R_{ord} \subseteq r_{u,k} \times N$, где N – множество натуральных чисел.

Аксиомы Ф задают способ перебора искажений ОД при построении тестов и моделирования функционирования ОД.

Каркас $\mathcal{D}$=<$\mathcal{M}$, Σ_u, Ф> описывает интенсионал диагностической теоретико-множественной модели. Экстенсионал диагностической модели получается на основе её интенсионала путём подстановки вместо первых двух символов их содержимого и выполнения аксиом Ф:

$$\mathcal{D} = \{M_0, M_1, \ldots, M_i, \ldots, M_N\}.$$

Здесь M_i – теоретико-множественная модель ОД с i-м искажением.

Применительно к аппаратуре модель M_i называется -ой неисправной модификацией объекта [51]. В экстенсионале $\mathcal{D}$ неисправные модификации упорядочены в соответствии с отношением R_{ord}. В частном случае порядок следования моделей может быть произвольным.

Рассмотрим диагностические модели, представляющие различные свойства ОД. С этой целью будем подставлять в интенсионал $\mathcal{D}$=<$\mathcal{M}$, R_u, R_{ord}, Ф> с раскрытой сигнатурой Σ_u вместо символа $\mathcal{M}$ модели, рассмотренные в раздел 4.3.

4.5.1. Диагностическая Ф – модель

Она представляется шестеркой символов

$$\mathcal{D}_{\text{ф}} = <X, Y, F, R_u, R_{ord}, \Phi_{\text{ф}}>.$$

Искажение значений переменных из множеств X и Y имеет место в том случае, когда они являются подмножествами более мощных (полных) множеств $X_{\text{п}}$ и $Y_{\text{п}}$, включающих помимо разрешённых значений из X и Y

запрещенные значения из множеств $X_з$ и $Y_з$:

$$X_n = X \cup X_з, \;\; Y_n = Y \cup Y_з.$$

В этом случае искажение заключается в замене разрешенных значений переменных на запрещённые значения, чему соответствует фактическое расширение алфавитов X и Y:

$$r_X^+ = X \cup x_з, \qquad X_з \setminus x_з,$$
$$r_Y^+ = Y \cup y_з, \qquad Y_з \setminus y_з.$$

Противоположным искажением является перевод разрешённых значений $x_i \in X$, $y_i \in Y$ в множества запрещенных значений $X_з$ и $Y_з$:

$$r_X^- = X \cup x_i, \qquad X_з \setminus x_i,$$
$$r_Y^- = Y \cup y_j, \qquad Y_з \setminus y_j.$$

Искажение самого функционального элемента заключается в замене одной функции $f_i \in F$ другой $f_j \in F_j$: $r_{fj} = f_i \,||\, f_j$. Иногда оно обозначается как переход $f_i \to f_j$.

Каноническими искажениями функций логических элементов являются константные неисправности: $r_0 = f_i \,||\, 0$ и $r_1 = f_i \,||\, 1$. Константная неисправность типа «константа 0» функционального элемента f_i с обобщёнными входом и выходом, представленная в виде замены функции f_i на 0, изображена рис. 4.3.

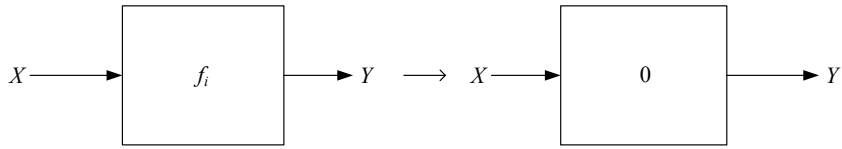

Рис. 4.3. Неисправность $f_i \,||\, 0$ логического элемента f_i.

Неисправность $f_i \,||\, 0$ проявляется наличием постоянного значения 0 на выходе логического элемента f_i при любых значениях на его входах. Неисправности входов логической схемы определяются относительно соответствующих им переменных, принимаемых за вырожденные булевы функции. Естественно, что в отличие от булевых функций логических элементов они могут заменяться только константами 0 и 1.

4.5.2. Диагностическая С-модель

Она представляется пятёркой символов

$$\mathcal{D}_{\text{с}}=<A_{\text{э}}, R_{\text{и}}, R_{u}, R_{ord}, \Phi_{\text{с}}>.$$

Искажение состава элементов С-модели заключается в изменении мощности множества $A_{\text{э}}$: $r_A = A_{\text{э}} \,||\, A_{\text{эи}}$. Ему соответствует две разновидности модели искажений:

выбывание элемента a_i из $A_{\text{э}}$: $r_A^- = A_{\text{э}} \setminus a_i$,

добавление элемента a_i в $A_{\text{э}}$: $r_A^+ = A_{\text{э}} \cup a_i$.

Искажение основного свойства С-модели – отношения инциденции $r_{ии} \in R_{\text{и}}$ заключается либо в уменьшении, либо в увеличении числа элементов этого двухместного множества. Это соответствует разрыву имеющихся, либо возникновению новых связей между парами элементов $a_{\mu}, a_{\text{v}} \in A_{\text{э}}$. Одиночное искажение связей описывается следующей моделью: $r_{ии} = R_{\text{и}} \,||\, R_{ии}$. Она имеет две разновидности, описывающие:

обрыв связи – $r_{ии} = R_{\text{и}} \setminus (a_{\mu}, a_{\text{v}})$,

возникновение новой связи – $r_{ии} = R_{\text{и}} \cup (a_{\mu}, a_{\text{v}})$.

Пример искажений С-модели иллюстрирует на рис. 4.4а и 4.4б. Отношение инциденции объекта без искажений (рис. 4.4) представляется двухместным отношением

$$r_{ии} = \{(1, 3), (1, 4), (2, 3), (1, 4), (3, 5), (4, 5)\}.$$

Модель с обрывом связи между элементами 1 и 4 (рис. 4.4а) описывается двухместным отношением

$$r_{ий} = \{(1, 3), (2, 3), (1, 4), (3, 5), (4, 5)\} = r_{ии} \setminus (1, 4).$$

Модель с новой связью между элементами 1 и 2 (рис. 4.4б) описывается двухместным отношением

$$r_{ий} = \{(1, 3), (1, 4), (1, 2), (2, 3), (1, 4), (3, 5), (4, 5)\} = r_{ии} \cup (1, 2).$$

4.5.3. Диагностическая ФС-модель

Она описывается восьмеркой символов:

$$\mathcal{D}_{\text{фс}}=<X, Y, Q, F_{\text{п}}, F_{\text{в}}, R_{u}, R_{ord}, \Phi_{\text{ф}}>.$$

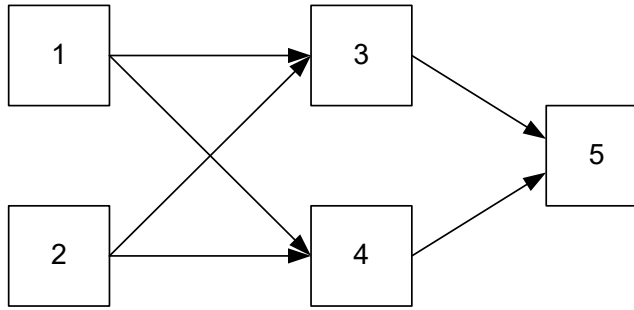

Рис. 4.4. С-модель исправного ОД.

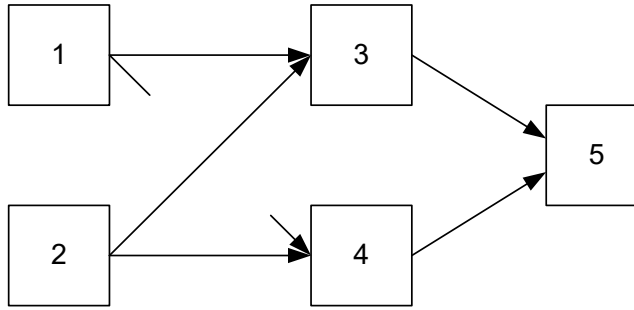

Рис. 4.4а. С-модель ОД с обрывом связи.

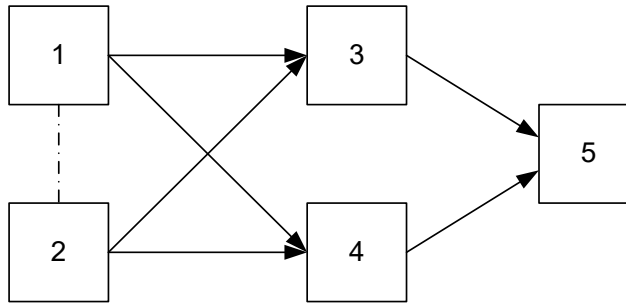

Рис. 4.4б. С-модель ОД с новой связью.

Искажения множества внутренних состояний Q аналогичны искажениям множеств входных и выходных состояний X и Y, рассмотренным выше. То же касается и выходных функций $F_{\text{в}}$.

Искажения специфических для ФС-моделей функций переходов заключается в изменении переходов. Переход $q_{\mu}(t) \rightarrow q_{\nu}(t+1)$ либо выполняется, сохраняя состояние $q_{\mu}(t)$, либо осуществляется в другое состояние $q_{\lambda}(t+1)$. Этим случаем соответствуют модели искажений:

$r_{\mu} = q_{\mu}(t) \rightarrow q_{\nu}(t+1) \,||\, q_{\mu}(t) \rightarrow q_{\mu}(t+1)$,

$r_{\lambda} = q_{\mu}(t) \rightarrow q_{\nu}(t+1) \,||\, q_{\mu}(t) \rightarrow q_{\lambda}(t+1)$, $q_{\mu}, q_{\nu}, q_{\lambda} \in Q$.

В С-модели аналогами этих искажений являются обрыв и возникновение новой связи.

4.5.4. Диагностическая СФ-модель

Она описывается семеркой символов:

$\mathcal{D}_{\text{сф}} = <A_{\text{э}}, R_f, R_{\text{и}}, R_{\in}, R_u, R_{ord}, \Phi_{\text{сф}}>$.

Искажения свойств СФ-модели представляет собой совокупность искажений свойств С- и Ф-моделей. Специфическим для СФ-модели является искажение информации, передаваемой от одного функционального элемента к другому. Оно заключается в замене одной функции транспортировки (передачи) информации $f_{\text{T}i} \in F_{\text{T}}$ другой $f_{\text{T}j} \in F_{\text{T}}$:

$r_{\text{T}j} = f_{\text{T}i} \,||\, f_{\text{T}j}, \quad f_{\text{T}i}, f_{\text{T}j} \in F_{\text{T}}$.

Здесь функция $f_{\text{T}i}$ отображает значения выходных переменных предыдущего элемента в значения входных переменных последующего: $f_{\text{T}i}$: $Y \rightarrow X$, $F_{\text{T}} \subset F$. Искажение пересылки информации описывается двухмерным множеством $r_{\text{T}u} \subset F_{\text{T}} \times F_{\text{T}}$.

Физической причиной возникновения рассмотренного вида искажения в электронных схемах является, например, недопустимое колебание напряжения источника питания.

Логическим электронным схемам наряду с вышеизложенными моделями искажений присуще кратное искажение, включающее различные модели. Причиной его является короткое замыкание соединений между функциональными элементами.

Оно вызывает выравнивание электрических потенциалов в точке замыкания, что равносильно появлению нового логического элемента [34]. Функция последнего определяется в зависимости от способа кодирования логическими значениями электрических потенциалов: И – при кодировании высокого потенциала единицей и ИЛИ – при кодировании его нулём. Появление нового логического элемента вызывает изменение структуры схемы – состава её элементов и связей между ними. Модель искажения для этого случая включает оба вида искажений С-модели:

$$r_A = A_э \setminus A_{э\,и}, \qquad A_{э\,и} = A_э \cup a_i,\ f_i \in F.$$
$$r_{ии} = R_и \setminus R^-_и,$$
$$r_{ии} = R_и \cup R^+_и.$$

Дополнительный элемент a_i реализует функцию И (ИЛИ). Множества $R^-_и(a_i)$ и $R^+_и(a_i)$ представляют соответственно исключенные и внесенные связи, инцидентные новому элементу a_i. Тот факт, что множества $R^-_и(a_i)$ и $R^+_и(a_i)$ не является одноэлементными, доказывается следующим утверждением.

Утверждение 4.2. Короткое замыкание выходов двух элементов электронной схемы вызывает искажение, по крайней мере, *шести* связей между элементами схемы.

Действительно, при замыкании выходов двух логических элементов (или первичных входов схемы) исчезают, по крайней мере, две их связи с элементами-последователями. В свою очередь, появляются, по крайне мере, две новые связи, соединяющие входы последователей с выходом фиктивного логического элемента и две связи, соединяющие входы последнего с замкнутыми выходами логических элементов. Если замкнутые логические элементы имеют разветвления выходов, то количество искажённых связей составляет более шести.

В качестве примера на рис. 4.5. приведена модель короткого замыкания между выходами логических элементов 1 и 2 электронной схемы, имеющей структуру С-модели, изображенной на рис. 4.4.

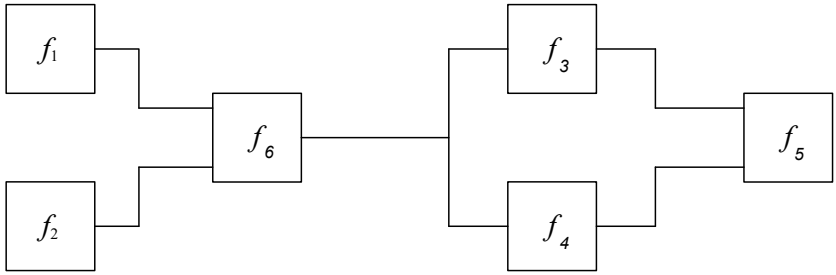

Рис. 4.5. СФ-модель ОД с коротким замыканием.

Отношение инциденции исправной схемы $r_{ии}$ = {(1, 3), (1, 4), (2, 3), (1, 4), (3, 5), (4, 5)} в результате короткого замыкания выходов элементов 1 и 2 трансформируется в следующее отношение инциденции: $r_{иj}$ = {(1, 6), (2, 6), (6, 3), (6, 4), (3, 5), (4, 5)}. Через исключение старых связей и включение новых связей отношение инциденции схемы с коротким замыканием выражается следующим образом:

$r_{иj} = \{r_{ии} \setminus \{(1, 3), (1, 4), (2, 3), (2, 4)\} \cup$

$\cup \{(1, 6), (2, 6), (6, 3), (6, 4)\}\}$.

Как следует из него, короткое замыкание искажает одновременно восемь связей, т.е. более шести в силу наличия разветвлений на выходах логических элементов 1 и 2.

4.5.5. Диагностические модели, отражающие управляющий аспект

Как показано ранее, исходной моделью для отражения аспектов управления может являться любая из Ф-, С-, ФС- и СФ-моделей. К особенностям получаемых на их основе моделей относится разделение их символов на *управляющие* и *управляемые*. В Ф- и ФС-моделях оно реализуется путём разделения алфавитов значений переменных, а в С- и СФ-моделях – путём разделения цепей ЭВМ. Это разделение позволяет учитывать специфику архитектуры управляющей и операционных частей ЭВМ при построении их диагностических моделей. В качестве примера можно сослаться на различие управляющих и управляемых переменных.

Первые из них, как правило, являются двоичными скалярами, а вторые – двоичными векторами. В силу этого один и тот же дефект аппаратуры может повлечь различные неисправности в управляющих и операционных цепях. Однако и те, и другие не выходят за рамки рассмотренных выше моделей искажений.

4.5.6. Сопоставление диагностических моделей ВС

Диагностические модели различных компонентов ВС помимо вышеизложенных основных свойств (функция, структура, управление), отражают некоторые дополнительные свойства, причём возможны различные выборки как тех, так и других свойств. Кроме того, для представления идентичных свойств в различных моделях используются разные языки. Отсюда практически важной задачей являются установление сходства и различий между различными диагностическими моделями устройства и узлов ВС. Она решается с помощью следующего утверждения.

Утверждение 4.3. Две диагностические модели $\mathcal{D}_1$ и $\mathcal{D}_2$ блока ВС эквивалентны, если равны сигнатуры и носители его соответствующих теоретико-множественных моделей.

Доказательство утверждения основывается на понятии изоморфизма алгебраических систем [59]. Содержательно утверждение 4.3 требует отражения моделями $\mathcal{D}_1$ и $\mathcal{D}_2$ одних и тех же свойств системы-оригинала и её искажений. Возможность сопоставления свойств обеспечивается использованием общего теоретико-множественного языка, применяемого для их описания.

Согласно утверждению 4.3 для выяснения эквивалентности двух моделей $\mathcal{D}_1$ и $\mathcal{D}_2$ необходимо построить теоретико-множественные диагностические модели объекта и сопоставить их носители и сигнатуры. Результаты сопоставления нескольких неэквивалентных моделей могут быть сведены в диаграмму, подобную изображенной на рис. 4.2. Более тонкие взаимосвязи между диагностическими моделями устанавливается с помощью теории категорий [67].

Основы систематизации диагностических моделей ВС [95-97] были использованы при разработке отраслевого руководящего материала [134].

Глава 5. СИСТЕМА МЕТОДОВ ТЕХНИЧЕСКОГО ДИАГНОСТИРОВАНИЯ ВС

5.1. Модели методов

Под методом будем понимать совокупность упорядоченных в пространстве и во времени действий, предназначенную для обработки (преобразования, транспортировки, хранения) некоторого объекта любой природы (информации, энергии, вещества).

Рассмотрим существенные признаки, содержащиеся в определении метода. К первым из них отнесём *взаимосвязь* обрабатываемого объекта и совокупности действий, используемых для его обработки. Это отнюдь не означает, что существует взаимно однозначное соответствие между объектом и совокупностью действий по его обработке. Однако особенности объекта безусловно влияют на выбор применяемых для его обработки действий.

Вторым существенным признаком метода является совокупность *действий*, применяемых для обработки объекта. Математическими моделями действий являются *n*-арные функции (операции). Набором последних (включая функцию, реализуемую объектом обработки) прежде всего и характеризуется любой метод. К предметным переменным, значения которых отображаются этими функциями, относятся входные и выходные данные метода. Согласно результатам, изложенным в главе 4, представляемый таким образом метод описывается Ф-моделью.

Третьим существенным (но не обязательным) признаком метода является *распределённость* и *упорядоченность* функций обработки в *пространстве*. Он выражается пространственным отношением инциденции между элементами, реализующими функции. Согласно главе 4 это свойство описывается СФ-моделью. Последняя отражает *статический* аспект метода, поскольку не учитывает временных соотношений между потоками обрабатываемой информации. На этом основании СФ-модель метода отождествима с моделью, реализующей метод системы элементов, обычно графически представляемой блок-схемой.

Четвертым существенным признаком является *распределённость* и *упорядоченность* функций обработки во *времени*. Этот признак отражается ФС-моделью метода.

Рассмотренные существенные признаки метода можно назвать *содержательными*, поскольку они входят в содержание понятия метода.

Поскольку понятие *метод* относится к классу процессуальных понятий, название наиболее общих методов следует непосредственно из первичной классификации понятия-процесса предметной области. Дальнейшая детализация методов в направлении уточнения их свойств есть, по существу, продолжение этой классификации.

Детализация метода начинается с раскрытия его функционального назначения F в функциональный базис $F=\{f_1,\ldots,f_n\}$. Если функциональный базис является *полным,* то на его основе можно формировать *специальные* базисы $F_j \subset F$, содержащие некоторые подмножества функций, удовлетворяющие заданным внешним ограничениям. Очевидно, что два метода, порождённые различными базисами, находятся в отношении толерантности и обладают различными свойствами. Назовём такие методы *базисными.*

Примером двух различающихся базисных методов является следующая пара последовательностей операций: (a, b, a, c), (a, b, a, d). Они толерантны относительно операций a, b и различаются операциями c и d.

Операции, присутствующие во всех специальных базисах (в примере *a, b*), образуют ядро $F_я \subset F$ универсального базиса F. Если операции интерпретировать существенными признаками понятий, то ядру соответствует родовое понятие *ядро методов*, а операции c и d представляют собой видовые отличия производных из ядра – видовых методов. Последние находятся между собой в отношении толерантности.

К другим, менее существенным, признакам различения методов M_i и M_j относятся следующие:

- отношение «общее-частное» между составами используемых операций (доминирование по операциям) $R_c\,(C(M_i), C(M_j))$;
- различный порядок следования операции a $R_a\,(ord\,(M_i), (ord\,(M_j))$;
- различное количество повторений операции a $N_a\,(N_i, N_j)$;
- различное количество операций $N\,(N_i \neq N_j)$;

В качестве примеров приведем следующие пары последовательностей операций для каждого признака:

(a, b, a, b), (a, b, a, d);
(a, b, a, c), (c, b, a, a);
(a, b, a, c), (a, b, b, c);
(a, b, a, c), (a, b).

В отличие от базисных методы, различающиеся относительно перечисленных признаков, назовём *параметрическими,* поскольку они различаются вариацией *значений* одного или нескольких признаков. Очевидно, что каждому базисному методу соответствует группа параметрических методов. Таким образом, параметрический метод характеризуется *длиной* последовательности операций (их общим количеством n), *перечнем* F_μ используемых в ней операций, его *мощностью* n_μ и *порядком* следования операций. В общем случае $F_\mu \subset F$. где F – перечень всех возможных операций, а не только используемых в методе. Множество последовательностей длины n образует n-мерное пространство M^n. Количество составляющих его последовательностей, включающих n_μ операций, равно $n_\mu{}^n$.

Если значения аргументов операций не являются вычисляемыми величинами, а наряду с операциями характеризуют некоторые свойства метода, то они также используются для идентификации метода. В этом случае метод дополнительно характеризуется местностью операций k_μ (обычно k_μ=1, 2) и множеством A_i всевозможных значений аргументов $i = \overline{1, n_\mu}$. Относительно этих переменных метод можно представить в виде n-арного отношения $M_\mu \subset A_1^{k_1} \times ,...,\times A_n^{k_n}$. Множество методов, принадлежащих декартовому произведению $A_1^{k_1} \times ,...,\times A_n^{k_n}$, образует подпространство методов в пространстве M^n. Оно характеризует класс методов, однородных относительно выбранных признаков.

Очевидно, что не каждая последовательность операций позволяет достичь поставленной цели, в силу чего часть последовательностей нельзя именовать методами. Методы, представленные остальными последовательностями, различаются между собой специфическими свойствами и потребляемыми ресурсами. Это различие объясняется различием операций из перечня $F\mu$.

5.2. Методика порождения методов

Приведенные выше модели методов применимы к любому уровню детальности различения методов. Любая функция из функционального базиса i-го уровня общности может быть рекурсивно раскрыта

в функциональный базис меньшего (i+k-го) уровня общности, что характеризует предложенную модель различения методов как многосортную. Будем использовать её для генерации семейств базовых и параметрических методов. Для этой цели применим нижеследующий алгоритм [98].

1. Формулируется назначение метода.
2. Устанавливается i-й уровень общности рассмотрения метода (степень детализации функций).
3. Определяется полный базис F_i.
4. Выделяется ядро базиса $F_{яi} = F_i$.
5. Определяется начальная функция (операция) ядра, с которой начинаются последовательности операций всех методов.
6. Устанавливаются ограничения на порядок следования f_i, $f_j \in F_i$.
7. Строится базовая СФ-модель системы генерации методов i-го уровня.
8. Строится k-й специальный базис F_{ik} путем *добавления* к ядру $F_{Яi}$ дополнительных функций $f \in F_i \setminus F_{я\,i}$, их *удаления* или *замены*.
9. Устанавливаются ограничения на вариацию значений признаков, характеризующих операции базисного метода.
10. Формируется параметрический метод путем установления значений признаков для каждой операции базисного метода.
11. Если не все значения признаков использованы, то меняется очередное значение и переход к 10, иначе к 12.
12. Если не все сочетания функций $f \in F_i \setminus F_{Яi}$ использованы для формирования методов, то переход к 8, иначе к 13.
13. Если требуется большая детализация процедур метода, то устанавливается i+1-й уровень детальности и переход к 3, иначе конец.

Предложенный алгоритм имеет переборный характер и может генерировать весьма большое число методов, часть из которых не представляет интереса [61].

Поскольку операции различаются между собой как свойствами, так и потребляемыми ресурсами, (временными и пространственными), те и другие возможно использовать в качестве классифицирующих признаков. Относительно них всё пространство методов разбивается на классы

эквивалентности. Практический интерес представляет фактор-множество методов. Его мощность равна 2^P, где p – суммарное количество свойств и ограничений на ресурсы, которым должен отвечать метод.

Отсюда следуют два способа порождения метода, отвечающего заданным свойствам и ограничениям на ресурсы:

- факторизация пространства методов M^n на 2^P классов эквивалентности и синтез *одного* из представителей класса;
- генерация *всех* представителей классов эквивалентности с последующей их селекцией относительно заданных свойств и ограничений на ресурсы.

Первый способ предполагает анализ *каждой* очередной операции на соответствие основному назначению, заданным свойствам и ограничениям на ресурсы для включения её в последовательность. Во втором случае операция анализируется на каждом шаге только на соответствие *назначению* метода. После завершения синтеза методов они селектируются относительно заданных свойств и ограничений на ресурсы.

Недостатком первого способа является отсутствие гарантии в синтезе представителей всех классов эквивалентности, ибо этот способ резонно отнести к задачам локальной оптимизации. Однако из теории оптимизации известно, что локальная оптимизация (на каждом шаге процедуры) не гарантирует получение глобального оптимума. Наличие одного метода в классе не позволяет также оптимизировать выбор методов при расширении перечня критериев.

К недостаткам второго способа следует отнести необходимость генерации *всех* методов, отвечающих основному назначению, с последующей селекцией их по классам.

Оба способа требуют полного перебора операций. Сокращение перебора осуществляется за счет применения синтаксических и семантических правил построения правильных последовательностей в принятой формальной системе. Противоположным подходом является применение синтаксических и семантических правил для отсеивания неудовлетворяющих их операций из входной последовательности. В качестве последней может использоваться псевдослучайная последовательность. Промежуточным вариантом является конструирование

операций не по всем правилам с последующим отсевом операций, не удовлетворяющих *всем* правилам.

5.3. Модель системы технического диагностирования

Назначением любого метода технического диагностирования является распознавание технического состояния объекта диагностирования. Оно описывается одноаспектной функциональной моделью $\mathcal{M}_{\text{стд}}=<X_{\text{в}}, Y_{\text{с}}, f_{\text{д}}>$. Функция $f_{\text{д}}$ отображает значения переменных из множества $X_{\text{в}}$ в множество $Y_{\text{с}}$, $f_{\text{д}}$: $X_{\text{в}} \rightarrow Y_{\text{и}}$. Эти множества интерпретируются из символов X и Y Ф-модели $\mathcal{M}_{\text{ф}} =<X, Y, F>$ функциями μ_x и μ_y:

μ_x: $X \rightarrow X_{\text{в}}$, где $X_{\text{в}}$ – множество входных воздействий на систему;

μ_y: $Y \rightarrow Y_{\text{с}}$, где $Y_{\text{с}}$ – множество оцененных технических состояний (ОТС), включая правильное.

Поскольку система технического диагностирования (СТД) на самом общем уровне реализует функцию обработки диагностической информации, её функциональный базис F формируется на основе функционального базиса обработки информации. Последний, согласно изложенному ранее, включает функции: *преобразования T*, *хранения S*, *передачи L*, *коммутации K*, *адресации* (выбора) *D*, а также функцию *оценки E* информации [8].

Обработке и оценке в СТД подлежит следующая диагностическая информация [99]:

- диагностическая модель (ДМ) объекта диагностирования;
- тестовые воздействия (ТВ) при тестовом диагностировании;
- рабочие воздействия (РВ) при рабочем диагностировании;
- базы сравнения (БС);
- выходные реакции (ВР);
- результаты сравнений (РС).

Выходной информацией СТД является оцененное техническое состояние ОД. В том случае, когда СТД включена в состав системы отказоустойчивых вычислений, выходной информацией последней являются восстановленные выходные реакции (ВВР).

Ядро функционального базиса образуют следующие функции диагностирования:

- генерация входных воздействий на ОД;
- генерация баз сравнения;
- компарация (сопоставление) реакций ОД на воздействия с эталонной базой сравнения при контроле;
- дешифрация состояния ОД при диагностировании.

Они интерпретируют общие функции обработки информации, перечисленные выше. Применительно к задаче диагностирования последние образуют следующие группы.

Функции преобразования:

- преобразование диагностической модели ОД в последовательность ТВ (генерация тестовой последовательности воздействий – ГТП) $T_{\text{ГТП}}$: $Z_{\text{ДМ}} \rightarrow X_{\text{ТВ}}$;
- преобразование тестовой последовательности ОД в последовательность БС (генерация баз сравнений – ГБС) $T_{\text{ГБС}}$: $X_{\text{ТВ}} \rightarrow Y_{\text{БС}}$;
- преобразование рабочих воздействий (ПРВ) в базы сравнений $T_{\text{ПРВ}}$: $X_{\text{РВ}} \rightarrow Y_{\text{БС}}$;
- преобразование выходных реакций (ПВР) к норме баз сравнений $T_{\text{ПВР}}$: $Y_{\text{ВР}} \rightarrow Y_{\text{ПВР}}$;
- преобразование баз сравнений (ПБС) в сигнатуры (свёртка выходных последовательностей) $T_{\text{ПБС}}$: $Y_{\text{БС}} \rightarrow Y_{\text{С}}$;

Таким образом, функциональный *базис преобразований* T представляется пятеркой:

$T = \{T_{\text{ГТП}}, T_{\text{ГБС}}, T_{\text{ПРВ}}, T_{\text{ПВР}}, T_{\text{ПБС}}\}$.

Функции хранения (фиксации) информации:

- хранение тестовой последовательности (ФТП) $S_{\text{ФТП}}$: $X_{\text{ТВ}} \rightarrow X_{\text{ТВ}}$;
- хранение баз сравнений (ФБС) $S_{\text{ФБС}}$: $X_{\text{БС}} \rightarrow X_{\text{БС}}$;
- хранение выходных реакция (ФВР) $S_{\text{ФВР}}$: $X_{\text{ВР}} \rightarrow X_{\text{ВР}}$.

Отсюда функциональный *базис хранения* представляется тройкой:

$S = \{S_{\text{ФТП}}, S_{\text{ФБС}}, S_{\text{ФВР}}\}$.

Коммутация информации

Коммутация (переключение) режимов тестового T и рабочего W диагностирования описывается одноместным предикатом K: $M \rightarrow \{0,1\}$, где $M = \{T, W\}$.

Оценивание информации

Сопоставление выходных реакций ОД с базами сравнения описывается при поэлементной оценке двухместным предикатом $C^{(2)}$: $R_{\text{ВР}} \times B \to \{0,1\}$, где $R_{\text{ВР}}$ – множество выходных реакций, B – множество баз сравнений. При интегральной оценке (с применением сигнатуры выходной последовательности) – сопоставление выходных реакций ОД с базами сравнений описывается L+1 – местным предикатом C: $R_{\text{ВР}}{}^{L} \times B_{\text{С}} \to \{0,1\}$, где $B_{\text{С}}$ – множество сигнатур исправного и неисправных объектов.

Дешифрация результатов диагностирования описывается N+1 – местным предикатом $D^{(N+1)}$: $Y_{\text{РС}}{}^{N+1} \to \{0,1\}$, где $Y_{\text{РС}}$ – множество результатов сравнений.

Помимо собственных функций СТД её функциональный базис должен включать функцию обработки информации $f_{\text{об}}$, реализуемую объектом диагностирования, поскольку она обеспечивает замкнутость системы.

Таким образом, полный функциональный базис F СТД имеет

$R_f = \{T_{\text{ГТП}}, T_{\text{ГБС}}, T_{\text{ПРВ}}, T_{\text{ПВР}}, T_{\text{ПБС}}, S_{\text{ФТП}}, S_{\text{ФБС}}, S_{\text{ФВР}}, K, C, D, f_{\text{ОБ}}\}$.

Из входящих в него функций ядро системы контроля технического состояния образуют функции $f_{\text{ОД}}$, $T_{\text{ГБС}}$, $C^{(2)}$, а ядро системы технического диагностирования – функции $f_{\text{ОБ}}$, $T_{\text{ГБС}}$, $D^{(N+1)}$. В ядро системы *тестового* диагностирования помимо них входит $T_{\text{ГТП}}$.

Взаимосвязь функций и реализация их элементами СТД описывается с помощью СФ-модели $\mathcal{M}_{\text{сф}} = \langle A_D, R_f, R_{\text{и}}, R_{\in} \rangle$ с сигнатурой $\Sigma_{\text{стд}} =$ $= \langle R_f, R_{\text{и}}, R_{\in} \rangle$. В ней $R_f = F_i$ представляет собой функциональный базис системы i-й сортности.

Минимальный состав множества элементов $A_{\text{э,min}}$ СФ-модели СТД обуславливается необходимостью реализации выбираемых функций из функционального базиса R_f. Фактический состав $A_{\text{э}}$ превосходит состав $A_{\text{э,min}}$ за счёт использования однотипных функций на *различных* этапах диагностирования и их *повторения* для неисправных модификаций ОД. Принадлежность функций из R_f конкретным элементам $A_{\text{э}}$ устанавливается с помощью отношений $r_{\in} \in R_{\in}$.

Отношение инциденции $\text{R}_{\text{и}}$ на множестве $A_{\text{э}}$ устанавливается на основе

причинно-следственных связей между операциями (функциями) из R_f. Например, невозможно оценить результат вычисления, не имея последнего и базы сравнения. В свою очередь, нельзя вычислить результат, не зная последовательности входных воздействий.

СФ-модель, по существу, описывает схему решения задачи диагностирования, отражая ее пространственный и частично временной аспекты. Последний отражается частично, поскольку схема, упорядочивая следование потоков диагностической информации, не отражает их синхронизацию.

5.4. Блок - схема СТД

Графическая интерпретация СФ-модели СТД, построенная на основе причинно-следственных связей между операциями из R_f, приведена на рис. 5.1. Она представляет собой блок-схему СТД [8], содержательно интерпретируемую следующим образом.

В качестве ОД может рассматриваться как объект в целом (вычислительная сеть, вычислительной модуль и т.д.), так и его части (микропроцессоры, функциональные узлы и т.д.).

Генераторы баз сравнения (ГБС) служат для формирования ожидаемых реакций исправной и неисправных модификаций объекта на входную последовательность воздействий.

Компараторы (КМП) служат для сопоставления выходных реакций и ожидаемых баз сравнений ОД.

Дешифратор (ДШ) используется для оценивания технического состояния (ОТС) объекта по исходам сопоставления выходных реакций (ВР) с ожидаемыми базами сравнений.

Преобразователи служат для преобразования кодов воздействий (ПРВ), реакций (ПВР) и баз сравнения (ПБС) в соответствии с принятым законом кодирования (декодирования).

Фиксаторы представляют собой память, предназначенную для запоминания тестовой последовательности (ФТП), выходных реакций (ФВР) и баз сравнения (ФБС) соответственно. Они используются как с целью постоянного хранения тестовой последовательности, так и для буферизации входных и ожидаемых реакций в процессе диагностирования.

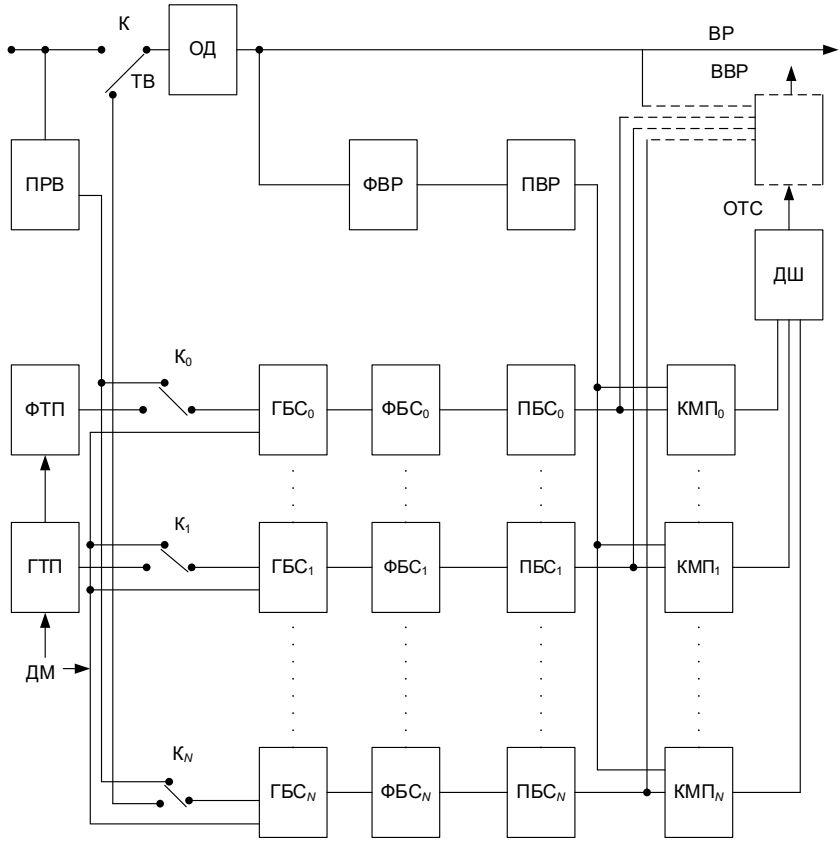

Рис. 5.1. Блок-схема системы технического диагностирования

Генератор тестовой последовательности (ГТП) реализует алгоритм нахождения тестовых воздействий. Исходными данными для ГТП служат модель объекта и перечень предполагаемых неисправностей, объединённые в диагностической модели объекта.

Коммутатор (К) предназначен для изменения режима функционирования объекта с рабочего на тестовый и наоборот.

Восстанавливающий орган (ВО) в системе отказоустойчивых вычислений (СОВ) служит для устранения ошибок в случае их возникновения и формирования восстановленных выходных реакций (ВВР) объекта. Исходной информацией для устранения ошибки является искажённая реакция объекта и найденный путём дешифрации источник ошибки, либо большинство одинаковых реакций на входное воздействие.

Все блоки рассматриваемой блок-схемы носят обобщённый характер. Они могут интерпретироваться различным образом и реализовываться как аппаратными, так и программными средствами.

Приведенная обобщенная блок-схема технического диагностирования практически сохраняет свою конфигурацию как для случая функционального, так и параметрического диагностирования. Различие заключается лишь в реализации блоков системы. Это же касается внешнего и внутреннего диагностирования объекта. Различие в реализации этих процессов определяется интерфейсами средств диагностирования с объектом.

Различие между тестовым и рабочим диагностированием определяется следующими признаками:

- источником генерации воздействий;
- коммутацией ОД с источником воздействий;
- видом обнаруживаемых неисправностей.

Тестовое диагностирование характеризуется использованием генератора тестовых воздействий в качестве источника генерации воздействий, коммутацией последнего со специальными входами ОД, а также нацеленностью на обнаружение устойчивых неисправностей. В противоположность тестовому рабочее диагностирование характеризуется источником рабочей информации в качестве генератора воздействия, коммутацией последнего с рабочими входами ОД и нацеленностью на обнаружение неустойчивых неисправностей (сбоев), порождающих ошибки вычислении (хотя оно обнаруживает также и устойчивые неисправности).

Блок-схемы контроля технического состояния объекта и поиска неисправностей (ошибок вычислений) различаются количеством копий генератора базы сравнения и связанных с ними блоков, а также наличием дешифратора неисправностей (ошибок) в системе поиска.

5.5. Базисные методы технического диагностирования

Рассмотренная блок-схема характеризует универсальную СТД, поскольку она охватывает:

- режимы тестового и рабочего диагностирования ОД;
- стадии проектирования и применения тестов;
- контроль и диагностику технических состояний ОД;
- прямой и сигнатурный анализ диагностической информации;
- буферизацию диагностической информации;
- кодирование и декодирование рабочей информации.

СФ-модель универсальной СТД является полной [67], так как включает полный функциональный базис методов диагностирования. С другой стороны, СФ-модель, построенная на основе ядра функционального базиса, представляет собой наименьший образ [67] модели СТД. Отсюда следует, что СФ-модели, занимающие промежуточное положение между названными моделями, образует решётку, максимальным элементом которой является полная модель СТД, а минимальным – её наименьший образ. Последнему гомоморфны все остальные СФ-модели решётки, которые находятся между собой в отношении толерантности.

Полезно указать на отношение часть-целое между СФ-моделями СТД и СОВ, наглядно показанное на рис. 5.1. Однако это не означает, что первая является наименьшим образом последней. Её назначение уже, о чём свидетельствует отсутствие в её функциональном базисе функций восстановления обрабатываемой информации V_I и самой системы V_s.

На практике получили широкое распространение специализированные СТД, реализующие часть функций из базиса R_f. Поскольку они находятся в отношении толерантности, количество вариантов СТД весьма велико. Поэтому приведем лишь некоторые из них, реализующие наиболее распространенные методы диагностирования.

5.5.1. Базисные методы тестового диагностирования

По времени генерации теста они также делятся на две группы – методы тестирования *предварительно* построенным тестом и тестом, генерируемым в процессе диагностирования объекта.

Тестирование предварительно построенным тестом

Система, реализующая методы тестирования предварительно построенным тестом, делится на две части – подсистему предварительного построения тестов (проверяющего или поиска дефекта) и подсистему тестирования предварительно построенным тестом, выполняющую контроль или диагностику ОД. По этой причине функциональные базисы подсистем не являются полными.

Функциональный базис подсистемы предварительного построения тестов (ППТ) включает следующие функции:

$F_{\text{ППТ}} = \{T_{\text{ГТП}}, T_{\text{ГБС}}, T_{\text{ПБС}}, S_{\text{ФТП}}, S_{\text{ФБС}}\}$,

а его ядро – $F_{\text{я,ППТ}} = \{ T_{\text{ГТП}}, T_{\text{ГБС}}, S_{\text{ФТП}}, S_{\text{ФБС}}\}$.

Базовая СФ-модель системы ППТ, отражающая отношение инциденции между элементами, реализующими функции из функционального базиса $F_{\text{ППТ}}$, имеет следующее аналитическое выражение:

$$F_{\text{ППТ}} = \{(T_{\text{ГТП}}, S_{\text{ФТП}}), (S_{\text{ФТП}}, T_{\text{ГБС0}}), (T_{\text{ГБС0}}, T_{\text{ПБС0}}), (T_{\text{ПБС0}}, T_{\text{ФБС0}}),$$
$$(S_{\text{ФТП}}, T_{\text{ГБС1}}), (T_{\text{ГБС1}}, T_{\text{ПБС1}}), (T_{\text{ПБС1}}, T_{\text{ФБС1}}),$$
$$. \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad . \quad .$$
$$(S_{\text{ФТП}}, T_{\text{ГБС}N}), (T_{\text{ГБС}N}, T_{\text{ПБС}N}), (T_{\text{ПБС}N}, T_{\text{ФБС}N})\}.$$

В нём варьируемыми членами являются ($T_{\text{ГБС1}}$, $T_{\text{ПБС1}}$) и ($T_{\text{ПБС1}}$, $T_{\text{ФБС1}}$). При отсутствии преобразований выходных последовательностей они заменяются членом ($T_{\text{ПБС}i}$, $T_{\text{ФБС}i}$).

Графическая интерпретация СФ-модели системы ППТ приведена в виде блок-схемы на рис. 5.2. Блоки, реализующие функции из ядра функционального базиса, изображены сплошными линиями, а дополнительные (факультативные) функции – пунктирными линиями.

Тестовая последовательность, генерируемая ГТП на основании модели объекта (МО) и моделей дефектов (МД), помещается в ФТП. Затем с помощью ГБС_0 вычисляется последовательность ожидаемых реакций объекта (совокупность баз сравнения) для построенной ранее тестовой

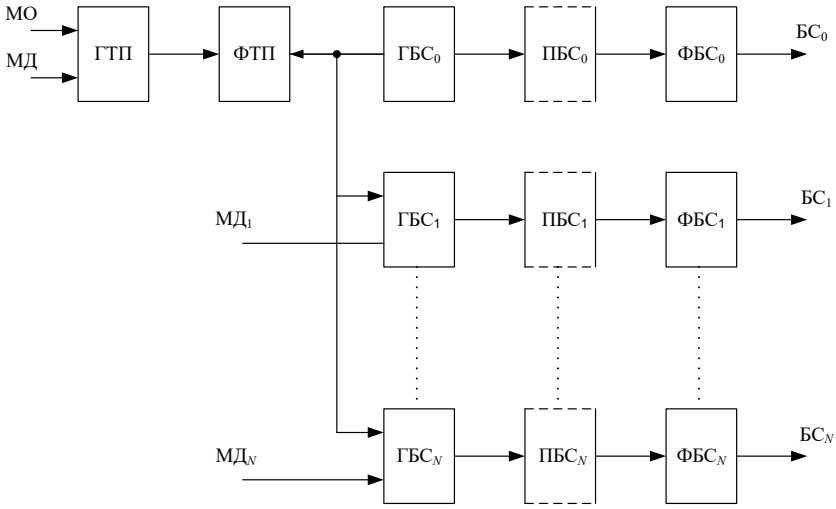

Рис. 5.2. Блок-схема системы предварительного построения теста объекта

последовательности. В качестве $ГБС_0$ используется система моделирования поведения исправного объекта. Для сжатия ожидаемой последовательности может вычисляться её сигнатура преобразователем базы сравнения $ПБС_0$. Непреобразованные или преобразованные базы сравнения фиксируются в $ФБС_0$ для последующего использования. Аналогичным образом вычисляются ожидаемые последовательности для предполагаемых неисправных модификаций объекта от 1 до N. Эти вычисления выполняются с помощью N цепочек блоков, изображенных на рис. 5.2, и реализуемых в системе моделирования неисправностей.

Система предварительного построения теста реализуется вручную или на инструментальной ЭВМ.

Поскольку отображение аналитической СФ-модели в её графическую форму (блок-схему) взаимно однозначно и последняя является более наглядной, чем первая, для остальных базисных методов технического диагностирования будем приводить только блок-схемы реализующих их систем.

Функциональный базис системы тестирования предварительно построенным тестом (ТПТ) включает следующие функции:

$F_{тпт} = \{S_{ФТП}, f_{ОД}, S_{ФВР}, T_{ПВР}, S_{ФБС}, T_{ПБС}, C, D\}$.

Из них функции $F_{ЯК,тпт} = \{S_{ФТП}, f_{ОД}, S_{ФБС}, C\}$ образуют ядро функционального базиса подсистемы контроля, а функции $F_{ЯД,тпт} = \{S_{ФТП}, f_{ОД}, S_{ФБС}, D\}$ образуют ядро функционального базиса подсистемы поиска неисправностей.

Блок-схема системы контроля предварительно построенным тестом представлена на рис. 5.3. Её блоки ФТП и $ФБС_о$ реализуются в виде памяти тестовых воздействий и ожидаемых реакций. Блок ФВР служит для запоминания последовательности выходных реакций объекта, либо выходной сигнатуры. Для получения последней при тестировании используется блок ПВР. По результату сопоставления выходных и ожидаемых реакций объекта делается вывод о его исправности (работоспособности, правильности функций – в зависимости от заданных классов технических состояний).

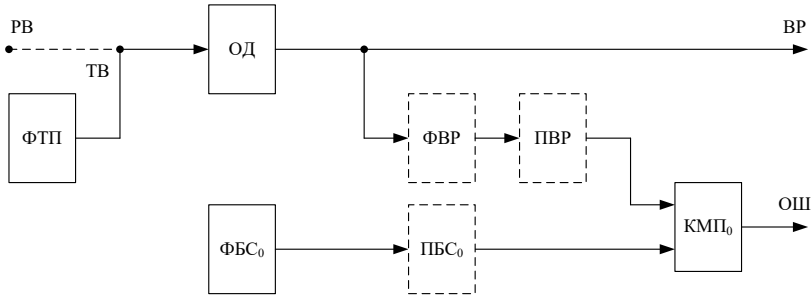

Рис. 5.3. Блок-схема системы контроля предварительного построения тестом.

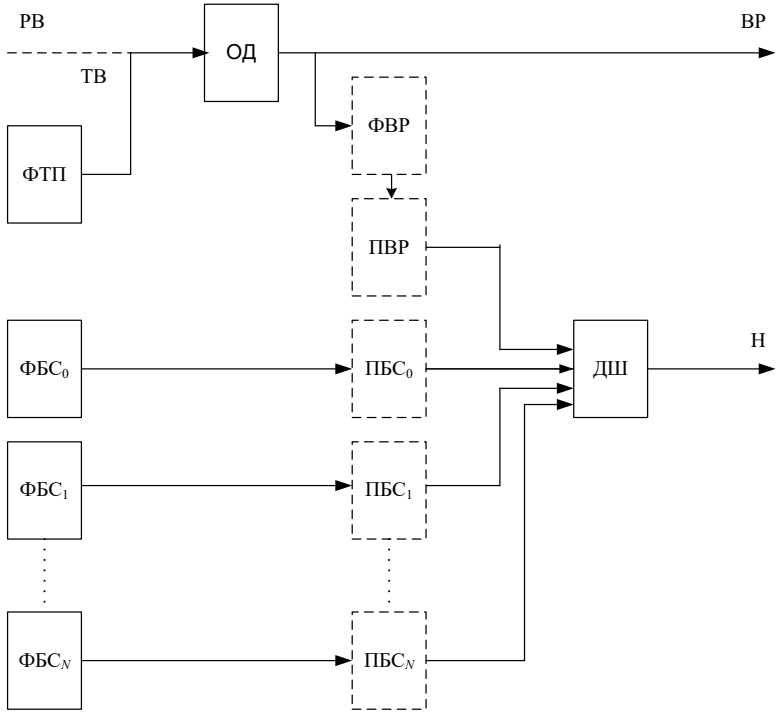

Рис. 5.4. Блок-схема системы поиска неисправностей с помощью предварительного построения теста

Система поиска неисправностей (рис. 5.4) отличается от системы тестового контроля наличием информации о неисправностях, хранящейся в памяти $ФБС_1$–$ФБС_N$. Она сопоставляется с выходной последовательностью или сигнатурой, хранящейся в блоке ФВР. В результате дешифрации выходных реакций и баз сравнений блоком ДШ делается вывод о состоянии объекта. Различные методы дешифрации словарей неисправностей изложены в [4].

В зависимости от моделей, применяемых для построения теста структурно-функциональной или модели функционирования [93], можно выделить методы структурного и функционального тестирования. Разновидностью последнего является контрольная задача. Он составляется на основе типового алгоритма функционирования объекта и использует типовые рабочие данные.

Тестирование генерируемым тестом

Функциональный базис подсистем тестирования генерируемым тестом (ТГТ) включает следующие функции:

$F_{тгт} = \{T_{ГТП}, f_{ОД}, T_{ГБС}, S_{ФВР}, S_{ФБС}, T_{ПВР}, C, D\}$.

Из них функции $F_{ЯК\ тгт}=\{T_{гтп}, f_{ОД}, T_{ГБС}, C\}$ образуют ядро функционального базиса подсистемы контроля, а функции $F_{ЯД\ тгт}=\{T_{ГТП}, f_{ОД}, T_{ГБС}, D\}$ образуют ядро функционального базиса подсистемы поиска неисправностей.

Генерация тестовых воздействий и ожидаемых реакций выполняется в процессе тестирования с помощью блоков ГТП и $ГБС_o$ соответственно (рис. 5.5). При сигнатурном тестировании применяются дополнительно преобразователи: для получения выходной сигнатуры – ПВР и ожидаемой сигнатуры – ПБС.

Блок-схема системы поиска неисправностей (рис. 5.6.) отличается от системы контроля наличием дешифратора и N дополнительных копий генератора и преобразователя базы сравнения.

Этот метод тестирования применяется для случая простых алгоритмов генерации тестов.

5.5.2. Базисные методы рабочего диагностирования

К ним относятся методы распознавания технического состояния объекта в каждый текущий момент времени, но не исправляющие ошибки вычислений. Поэтому блок-схемы систем, реализующих эти методы

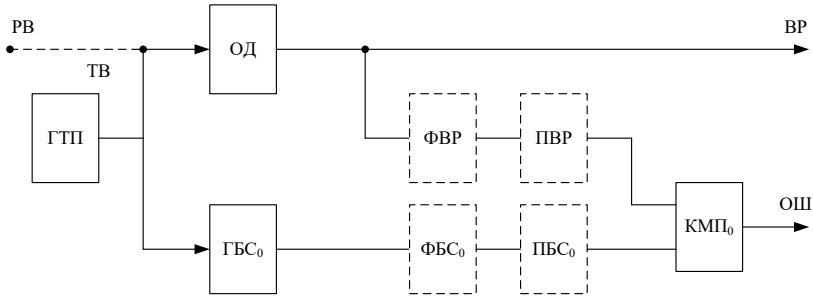

Рис. 5.5. Блок-схема системы контроля с генерацией теста
в процессе тестирования

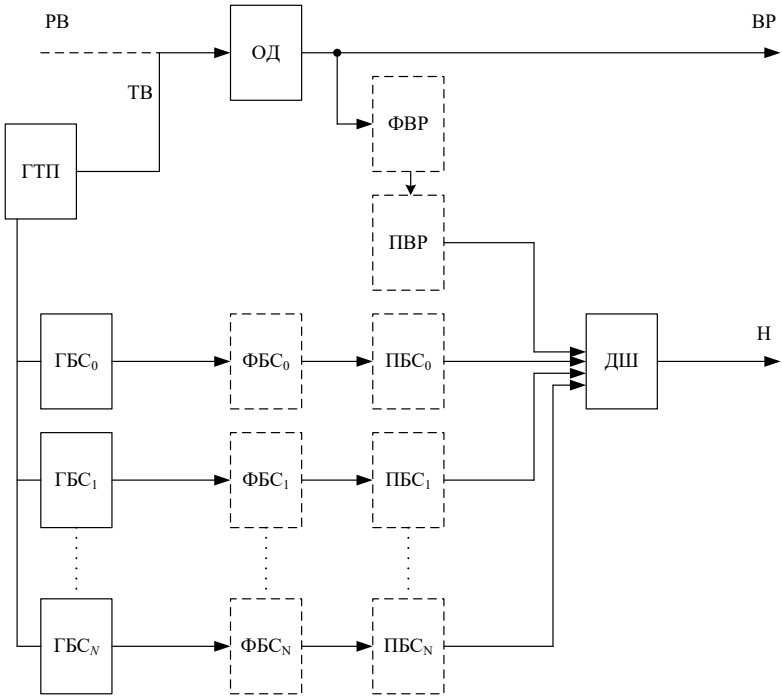

Рис. 5.6. Блок-схема системы поиска неисправностей с генерацией теста
в процессе тестирования

не содержат восстановительных органов [8].

По виду информации, принимаемой за базу сравнения (см. раздел 3.5.), методы делятся на две группы – использующие первичную и преобразованную информацию. В зависимости от вида затрат на обеспечение рабочего диагностирования – аппаратных или временных – первая группа методов делится на методы дублирования и методы повторения вычислений.

Метод дублирования

Функциональный базис системы контроля функционирования дублированием вычислений (КФД) включает следующие функции:

$F_{\text{ттт}} = \{T_{\text{ГБС}}, f_{\text{ОД}}, S_{\text{ФВР}}, S_{\text{ФБС,0}}, T_{\text{ПВР}}, T_{\text{ПБС,0}}, C\}$.

Из них в ядро входят функции $F_{\text{як фд}} = \{T_{\text{ГБС}}, f_{\text{ОД}}, C\}$.

Метод характеризуется параллельным вычислением выходных реакций и баз сравнения. Их сопоставление позволяет реализовать контроль функционирования ОД. Блок-схема системы контроля функционирования объекта методом дублирования представлена на рис. 5.7. На нём генератор базы сравнения реализует один из алгоритмов генерации ожидаемых реакций. В частном случае ГБС_0 представляет собой копию объекта диагностирования. С целью задержки и синхронизации сравнения выходных и ожидаемых реакций в схему включаются фиксаторы ФВР и ФБС_0. При не сравнении на выходе компаратора формируется сигнал ошибки.

В зависимости от вида базы сравнения и, следовательно, от алгоритма её генерации, методы дублирования делятся на *прямой*, *обратный* и *дополняющий*. Названия методов соответствуют соотношению генерируемой ими базы сравнения с выходной реакцией ОД. По отношению к ней в последних двух методах база сравнения является инверсной, либо дополняющей реакцию до некоторого модуля, например, 1. Метод дублирования с вычислением дополняющей базы сравнения обычно называют методом *контрольных соотношений*.

В зависимости от степени соответствия алгоритма генерации базы сравнения алгоритму функционирования ОД различают методы *полного* и *частичного* дублирования. Последние генерируют базу сравнения по некоторым упрощенным алгоритмам, синтезируемым в зависимости от заданного класса предполагаемых ошибок или неисправностей ОД.

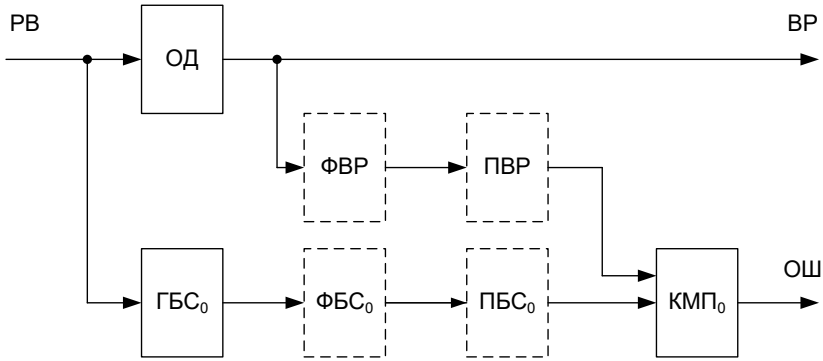

Рис. 5.7. Блок-схема системы контроля функционирования дублированием вычислений

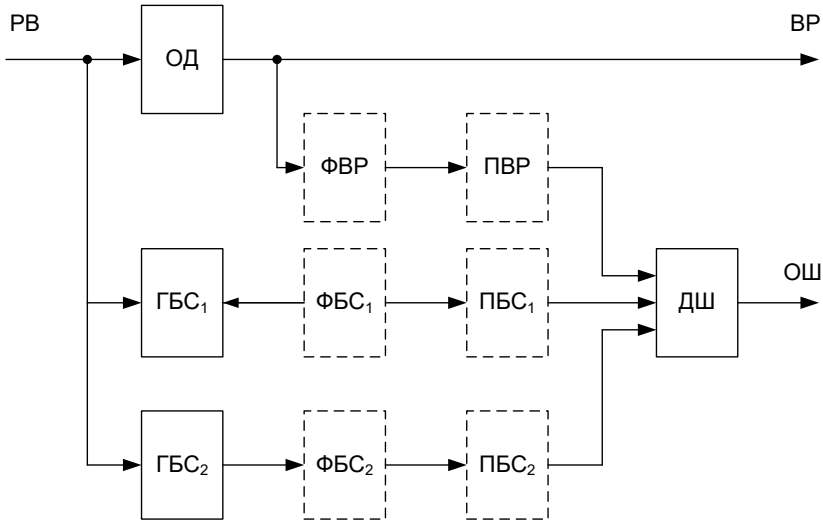

Рис. 5.8. Блок-схема системы диагностирования с применением мажорирования

Метод повторения вычислений

Этот метод предполагает использование в качестве ГБС самого ОД, на входы которого рабочие воздействия подаются повторно. С применением одного повторения вычислений метод позволяет обнаружить сбой функционирования ОД.

Метод мажорирования

Он использует принцип большинства для поиска источника ошибки вычисления. В его функциональный базис входят следующие функции:

$F_{м} = \{T_{ГБС}, f_{ОД}, T_{ПВР}, T_{ПБС}, S_{ФВР}, S_{ФБС}, D\}$

Из них в ядро входят функции $F_{ЯК\ м} = \{T_{ГБС}, f_{ОД}, D\}$.

Блок-схема системы диагностирования с применением мажорирования приведена на рис. 5.10. Она реализует метод *пространственного* мажорирования. Противоположным ему является метод *временного* мажорирования, основанный на методе повторения вычислений нечётное число раз. Смешанным является метод *пространственно-временного* мажорирования, в котором часть результатов вычисляется в копиях ОД, а часть – повторением вычислений в ОД.

Кодовые методы

Они основаны на принципе кодирования входной и выходной информации ОД и используются как для контроля функционирования, так и для поиска ошибок вычислений. Функциональный базис системы диагностирования с применением кодовых методов включает следующие функции:

$F_{кд} = \{T_{ПРВ}, f_{ОД}, T_{ПВР}, T_{ГБС}, S_{ФВР}, S_{ФБС}, T_{ПБС}, C, D\}$.

Из них функции $F_{ЯК\ кд} = \{T_{ПРВ}, f_{ОД}, T_{ПВР}, T_{ГБС}, C\}$ образуют ядро подсистемы контроля, а функции $F_{ЯД\ кд} = \{T_{ПРВ}, f_{ОД}, T_{ПВР}, T_{ГБС}, D\}$ – ядро подсистемы поиска ошибки. Блок-схемы этих подсистем представлены на рисунках 5.8. и 5.9. соответственно. Особенностью этих систем является наличие в их составе преобразователей воздействий и реакций. Система поиска ошибок (рис. 5.9.) в отличие от системы их обнаружения (рис. 5.8.) включает дешифратор, а также в общем случае содержит более одного варианта ГБС и соответствующих преобразователей.

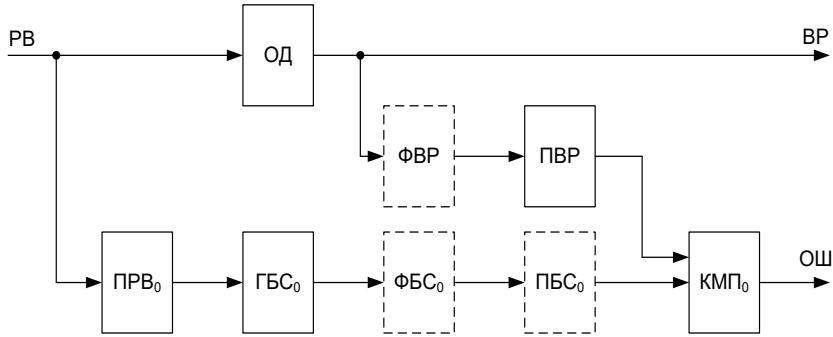

Рис. 5.9. Блок-схема системы контроля функционирования
с использованием кодирования информации

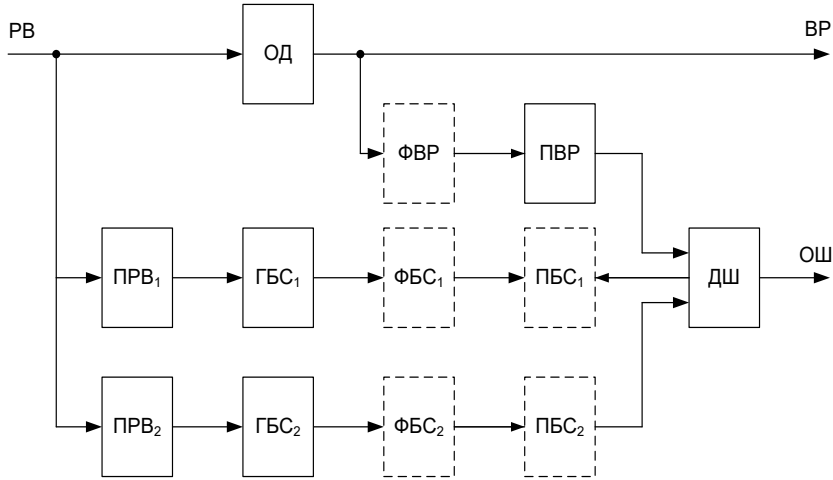

Рис. 5.10. Блок-схема системы диагностирования
с применением кодирования информации

Следует отметить, что система поиска ошибок, основанная на использовании кодовых методов, как правило, не имеет самостоятельного применения, ибо автоматическое исправление найденной ошибки обычно не представляет затруднений. Исходя из этого, при наличии надлежащих ресурсов систему поиска ошибок целесообразно включать в качестве составляющей в систему отказоустойчивых вычислений, содержащую дополнительно к ней блок ВО.

Конфигурация системы поиска ошибок определяется используемым кодом. Реализация любого кода требует включения в систему поиска ошибок преобразователей воздействий, выполняющих функцию кодирования информации, преобразователей реакций и дешифратора, выполняющего функции декодирования обработанной информации. Применение генераторов баз сравнения необходимо лишь для выполнения операций над кодами, контролирующими преобразование информации. К последним относятся, например, коды с двумя вычетами, применяемые для исправления одиночных арифметических ошибок [100]. Количество преобразователей в системе определяется числом проверок, присущих конкретному коду с коррекцией ошибок.

Техническая реализация системы определяется принципом кодирования и декодирования информации и наличием деления обрабатываемой последовательности на блоки (слова).

5.5.3. Смешанные методы технического диагностирования

К ним относятся методы, объединяющие черты тестового и рабочего диагностирования. По этой причине они выделяются в группу методов рабочего тестирования, соответствующую собирательному понятию. Здесь термин «тестирование» используется в качестве краткой формы тестового диагностирования.

Мягкое тестирование

В его функциональный базис входят следующие функции:

$F_{мт} = \{ f_{ОД}, S_{ФТП}, T_{ПБС}, T_{ПВР}, S_{ФВР}, S_{ФБС}, K, C, D\}$.

Из них в ядро входят функции $F_{ЯК\ мт} = \{ f_{ОД}, S_{ФТП}, K, C\}$ при контроле и $F_{ЯД\ мт} = \{f_{ОД}, S_{ФТП}, K, D\}$ – при поиске. Коммутатор используется для переключения режимов – рабочего и тестового.

«Мягким» этот метод называют в силу того, что тестирование объекта выполняется без разрушения находящейся в нем рабочей информации. При этом она используется в качестве тестовых воздействий. Объект на время рабочего тестирования отключается от внешней среды. Метод позволяет обнаруживать устойчивые неисправности.

Временной контроль

Ядром его функционального базиса являются функции:

$F_{вк} = \{f_{ОД}, S_{ГБС}, E, C\}$.

Этот метод использует в качестве диагностического признака *максимальное* время обработки информации. Этот же параметр генерируется и базой сравнения. По источнику генерации базы сравнения, выполняющему предварительное формирование ожидаемого времени обработки данных метод относится к методам тестового диагностирования, а по признаку коммутации ОД и виду обнаруживаемых неисправностей – к методам рабочего диагностирования.

5.6. Базисные методы генерации тестовой последовательности

Согласно разделу 5.1 любая функция из функционального базиса F_i i-го уровня системы порождения методов может быть преобразована в совокупность элементарных функций. В терминах аксиоматической теории, изложенной в 2.9. функция $f_j \in F_i$ с помощью правила подстановки замещается соответствующим ей функциональным базисом. В функциональном базисе системы порождения методов технического диагностирования наибольший интерес с точки зрения технической диагностики представляет функция генерации тестовой последовательности $T_{ГТП}$. Рассмотрим порождаемые на основе соответствующего ей функционального базиса базисные методы генерации (построения) тестовых последовательностей для объектов диагностирования общего вида.

Назначением тестовой последовательности является выявление по реакциям на неё искажений ОД на его наблюдаемых выходах. Искажениям могут подвергаться как элементы ОД и связи между ними, так и непосредственно обрабатываемая им информация.

В задачу генерации тестовой последовательности входит включение в нее входных воздействий (ВВ), обладающих определенным

диагностическим свойством. Под последним понимается способность ВВ изменять функции СД в присутствии какого-либо искажения из заданного перечня. Если последний неизвестен, то в последовательность включаются всевозможные воздействия, выбираемые *последовательно*, либо *псевдослучайным* способом. Такой исчерпывающий перебор воздействий формирует тривиальную или исчерпывающую тестовую последовательность [51]. В том случае, когда перечень возможных искажений известен, основной целью генерации тестовой последовательности становится минимизация числа её членов.

Существует два способа определения члена тестовой последовательности – *выбор* его из имеющейся входной последователь-ности воздействий, либо *синтез* в случае отсутствия или неиспользования таковой. В первом случае используется универсальная диагностическая Ф-модель ОД – таблица функций неисправностей (ТФН), а во втором – одна из специальных моделей ОД и согласованные с ней модели дефектов.

В функциональный базис системы генерации тестовой последовательности (СГТП) $F_{\text{ГТП}}$ входят следующие функции:

- выбор наблюдаемой точки (выхода) $f_{\text{нт}}$;
- упорядочение неисправностей относительно наблюдаемого выхода (с использованием отношения доминирования) $f_{\text{ун}}$;
- объединение эквивалентных по проявлению неисправностей в классы эквивалентности $f_{\text{экв}}$;
- определение установочной последовательности для приведения ОД в исходное для диагностического эксперимента состояние $f_{\text{уст}}$;
- определение диагностической последовательности, обеспечивающей проявление неисправностей на наблюдаемом выходе $f_{\text{д}}$.

Таким образом, $F_{\text{ГТП}} = \{f_{\text{нт}}, f_{\text{ун}}, f_{\text{экв}}, f_{\text{уст}}, f_{\text{д}}\}$.

Ядро функционального базиса составляют функции $f_{\text{нт}}$, $f_{\text{уст}}$, $f_{\text{д}}$. Наименьший образ СФ-модели СГТП, построенный на основе этого ядра, отражает последовательность их перечисления, начиная с начальной функции $f_{\text{нт}}$. Функции $f_{\text{ун}}$, $f_{\text{экв}}$ являются факультативными. Они, расширяя ядро СГТП, могут меняться местами и применяться также после функции $f_{\text{д}}$. Функции $f_{\text{нд}}$, $f_{\text{уст}}$ в частном случае вырождаются в функцию определения члена тестовой последовательности $f_{\text{тп}}$, что присуще

автоматам без памяти – логическим комбинационным устройствам.

Методы генерации тестовой последовательности помимо состава функционального базиса различаются значениями признаков, характеризующих способы реализации функций. Как было отмечено выше, в зависимости от выбора или синтеза тестового воздействия (ТВ) методы делятся на две различные группы.

Условием реализации методов первой группы является возможность включения в тестовую последовательность исходных входных воздействий или их кластеров в *произвольном* порядке. Порядок включения входных воздействий в тестовую последовательность определяется её назначением и принятым критерием. При построении *проверяющей* тестовой последовательности используются стратегии комбинационного поиска [54], либо поиска исправной модификации ОД с максимальным приращением информации о ней на каждом шаге процедуры [54]. При построении *различающей* (диагностической) тестовой последовательности используются стратегии максимальным приращением информации о *всех* модификациях ОД на каждом шаге процедуры (максимизация в среднем) и с максимальным приращением информации о каждой последовательно выделяемой модификации ОД (целевая максимизация). Различие между этими стратегиями необходимо оценивать количественно, что и будет выполнено далее. А пока уместно заметить, что последняя из перечисленных стратегий требует обязательного предварительного выполнения функций $f_{\text{ун}}$ и $f_{\text{экв}}$.

Методы второй группы определяются спецификой ОД. Для объектов непрерывного типа модели дефектов выражаются интегральными функциями. При использовании Ф-модели ОД тестовая последовательность воздействий формируется на основе теории чувствительности [101, 102]. При использовании СФ-модели ОД модели дефектов выражаются относительно промежуточных переменных модели ОД. Для объектов дискретного типа модели дефектов вырождаются в булевы константы – нуль и единицу. В зависимости от используемой модели ОД методы делятся на *функциональные* (Ф-модель) и *структурные* (СФ-модель). Последние получили широкое применение для построения тестов дискретных ВС. Основное различие между ними заключается в способах *активизации* путей распространения неисправностей к наблюдаемому выходу СФ-модели. По этому критерию различают следующие методы:

1. а) с заранее построенным путём распространения неисправности;
   б) с нахождением пути в процессе синтеза ТП;
2. а) с предварительной группировкой неисправностей относительно активируемого пути и последующей коррекцией;
   б) с последующей после активизации пути группировкой неисправностей:
3. а) с началом активизации пути от наблюдаемого выхода ОД;
   б) с началом активизации пути от неисправности ОД;
4. а) со сплошной деактивизавацией неактивируемых путей;
   б) с выборочной деактивизациай неактивируемых путей;
5. а) с активизацией одномерных путей;
   б) с активизацией многомерных путей;
6. а) с активизацией пути для одного значения функции ОД;
   б) с активизацией пути дня обеих противоположных значений функции ОД (синтез парных наборов);
7. а) без условного разрыва замкнутого пути (обратной связи);
   б) с условным разрывом замкнутого пути (обратной связи);
   Перечисленный перечень методов является открытым.

Выбор метода активизации влияет на свойства синтезируемого теста. Например, применительно к комбинационным схемам количество одновременно активируемых путей при построении тестового воздействия определяет назначение теста. Максимальное количество активируемых путей, а, следовательно, максимальное приращение неисправностей на каждом наборе, характеризует тест контроля, а минимальное количество – тест поиска дефектов. Выбор стратегии деактивизации путей в комбинационных схемах влияет на полноту теста по отношению к кратным неисправностям [103].

Вторым критерием, определяющим различие методов, является совокупность свойств, отражаемых диагностической моделью (см. гл.4), а третьим – используемая математическая форма представления диагностической модели: аналитическая, графовая иди табличная (матричная). Примерами методов, использующих перечисленные формы, для логических схем являются методы: алгебраические – ЭНФ [25], графовые – цепей и сечений [28], матричные – D-кубов [31].

Глава 6. КОНСТРУКТИВНЫЕ ДИАГНОСТИЧЕСКИЕ МОДЕЛИ УЗЛОВ ВС И МЕТОДЫ ПОСТРОЕНИЯ ТЕСТОВ

Рассмотрим конкретные интерпретации общих диагностических моделей ВС и методов построения тестовых последовательностей, предназначенные для построения тестов основных узлов ВС. Ниже приводятся оригинальные модели и методы, разработанные автором для конкретных практических применений.

6.1. Комбинационные схемы

6.1.1. Диагностическая модель КС

Диагностическая модель, отражающая не только предполагаемые искажения КС, но и гипотетический словарь одиночных константных неисправностей, представляется матрицами контролирующих цепочек [32]. Она относится к классу *табличных* моделей и представляет собой матрицы включения и выключения специального вида [27]. Для отражения структуры комбинационной схемы наряду со значениями входных переменных схемы в матрицы включения $\mathbf{C}_1$ и выключения $\mathbf{C}_0$ входят значения её промежуточных переменных. Это увеличивает размерность исходных матриц пропорционально числу внутренних связей схемы – на число внутренних переменных по горизонтали и на число параллельных путей – по вертикали.

Каждому пути p_{ab} схемы от a-го её входа до выходя b, $a = \overline{1,m}$, ставятся в соответствие две контролирующие цепочки $c_{ab}{}^1$ и $c_{ab}{}^0$, входящие соответственно в матрицы $\mathbf{C}_1$ и $\mathbf{C}_0$. Цепочка $c_{ab}{}^1$ ($c_{ab}{}^0$) матрицы $\mathbf{C}_1$ ($\mathbf{C}_0$) моделирует влияние части неисправностей логических элементов, через которые проходит путь p_{ab}, на значение функции схемы, меняя его при наличии неисправности с 1 на 0 (с 0 на 1). Это влияние обеспечивается за счёт контролирующих компонент $\dot{0}$ и $\dot{1}$, входящих в алфавит цепочки: $\{0, 1, x, \dot{0}, \dot{1}\}$. Они идентичны символам $\overline{D}$ и D в D-кубах неисправностей [31], но различаются относительно неисправностей и путей их распространения.

Контролирующими компонентами трёхвходового логического элемента ИЛИ являются все компоненты проверяющего вектора $(\dot{0}\,\dot{0}\,\dot{0})$ и одна из компонент каждого 1-кратного диагностического вектора: $(\dot{1}\,0\,0)$, $(0\,\dot{1}\,0)$, $(0\,0\,\dot{1})$. Они характеризуют существенные переменные функции

ИЛИ (НЕ ИЛИ). Контролирующие компоненты для элемента И (НЕ И) имеют противоположные значения: $(\dot{1}\dot{1}\dot{1})$, $(0\dot{1}\dot{1})$, $(\dot{1}0\dot{1})$, $(\dot{1}\dot{1}0)$. Контролирующие компоненты $\dot{0}$ и $\dot{1}$ обнаруживают соответственно константные неисправности $x_i||1$ и $x_i||0$ на i-м входе логического элемента.

Исходные контролирующие цепочки в матрицах $\mathbf{C}_1$ и $\mathbf{C}_0$ состоят из контролирующих компонент $\dot{0}$ и $\dot{1}$ и неопределенных значений x. Правила построения цепочек основываются на существенности переменных и монотонности функций логических элементов схемы. Исходя из этих свойств контролирующие компоненты на i-м входе элемента и его выходе имеют одинаковые значения для функций $\vee$, $\wedge$ и противоположные значения – для функций $\overline{\vee}$, $\overline{\wedge}$, $\neg$. Это позволяет формальным путем определять значения контролирующих компонент цепочки c^1_{ab} (c^0_{ab}) для пути p_{ab} относительно значения контролирующей компоненты на выходе схемы (последнего элемента b).

В качестве примера построим контролирующие цепочки для пути p_{135}, проходящего через элементы 1, 3, 5 в логической схеме, С-модель которой изображена на рис. 4.4. Пусть элементы 1, 2, 3, 4, 5 реализуют соответственно функции x_1, x_2, $y_3 = x^1_1 \overline{\vee} x^1_2$, , $y_4 = x^2_1 \wedge x^2_2$, $y_5 = y_5 \vee y_4$ с учётом разветвлений переменных x_1 на x^1_1 и x^1_2 и x_2, на x^2_1 и x^2_2. Тогда относительно полного вектора переменных $(x^1_1, x^2_1, x^1_2, x^2_2, y_3, y_4, y_5)$ цепочки $c^1{}_{135}$ и $c^0{}_{135}$ имеют вид:

$$c^1{}_{135}=(\dot{0}x\,x\,\dot{1}\,x\,\dot{1}),\ c^0{}_{135}=(\dot{1}x\,x\,x\,\dot{0}\,x\,\dot{0}).$$

Поскольку для функционального базиса ($\vee$, $\wedge$, $\overline{\vee}$, $\overline{\wedge}$, $\neg$) цепочки взаимно обратны относительно контролирующих компонент, одна из них может получаться из другой с помощью инвертирования этих компонент. Ниже для рассматриваемой схемы приведены матрицы путей $\mathbf{P}$ и контролирующих цепочек $\mathbf{M}_1$ и $\mathbf{M}_0$:

$$\mathbf{P} = \begin{array}{|cccc|ccc|c|} & & & & & & \overline{\vee} & \wedge & \vee \\ x^1_1 & x^2_1 & x^1_2 & x^2_2 & y_3 & y_4 & y_5 \\ \hline 1 & & & & 1 & & 1 \\ & 1 & & & & 1 & 1 \\ & & 1 & & 1 & & 1 \\ & & & 1 & & 1 & 1 \\ \hline \end{array}$$

$$\mathbf{M}_0 = \begin{array}{|cccc|c|c|} \multicolumn{4}{c|}{} & \overline{\vee} & \wedge & \vee \\ x_1^1 & x_1^2 & x_2^1 & x_2^2 & y_3 & y_4 & y_5 \\ \hline \dot{1} & & & & \dot{0} & & \dot{0} \\ & \dot{0} & & & & \dot{0} & \dot{0} \\ & & \dot{1} & & \dot{0} & & \dot{0} \\ & & & \dot{0} & & \dot{0} & \dot{0} \\ \hline \end{array}$$

$$\mathbf{M}_1 = \begin{array}{|cccc|c|c|} \multicolumn{4}{c|}{} & \overline{\vee} & \wedge & \vee \\ x_1^1 & x_1^2 & x_2^1 & x_2^2 & y_3 & y_4 & y_5 \\ \hline \dot{0} & & & & \dot{1} & & \dot{1} \\ & \dot{1} & & & & \dot{1} & \dot{1} \\ & & \dot{0} & & \dot{1} & & \dot{1} \\ & & & \dot{1} & & \dot{1} & \dot{1} \\ \hline \end{array}$$

Матрицы $\mathbf{M}_0$ и $\mathbf{M}_1$ секционированы относительно разветвлений и ярусов логической схемы. Произвольные значения *x* опущены.

Матрицы контролирующих цепочек отражают структурный и функциональный аспекты общей двух-аспектной диагностической СФ-модели. Им также присущи свойства явного отражения неисправностей схемы и приспособленности модели к синтезу теста. Последнее свойство выражается наличием контролирующих цепочек *перед* синтезом теста. Однако для сложных схем более выгодно с целью экономии памяти строить цепочки в ходе синтеза теста [104]. Модель матриц контролирующих цепочек является полной по отношению к словарю неисправностей, который строится одновременно с синтезом тестовых воздействий.

Рассмотренные свойства матриц контролирующих цепочек позволяют конкретизировать отношение толерантности как совокупности общих и различающих признаков между этой моделью и распространенной моделью *D*-кубов Рота [31].

6.1.2. Метод построения теста КС

Как и все структурные методы, он основан на активизации путей распространения неисправностей. Применительно к используемой диагностической модели активизация представляет собой процесс

обеспечения условий истинности признаков контроля булевых компонент цепочки c^1_{ab} (c^0_{ab}).

Признак контроля компоненты $\dot{u}_{\mu\nu} \in c^1_{ab}$ (c^0_{ab}) истинен, если изменение его значения влечет изменение значений компонент подцепочки $c_{\nu b}=(\dot{u}_{\nu\chi}, \ldots, \dot{u}_{\xi b})$. Для обеспечения этого условия на входах ν-й вершины графа схемы строится проверяющий вектор $\mathbf{e}_\text{п}$, если $\dot{u}_\nu = \varphi(\mathbf{e}_\text{п})$ и 1-кратный диагностический вектор $\mathbf{e}_\mu$, если $\dot{u}_\nu = \overline{\varphi}(\mathbf{e}_\text{п})$. В первом случае одновременно с подцепочкой $c_{a\nu}=(\dot{u}_{a\gamma},\ldots,\dot{u}_{\mu\nu})$ активируются ещё $m_\nu - 1$ подцепочек (m_ν-мерная активизация), где m_ν – число входов ν-го элемента. При наличии сходящихся в ν-й вершине разветвлений возможна также её r_ν-мерная активизация ($1<r_\nu\le m_\nu$). При $\dot{u}_\nu = \overline{\varphi}(\mathbf{e}_\text{п})$ она обеспечивается r_ν-кратным диагностическим вектором $\mathbf{e}_{\mu,\lambda,\sigma,\nu}$ где μ, λ, σ – входы элемента ν, принадлежащие сходящимся разветвлениям (рис. 6.1.).

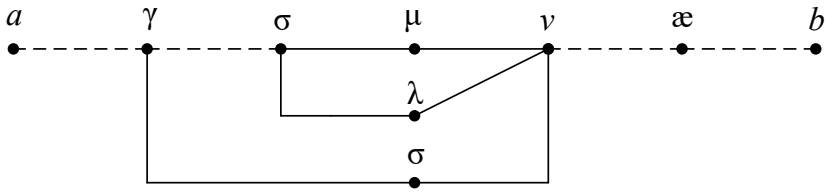

Рис. 6.1. Подграф схемы разветвлениями

Обобщённо активизация и деактивация ν-й вершины – стока параллельных путей по отношению к вершинам – истокам описывается через активизацию и деактивацию сходящихся (параллельных) путей следующими булевыми выражениями:

$$\mathrm{A}_\nu = (\Phi_\nu \wedge \bigvee_{\sigma=1}^{m_\nu} \mathrm{A}\, p_{\sigma\nu}) \vee (\Phi_\nu \wedge (\bigwedge_{\mu=1}^{r_\nu} \mathrm{A}\, p_{\mu\nu} \wedge \bigwedge_{\lambda=1}^{m_\nu - r_\nu} \mathrm{D}\, p_{\lambda\nu}))$$

$$\mathrm{D}_\nu = (\Phi_\nu \wedge \bigvee_{\sigma=1}^{m_\nu} \mathrm{D}\, p_{\sigma\nu}) \vee (\Phi_\nu \wedge (\bigvee_{\mu=1}^{r_\nu} \mathrm{D}\, p_{\mu\nu} \vee \bigwedge_{\lambda=1}^{m_\nu - r_\nu} \mathrm{A}\, p_{\lambda\nu}))$$

Здесь $\Phi_\nu = \begin{cases} 1, \text{ если } \dot{u}_\nu = \varphi(\mathbf{e}_\text{п}); \\ 0, \text{ если } \dot{u}_\nu = \overline{\varphi}(\mathbf{e}_\text{п}). \end{cases}$

В том случае, когда число параллельных путей, входящих в ν-ю вершину, меньше m_ν, вместо отсутствующих дизъюнктивных членов, соответствующих непараллельным путям, подставляются нули, а вместо конъюнктивных – единицы. Размерность активизации вершины ν равна числу входящих в неё активированных путей. Размерность активизации исходного пути p_{ab} определяется вершиной $\nu \in p_{ab}$ с максимальной размерностью активизации.

Необходимо различать многомерную активизацию от деактивизации. Пусть, например, на входах дизъюнктора построен вектор $\mathbf{e}_{1,2}=(1100)$. $A_\nu=1$ ($D_\nu=0$), если по крайней мере два первых входа дизъюнктора являются конечными дугами параллельных путей, причём последние активированы. Если в вершину входят другие параллельные пути, то они должны быть деактивированы. Если эти условия не выполняются, то $A_\nu=0$ ($D_\nu=1$).

Активизация одних и дезактивация других вершин расширяет начальное булево покрытие схемы, определяемое цепочкой c^1_{ab} (c^0_{ab}). Конечным результатом этого расширения является нахождение значений входных переменных схемы, обеспечивающих активизацию этой цепочки.

Если в схеме без разветвлений последовательное расширение булева покрытия является бесконфликтным, то в схемах с разветвлениями при расширении булева покрытия могут иметь место логические противоречия, когда дуге (μ,ν) графа схемы сопоставляются противоположные булевы значения. При этом не для всех компонент исходной контролирующей цепочки c_{ab} признак контроля оказывается истинным. Это означает, что соответствующие им неисправности не обнаруживаются при попытке активизации пути p_{ab}. Иными словами, активизация их оказывается невозможной.

Рассмотрим условия возникновения противоречий при расширении булева покрытия схемы на базе цепочки c_{ab}. Пусть вершина является стоком параллельных путей $p_{\sigma,\mu,\nu}$ и $p_{\sigma,\lambda,\nu}$ (рис. 6.1.). Количество логических элементов с инверсией, принадлежащим этим путям, обозначим

соответственно $N_{\sigma,\mu,\nu}$ и $N_{\sigma,\lambda,\nu}$. Для схем, синтезированных в базисе $\vee, \wedge, \overline{\vee}, \overline{\wedge}$, справедливо следующее утверждение.

Предложение 6.1. Цепочка $c_{a,\mu,\nu,b}$ является противоречивой, если $u_\nu=\varphi(\mathbf{e}_\text{п})$ и при расширении покрытия получается цепочка $c_{\sigma,\lambda,\nu}$, причём $c_{\sigma,\mu,\nu}\neq c_{\sigma,\lambda,\nu}\,(\text{mod } 2)$.

Это предложение легко проверяется рассмотрением параллельных цепочек, соединяющих вершины a и b на рис. 6.2.

Пользуясь вышеприведёнными формулами, рассмотрим случаи не истинности признаков контроля в цепочке c_{ab}. Для случая многомерной активизации вершины ν справедливо следующее утверждение:

Предложение 6.2. Если при $\Phi_\nu=0$ и $r_\nu<1$ $A_\nu=1$, то компоненты подцепочки $c_{\gamma,\nu}$ не являются контролирующими.

Доказательство. Одиночная неисправность соединения $(\sigma,\nu)\in p_{ab}$ не вызывает многомерную активизацию вершины ν. Следовательно, компоненты цепочки c_{ab}, соответствующие участку пути, заключенному между вершинами истоком и истоком параллельных путей, не являются контролирующими. В случае, если активированы вершины – истоки σ и γ, границей истинности признаков контроля является младшая – вершина исток γ.

Для случая многомерной активизации вершины ν справедливо следующее утверждение:

Предложение 6.3. Если при $\Phi_\nu=0$ и $r_\nu=1$ $\bigwedge\limits_{\lambda=1}^{m_\nu-r_\nu} \mathrm{D}\, p_{\lambda\nu}=0$, то компоненты подцепочки $c_{\gamma,\nu}$ не являются контролирующими.

Доказательство. Согласно условиям предложения наряду с единственным активируемым путём оказывается активированным по крайней мере ещё один путь. Ввиду этого неисправность от вершины – истока σ, распространяясь по параллельным путям, компенсируется ($A_\nu=0$). Если имеется несколько вершин-истоков активированных параллельных путей, то границей истинности признаков контроля является старшая вершина – исток σ, поскольку, согласно формуле, деактивация вершины ν определяется дизъюнкцией параллельных активированных путей. $A_\nu=1$ имеет место только для участка пути $p_{\sigma,\nu}$, так как его неисправности не компенсируются через параллельные пути.

Рассмотрим следующий характерный вариант расположения параллельных путей (рис. 6.3.). Пусть вершина η, принадлежащая

параллельному пути, сама является вершиной – истоком параллельных путей. Последние в общем случае могут иметь различные вершины-истоки $\gamma \prec, \ldots, \prec \sigma$, причем $A_{\sigma,\eta,\nu}=1, \ldots, A_{\gamma,\eta,\nu}=1$.

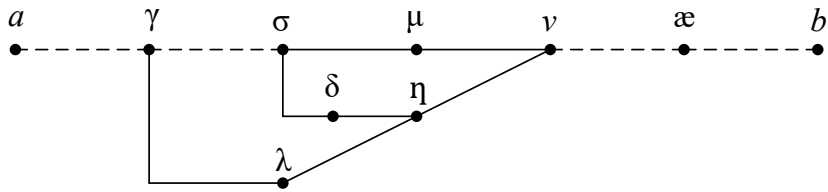

Рис. 6.2. Подграф схемы с вложенными разветвлениями

Для вершины ν при многомерной её активизации имеет место следующее утверждение:

Предложение 6.4. Если при $\Phi_\nu=0$ и $r_\nu>1$ $A_\nu=1$, то при $\Phi_\eta=1$ не являются контролирующими компоненты подцепочки $c_{\sigma\nu}$, а при $\Phi_\eta=0$ – компоненты подцепочки $c_{\gamma\nu}$.

Доказательство. Оно является расширением доказательства предложения 6.2. Пусть вершина η принадлежит активизируемому параллельному пути. Если при $u_\eta=\varphi(\mathbf{e}_\pi)$ ($\Phi_\eta=1$) имеет место r_η-мерная активизация вершины η ($r_\eta>1$), причём активированные параллельные пути имеют различные вершины-истоки $\gamma \prec, \ldots, \prec \sigma$, то для многомерной активизации вершины η оказывается достаточной активизации старшей – вершины истока σ. При $u_\eta=\overline{\varphi}(\mathbf{e}_\pi)$ ($\Phi_\eta=0$) и $r_\eta>1$ активизация вершины η имеет место только при одновременной активизации r_η параллельных путей. Активизация старшей вершины-истока параллельных путей σ зависит от активизации младшей вершины – истока γ. Следовательно, вершина γ определяет нижнюю границу неистинных признаков контроля.

Для вершины ν при одномерной её активизации имеет место следующее утверждение:

Предложение 6.5. Если при $\Phi_\nu=0$ и $r_\nu=1$ $\bigwedge_{\lambda=1}^{m_\nu-r_\nu} \mathrm{D}\, p_{\lambda\nu}=0$,

то при $\Phi_\nu=1$ не являются контролирующими компоненты подцепочки $c_{a\sigma}$, а при $\Phi_\nu=0$ – компоненты подцепочки $c_{a\gamma}$.

Это предложение доказывается аналогично предыдущему.

Из рассмотренных свойств контролирующих цепочек следует:

Предложение 6.6. Если контролирующая цепочка c_{ab} является непротиворечивой, то в не избыточной схеме она позволяет обнаружить все неисправности, поставленные в соответствие её контролирующим компонентам.

Следствие. Пусть p_{ab} в не избыточной схеме, контролирующая цепочка которого оказалась скорректированной, активируется более одного раза.

С учетом изложенного активизация цепочки c_{ab} осуществляется в следующей последовательности.

1. На её основе выполняется начальное расширение булевого покрытия схемы путём вычисления значений зависимых переменных. Противоречивая цепочка исключается из рассмотрения.

2. Делается попытка одновременной активизации пути p_{ab} в направлении от b к a. Она заключается в построении на входах вершины $\nu\in p_{ab}$ с $u_\nu=\overline{\varphi}(\mathbf{e}_\text{п})$ 1-кратного диагностического вектора, причем последний строится покомпонентно. Это означает, что после задания очередной его неконтролирующей компоненты выполняется расширение булевого покрытия схемы.

Вследствие расширения покрытия может оказаться невозможным построение 1-кратного диагностического вектора. В этом случае выполняется r_ν-мерная активизация вершины ν в направлении от вершины истока.

3. Выполняется анализ результатов активизации пути p_{ab} с использованием предложений 6.2-6.5. При этом формируется список неактивированных соединений $(\mu\nu)\in p_{ab}$, для которых признак контроля g_{ab} корректируется с 1 в 0 (в ручной записи точка над булевым значением заменяется на крестик).

4. Производится активизация и деактивизация путей, сходящихся с p_{ab}. Решение об их активизации или деактивизации применяется в зависимости от значений компонент цепочки ν (анализируемой) и μ (предшествующей ей) и *назначения* теста. Если $u_\nu=\overline{\varphi}(\mathbf{e}_\text{п})$, то независимо от значения признака контроля $g_{\mu\nu}$ и назначения теста пути, сходящиеся с

p_{ab}, деактивируются. Если $u_v=\varphi(\mathbf{e}_п)$ и $g_{ab}=1$, то эти пути *активируются* при построении теста контроля и *деактивируются* при построении теста поиска дефектов. Для активизируемых путей используются соответствующие контролирующие цепочки. Истинность признаков контроля определяется так же, как и для основного пути p_{ab}. Если сходящийся путь неактивируем, он деактивируется. Процесс активизации (деактивизации) сходящихся путей завершается после нахождения полного покрытия схемы.

5. Если после активизации пути p_{ab} в результате коррекции признаков контроля список неактивированных соединений оказался непустым, осуществляется активизация этого пути на основе той же цепочки от входа к выходу. Неактивируемые за оба прохода соединения включаются в список избыточных соединений. Их признаки контроля исключаются из рассмотрения.

Процесс активизации путей завершается после покрытия контролирующими компонентами 0 и 1 всех неизбыточних соединений схемы. После построения теста результирующая матрица **M** для рассматриваемого примера имеет следующий вид:

$$\mathbf{M} = \begin{array}{|cccc|ccc|} & & & & \overline{\vee} & \wedge & \vee \\ x_1^1 & x_1^2 & x_2^1 & x_2^2 & y_3 & y_4 & y_5 \\ \dot{1} & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \dot{0} & \dot{1} & 1 & 0 & 0 & 0 \\ \dot{0} & 0 & \dot{0} & 0 & \dot{1} & 0 & \dot{1} \\ 1 & \dot{1} & 1 & \dot{1} & 0 & \dot{1} & \dot{1} \end{array}$$

По существу матрица **M** представляет собой словарь константных неисправностей схемы, выраженный через контролирующие компоненты. Компонентам $\dot{1}$ и $\dot{0}$ соответствуют постоянные значения переменных 0 и 1, обнаруживаемые на выходе схемы. Тестовая последовательность для данной схемы включает 4 входные воздействия: {(10), (01), (00), (11)}.

Описанный метод построения тестов КС характеризуется следующими признаками классификации методов, приведенной в разделе 5.6: 2а, 3а, 4а (при построении теста поиска дефектов), 5а, 6а.

Изложенные модель и метод были использованы для синтеза тестов

и словарей неисправностей различных логических преобразователей [32, 105-107], в том числе синтезированных в произвольном логическом базисе [108] и многозначном алфавите [109], а также для динамического контроля БИС [110]. Их программная реализация – система построения тестов и словарей неисправностей многовыходных комбинационных схем ПОТЕМС была описана в [104].

6.2. Последовательностные схемы

6.2.1. Диагностическая модель

В качестве таковой рассмотрим тестовый граф переходов, использованный для построения тестов последовательностных схем, имеющих цепи установки в начальное состояние [111]. Таблица переходов и запрещенные состояния схемы считаются известными. Модель строится на основе СФ-модели эквивалентной комбинационной схемы, получаемой путём условного обрыва обратных связей исходной последовательной схемы.

Вершинами тестового графа переходов являются полные состояния, т.е. состояния, включающие значения входных и внутренних переменных. В качестве полных состояний используются входные состояния эквивалентной комбинационной схемы, входящие в её тест. С учётом обратных связей часть полных состояний оказываются неустойчивыми. Признаком неустойчивого состояния является неравенство значений переменных на разных концах оборванной обратной связи.

Тестовый граф переходов строится на основе найденных полных состояний по следующему алгоритму [112].

1. Фиксируется начальное полное тестовое состояние автомата.
2. В списке полных тестовых состояний находится непомеченное (неиспользованное) состояние, включающее то же внутреннее состояние, что и предыдущее.
3. Если таковое имеется, то перейти к 4, иначе к 6.
4. Найденное полное состояние включается в тестовую последовательность и помечается. Фиксируется соответствующее ему входное состояние.
5. Если не все состояния в списке помечены, то перейти к 2, иначе к 10.

6. Осуществляется поиск неустойчивого тестового состояния.
7. Если оно имеется в списке, то перейти к 8, иначе к 9.
8. Неустойчивое состояние помечается и на базе его вычисляется новое полное устойчивое состояние (методом итераций). Перейти к 2.
9. Перечисляются разрешённые входные состояния с целью перевода автомата в новое внутреннее состояние, входящее в непомеченное полное тестовое состояние. Перейти к 2.
10. Конец.

В качестве примера на рис. 6.4 представлен тестовый граф переходов, построенный для RS-триггера, изображённого на рис. 6.3.

Вариант тестовой последовательности для эквивалентной комбинационной схемы со входами RSQ включает 4 тестовых вектора: $(\dot{0}0\dot{1})$, $(1\dot{0}\dot{0})$, $(0\dot{0}\dot{0})$, $(0\dot{1}\dot{1})$. Они и принимаются за полные состояния.

На рис. 6.4 векторы полных состояний определены над базой *RSQ*, а входные векторы – над базой *RS*. Неустойчивые состояния 010 и 101 обведены пунктиром. Переходы из устойчивых состояний помечены входными векторами последовательной схемы.

По сравнению с ФС-моделью обычного графа переходов, тестовый граф переходов дополнительно отражает структурно-пространственный аспект последовательностной схемы, поскольку при его построении используется информация о структуре схемы. Эта модель имеет большую приспособленность к построению тестовой последовательности, чем модель итеративной сети [37], поскольку из этого процесса исключается этап вычисления значений функции по структуре схемы. Он переносится на подготовительную фазу – построения диагностической модели последовательностной схемы.

Диагностическая модель сложных схем с памятью строится с применением модульно-функционального принципа [113]. В соответствии с ним схема декомпозируется на соответствующие подавтоматы. К последним предъявляются требования функциональной законченности, минимального количества обратных связей и связей с другими подавтоматами, небольшого объёма. Наименьшая ступень разбиения, называемая простым подавтоматом, представляет собой либо элемент памяти с относящимися к нему схемами возбуждения, либо выходной преобразователь.

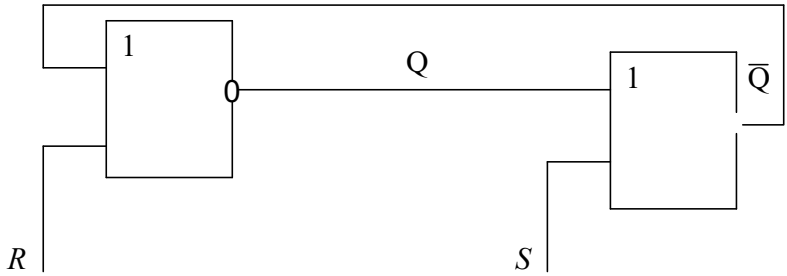

Рис. 6.3. Функциональная схема RS-триггера

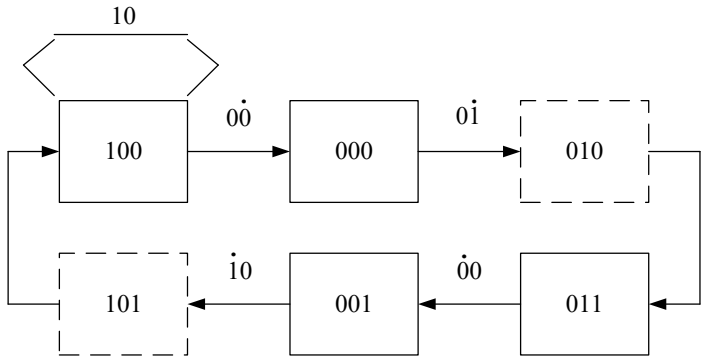

Рис. 6.4. Тестовый граф переходов RS-триггера

6.2.2. Метод построения тестов

Для подавтомата с обратными связями он реализуется путём нахождения эйлерова, либо гамильтонова цикла (в зависимости от характера предполагаемых неисправностей). В примере на рис. 6.4 эти циклы совпадают. Тестовая последовательность имеет вид: (10), ($0\dot{0}$), ($0\dot{1}$), ($\dot{0}0$), ($\dot{1}0$), причём первый вектор является установочным. Изменение значений контролирующих компонент в тестовых векторах влечёт изменение состояний триггера.

Построение теста для подавтомата, включающего триггер со схемами возбуждения, заключается в совмещении их тестов. При этом выполняется следующее условие – тестовый вектор подавтомата должен в присутствии неисправности переводить триггер в противоположное состояние. Для его удовлетворения используются как тестовые, так и не тестовые векторы схем возбуждения триггера.

Тестовая последовательность автомата формируется на базе тестовых последовательностей подавтоматов с помощью следующего алгоритма.

1. Выбирается задающий подавтомат (в качестве задающего в начале по возможности выбирается внешний подавтомат).
2. Фиксируется начальное полное состояние задающего подавтомата.
3. Выполняется пассивная фаза определения полного состояния автомата на основе заданного полного состояния подавтомата, заключающаяся в нахождении значений зависимых переменных других подавтоматов.
4. Если состояние хотя бы одного подавтомата определено не полностью, перейти к 5, иначе – к 6.
5. Выполняется активная фаза доопределения полного состояния автомата. В зависимости от назначения теста в этой фазе делается попытка либо выделить полное состояние одного из подавтоматов (при построении теста поиска дефектов), либо максимально совместить состояния подавтоматов (при построении теста контроля). Активная фаза доопределения чередуется с пассивной до завершения доопределения полного состояния автомата.

6. Если в полное состояние автомата вошли только помеченные состояния подавтоматов, перейти к 8, иначе – к 7.

7. В тестовых последовательностях подавтоматов помечаются состояния, впервые вошедшие в полное состояние автомата. Фиксируется полученное входное состояние автомата.

8. Если все состояния в тестовых последовательностях подавтоматов оказались помеченными, перейти к 14, иначе – к 9.

9. Если выбраны все состояния задающего подавтомата, перейти к 11, иначе – к 10.

10. Выбирается очередное состояние задающего подавтомата, перейти к 3.

11. Если все подавтоматы использовались в качестве задающих, перейти к 13, иначе – к 12.

12. Выбирается следующий подавтомат в качестве задающего, перейти к 2.

13. Если длина тестовой последовательности автомата достигла заданной границы, перейти к 14, иначе – к 11.

14. Конец.

Приведённый алгоритм не гарантирует минимальной длины теста, однако в случае сильно связанного графа переходов тест получается полным. В противном случае для достижения полноты теста требуются промежуточные установки автомата в граничные состояния.

Изложенные модель и метод были использованы для построения теста микросхемы 2ИЕ311, представляющей собой четырёхразрядный счётчик с 17-ю входами, каждый разряд которого содержит по 3 *RS*-триггера и выходной преобразователь [113].

6.3. Однородная структура

6.3.1. Диагностическая модель однородной структуры

Однородные структуры реализуют различные виды обработки информации: преобразование (решающие поля), хранение (ЗУ), передачу (шины). Им присуща регулярность связей между элементами и однородность функций последних.

Рассмотрим однородную структуру, предназначенную для преобразования информации. Каждая ячейка *n*-мерной однородной

структуры имеет m входов ($m \geq n$). В i-м измерении однородной структуры, $i = \overline{1,n}$, расположено s_i функциональных ячеек. По отношению к выходам они разделяются на внутренние и граничные. Внутренняя ячейка имеет n промежуточных и $m–n$ внешних входов. Граничная ячейка имеет менее n промежуточных входов или не имеет их совсем.

Количество входов одномерной структуры равно:

$m_x = (m - 1) \cdot s + 1$.

Если $m = n$, то внешними входами структуры являются только входы её граничных ячеек.

Повторяемость функций и связей однородной структуры позволяет использовать для построения тестов *итеративную* модель. Её диагностическая модель дополнительно включает контролирующие компоненты тестовых векторов на входах элемента структуры.

6.3.2. Метод построения тестов

В основе метода лежит операция обратного отображения функций i-го измерения структуры. Он реализуется с помощью следующего алгоритма.

1. i-му выходу старшей ячейки однородной структуры присваивается значение $\dot{u} = 0$ (1).

2. С учётом значения $\dot{u}$ находится значение u_j для j-го выхода ячейки, $j = \overline{1,n}, \quad j \neq i.$,

3. Выбирается тестовый вектор $\mathbf{e}_t$ ячейки из множества векторов $\{ \mathbf{e}_t \mid \varphi_i^{-1}(u_i) \}$, на которых функция φ_i принимает значение 0 (1).

4. По значению $\dot{u}_i \in \mathbf{e}_t$ выбирается тестовый вектор для входов предыдущей ячейки.

5. Процесс обратного отображения функций ячейки по всем n измерениям структуры продолжается до достижения всех её внешних входов, т.е. до получения тестового вектора структуры.

6. Затем тем же способом строится следующий тестовый вектор структуры. Построение теста завершается тогда, когда построены все тестовые векторы для каждой ячейки структуры.

Построение теста поиска дефектов отличается тем, что на каждом шаге построения теста дополнительно отображаются ещё два различающих вектора – для $\dot{u}_i = 0$ и $\dot{u}_i = 1$. Таким образом, количество векторов на каждом шаге процедуры увеличивается на $l_{i,0} + l_{i,1} + 2$. Здесь первое слагаемое характеризует количество тестовых векторов ячейки, на которых $\dot{u}_i = 0$, а второе – на которых $\dot{u}_i = 1$.

Поскольку выбор тестового вектора на каждом шаге алгоритма зависит от выбранного тестового вектора предыдущей ячейки однородной структуры, удобно изображать процесс построения теста в виде дерева. Каждой граничной ячейке *i*-го измерения структуры ставится в соответствии два тестовых дерева (для выходных значений 0 и 1) . Оба дерева состоят из s_i+1 ярусов. Общее количество тестовых деревьев равно удвоенной сумме количества всех граничных ячеек по каждому измерению однородной структуры.

Для двухмерной однородной структуры более компактной является табличная форма построения тестового вектора. Таблица ассоциируется с направленным графом двухмерной структуры. Для каждой её связи находится компонента тестового вектора соответствующей ячейки. Поскольку для нахождения тестового вектора структуры строится одна таблица, общее их количество равно длине теста однородной структуры.

При определённых ограничениях на функции ячеек однородной структуры процесс построения её теста упрощается. Представляют интерес однородные структуры, длина теста контроля которых не зависит от количества ячеек.

Предложение 6.6. Длина теста контроля *n*-мерной однородной структуры равна длине теста функциональной ячейки, если её функция удовлетворяет следующим условиям:

1. $l_{i,0} + l_{i,1},\ i = \overline{1,n}$;
2. каждая компонента принимает в тестовых векторах ячейки 2^{n-1} нулевых и 2^{n-1} единичных значений;
3. Все компоненты тестовых векторов являются контролирующими.

Перечисленным условиям удовлетворяют линейные булевы функции равнозначности и неравнозначности. Построение теста однородной

структуры, чьи ячейки реализуют эти функции, упрощается за счёт его периодичности. В тесте контроля период следования тестовых векторов ячейки равен длине её теста. Таким образом, найденная путём начального подбора последовательность тестовых векторов для $l_{i,0} + l_{i,1}$ ячеек структуры расширяется на остальную её часть.

В качестве примера на рисунке 6.5. приведены диаграммы и деревья построения теста двухмерной структуры, ячейки которой по горизонтали реализуют функцию неравнозначности, а по вертикали – равнозначности (в каждом измерении по 4 ячейки). Четыре пути двух первых деревьев соответствуют построению теста для *верхней* горизонтали ячеек, а четыре пути третьего и четвертого деревьев – для *правой* вертикали ячеек в каждой из диаграмм. Для построения теста всей структуры требуется 4 диаграммы или $2(s_i + s_j)=16$ тестовых деревьев. Длина теста равна числу тестовых диаграмм, т.е. четырём двоичным наборам. Тестовые векторы формируются на входах каждой диаграммы и представляют собой следующую последовательность: {(0100 1001), (1111 0100), (0010 0010), (1001 1111)}.

Рассмотрим специфику построения теста двухмерной структуры, функции *i*-го измерения которой не удовлетворяют условиям предложения 6.6, а функции *j*-го измерения являются линейными. Пусть $l_{i,1} - l_{i,0} > 0$. Длина теста этой структуры зависит от количества s_i ячеек в *i*-м измерении.

При построении теста контроля этой структуры следует стремиться, прежде всего, получить единичные значения функции φ_i. Вследствие свойства периодичности линейных функций минимальный период следования единиц на *i*-х выходах ячеек 1-го уровня структуры, соответствующий некоторому входному вектору, равен θ_j, а на выходах ячеек s_j-го уровня – $\theta_j^{s_i}$. Таким образом, входной тестовый вектор структуры позволяет получить единицу на *i*-м входе каждой $\theta_j^{s_i-1}$ ячейки s_j-го уровня. Для того, чтобы получить единицу на *i*-м входе каждой ячейки s_j-го уровня понадобится $\theta_j^{s_i-1}$ входных тестовых векторов, а для получения $l_{i,1}$ единиц на *i*-х входах этих ячеек потребуется $l_{i1} \cdot \theta_j^{s_i}$ входных тестовых векторов. Отсюда длина теста рассматриваемой

однородной структуры равна: $L_{\text{п}}=l_{i1}\cdot\theta_j^{s_i-1}+\sigma,$

где σ – дополнительное количество тестовых векторов, требуемых для получения нулей на *i*-х входах ячеек структуры.

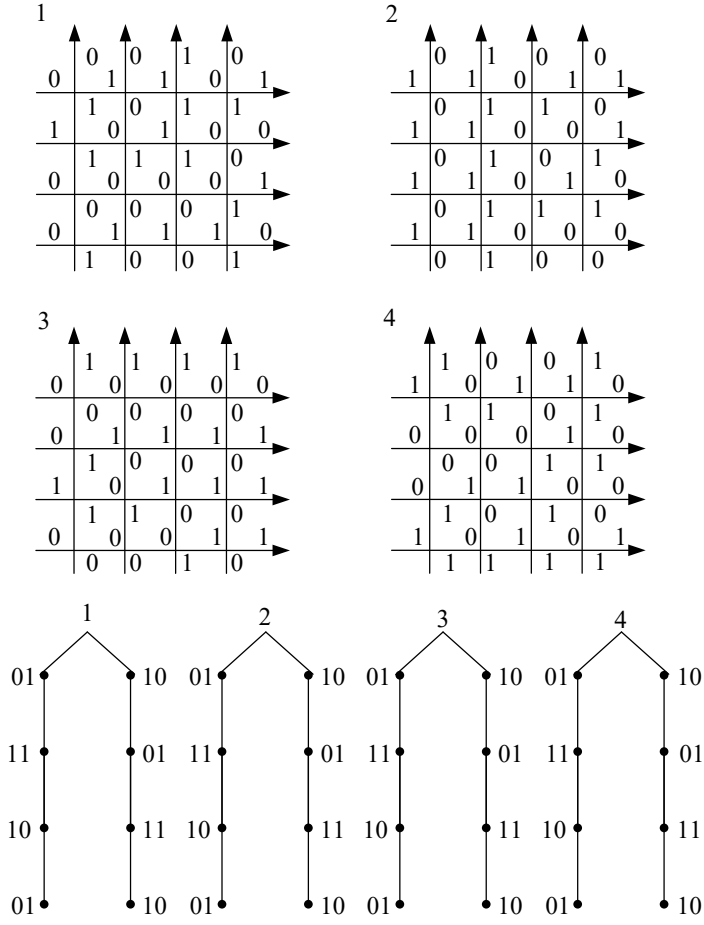

Рис. 6.5. Диаграммы и деревья построения теста двухмерной однородной структуры

Длина теста поиска дефектов однородной структуры относительно внешних выходов j-го измерения равна:

$$L_D=((2^m-2)\cdot s_j+2)\cdot 2^{-1}\cdot\theta_j^{s_i-1}+\sigma.$$

Изложенные модель и метод были применены для построения теста специализированной однородной структуры, выполняющей суммирование данных, поступающих на её входы в каждый такт времени [114]. Она представляет собой массив полусумматоров, реализующих линейную функцию $\varphi_j=x_1\oplus x_2$ в j-м измерении $j=\overline{1,s_j}$ и функцию переноса $\varphi_i=x_1\wedge x_2$ в i-м измерении. Тест ячейки является исчерпывающим (тривиальным) и каждая компонента его тестовых наборов является контролирующей. Согласно выше изложенному, тест этой однородной структуры обладает свойством периодичности. Длины тестов контроля и поиска дефектов структуры равны соответственно:

$$L_{\text{П}}=2\cdot 2^{s_i-1}+2=2^{s_i}+2,$$

$$L_D=(s_i+1)\cdot 2^{s_i}+2.$$

Эти тесты были использованы при испытаниях специализированной однородной структуры и отладки её печатных плат [115, 116].

6.4. Микропрограммно-управляемые устройства

6.4.1. Диагностическая модель микропрограммного – управляемого устройства

Микропрограммно-управляемое устройство характеризуется следующими особенностями:

- управляющие воздействия представляются системой микроинструкций;
- каждая микроинструкция выполняется в один такт машинного времени и активирует выполнение одного или более логико-арифметических выражений;
- результаты вычислений выражений засылаются в выходную (наблюдаемую) или промежуточную регистровую память, либо непосредственно поступают на выходы объекта;
- микроинструкции могут выполняться в различной последовательности;

- обрабатываемые данные могут иметь различную разрядность.

Обобщённая операторная модель микропрограммно-управляемого устройства (МПУУ) представляет собой граф, вершинами которого являются операторы хранения, преобразования и передачи данных, а дугами – связи оператора по данным [41]. Логико-арифметическим выражениям, активируемым микроинструкциями, ставятся в соответствие операторы преобразования и передачи данных. Регистры МПУУ отображаются операторами хранения данных и обозначаются штрихованными вершинами графа. Помимо информационных входов и выходов операторы преобразования и передачи данных имеют предикатные (потенциальные) и управляющие (импульсные) входы. Первые отражают состояние операционного устройства и режим работы управляющего устройства, а вторым ставятся в соответствие микроинструкции, активирующие операторы.

С целью сокращения размерности модели операторные вершины графа объединяются относительно общих последователей и предшественников данных. Входы обобщенного оператора включают все входы составляющих его операторов. Первичным входам и выходам объекта ставятся в соответствие пустые операторы, представляемые терминальными вершинами графа. Для различения хранимых и не хранимых выходных переменных последние обозначаются пунктирными пустыми вершинами. Пути передачи данных в эти вершины также обозначаются пунктирными линиями.

Микроинструкции упорядочиваются по (У,Н)-доступности относительно аргументов и результирующих переменных с использованием таблицы разбиения микроинструкции. Здесь У (управляемость) – минимальное количество микроинструкций, доставляющих данные для оцениваемой микроинструкции, плюс единица (сама микроинструкция), а Н (наблюдаемость) – минимальное число микроинструкций, доставляющих результат операции на внешний выход. Для каждой микроинструкции, принадлежащей одному пути от входной вершины графа до выходной вершины, У+Н=N+2, где N – количество операторных

вершин хранения в пути. Очевидно, что для одной и той же микроинструкции, активирующей оператор, принадлежащий разным путям, с разным количеством операторных вершин хранения *N*, суммы У+Н различаются, а, следовательно, различаются и характеристики (У, Н)-доступности. Таким образом, (У, Н)-доступность является функцией пути. Эта характеристика определяется для *минимальных* путей в графе. Каждая обобщенная операторная вершина, активируемая группой микроинструкций с одинаковыми параметрами У и Н, помечается символом Ф с индексами У, Н.

Упорядочение микроинструкций внутри группы с одинаковыми параметрами У и Н осуществляется по следующим критериям в порядке их возрастания:

- число одновременно активируемых микроинструкцией выражений;
- количество входных переменных в выражении;
- наличие констант в выражении;
- количество операций в выражении;
- соотношение арифметических и логических операций в выражении.

В качестве примера на рисунке 6.7. приведена обобщённая операторная модель двухразрядного центрального процессорного элемента К589 ИК02, структурная схема которого изображена на рис. 6.6. В ней использованы следующие обозначения:

B_0, B_1 – входы внешней шины;

K_0, K_1 – входы маскирующей шины;

X, *Y* – выходы ускоренного переноса;

C_0 – выход переноса;

$СП_0$ – выход сдвига вправо;

$СП_1$ – вход для сдвига вправо;

C_1 – вход переноса;

BA – вход разрешения адреса;

A_1, A_0 – выходы адреса памяти;

(F_6, F_5, F_4), (F_3, F_2, F_1, F_0) – входы кода микрокоманды;

C – вход синхронизации;

D_1, D_0 – выходы информации;

M_1, M_0 – входы информации;

BD – вход разрешения данных.

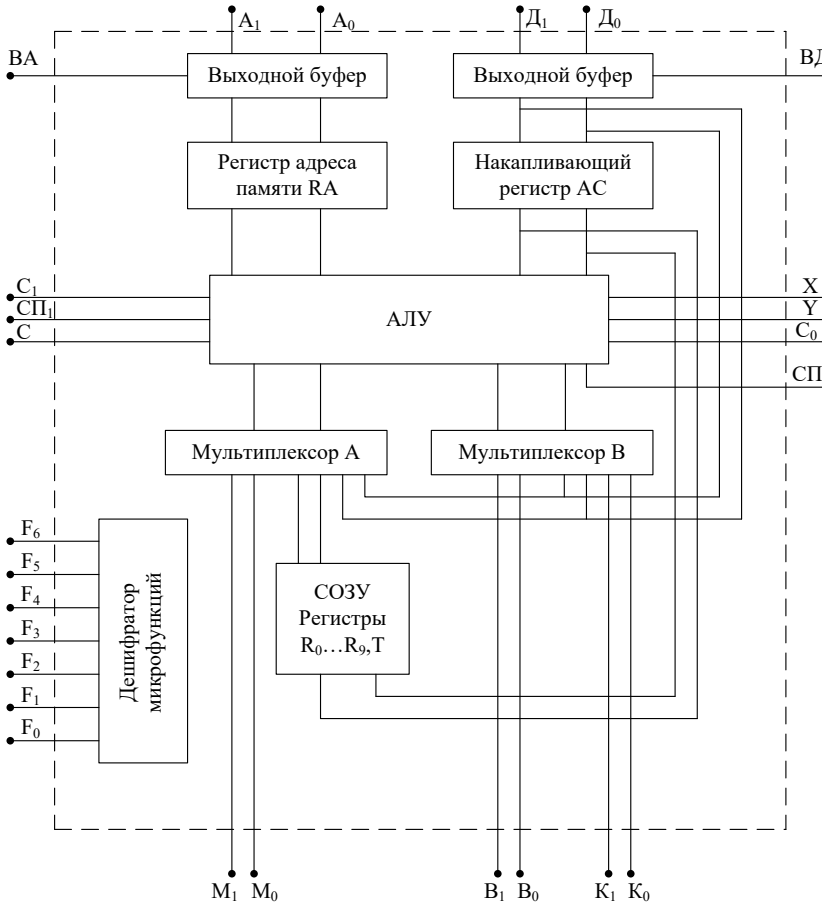

Рис. 6.6. Блок – схема ЦПЭ К529 ИК02

Таблица 6.1.

Перечень микроинструкций ЦПЭ.

F-гр	P-гр	Логико – арифметическое выражение
0	1	$R_n + (AC + K) + C1 \rightarrow R_n$, AC
	2	$M + (AC + K) + C1 \rightarrow AT$
	3	$AT_0 + \overline{(B_0 \wedge K_0)} \rightarrow СП_0$ $СП_1 \vee ((B1 \wedge K1)) \wedge AT_1) \rightarrow AT_1$ $(AT_0 \wedge (B_0 \wedge K_0)) \vee (AT_1 \wedge (B_1 \wedge K_1)) \rightarrow AT_0$
1	1	$K \vee R_n \rightarrow RA$ $R_n + K + C_1 \rightarrow R_n$
	2	$K \vee M \rightarrow RA$ $M + K + C_1 \rightarrow AT$
	3	$(\overline{AT} \vee K) + (AT \wedge K) + C_1 \rightarrow AT$
2	1	$(AT \wedge K) - 1 + C_1 \rightarrow R_n$
	2	$(AT \wedge K) - 1 + C_1 \rightarrow AT$
	3	$(B \wedge K) - 1 + C_1 \rightarrow AT$
3	1	$R_n + (AC \wedge K) + C_1 \rightarrow R_n$
	2	$M + (AC \wedge K) + C_1 \rightarrow AT$
	3	$AT + (B \wedge K) + C_1 \rightarrow AT$
4	1	$C_1 \vee (R_n \wedge AC \wedge K) \rightarrow C_0$ $R_n \wedge (AC \wedge K) \rightarrow R_n$
	2	$C_1 \vee (M \wedge AC \wedge K) \rightarrow C_0$ $M \wedge (AC \wedge K) \rightarrow AT$
	3	$C_1 \vee (AT \wedge B \wedge K) \rightarrow C_0$ $AT \wedge (B \wedge K) \rightarrow AT$
5	1	$C_1 \vee (R_n \wedge K) \rightarrow C_0$ $K \wedge R_n \rightarrow R_n$
	2	$C_1 \vee (M \wedge K) \rightarrow C_0$ $K \wedge M \rightarrow AT$
	3	$C_1 \vee (AT \wedge K) \rightarrow C_0$ $K \wedge AT \rightarrow AT$
6	1	$C_1 \vee (AC_n \wedge K) \rightarrow C_0$ $R_n \vee (AC \wedge K) \rightarrow R_n$
	2	$C_1 \vee (AC \wedge K) \rightarrow C_0$ $M \vee (AC \wedge K) \rightarrow AT$
	3	$C_1 \vee (B \wedge K) \rightarrow C_0$ $AT \vee (B \wedge K) \rightarrow AT$
7	1	$C_1 \vee (R_n \wedge AC \wedge K) \rightarrow C_0$ $R_n \equiv (AC \wedge K) \rightarrow R_n$
	2	$C_1 \vee (M \wedge AC \wedge K) \rightarrow C_0$ $M \equiv (AC \wedge K) \rightarrow AT$
	3	$C_1 \vee (AT \wedge B \wedge K) \rightarrow C_0$ $AT \equiv (B \wedge K) \rightarrow AT$

Перечень микроинструкций ЦПЭ приведен в таблице 6.1. Относительно их дешифрации он секционирован на 8 F-групп (с 0 по 7) по 3 F-группы в каждой. В клетках таблицы приведены логико-арифметические выражения (одно или два), активируемые каждой микроинструкцией. Они и представляют собой исходную информацию для оценки (У,Н)-доступности операторных вершин, объединения их в группы и упорядочения относительно (У,Н)-доступности.

Модель отражает неисправности компонентов двух уровней МПУУ – микропрограммного управления и операционного устройства. К неисправностям первого уровня относятся:

- не возбуждение шины дешифратора для заданного кода;
- возбуждение шины дешифратора для незаданного кода;
- одновременное возбуждение двух или более шин дешифратора для заданного кода.

К неисправностям второго уровня относятся константные неисправности операционных цепей и взаимовлияния соседних разрядов.

Обобщённая оперативная модель относится к классу двухсортных СФУ-моделей (см. раздел 4.3.8.). Как диагностическая модель она характеризуется упорядочением элементов, а, следовательно, и их неисправностей. Ей присуща большая приспособленность к построению тестов по сравнению с распространённой моделью регистровых передач [42], что объясняется большей детальностью в отражении свойств операций и упорядочение последних по (У, Н)-доступности. Относительно функций вершин (хранения и преобразования информации) модель является *гетерогенной*.

6.4.2. Метод построения теста

Построение тестовой последовательности микроинструкций основано на применении метода раскрутки. Он заключается в пошаговом расширении проверенного ядра объекта. При внешнем тестировании к ядру относятся входные и выходные шины объекта.

Проверка начинается с микроинструкций с $У_{min}$ и $Н_{min}$, находящихся на кратчайших путях от входов к выходам структуры. От наблюдаемых регистров она распространяется на перезапись информации в них вначале с первичных входов (если возможно), а затем через промежуточные регистры. Если последние соединены каскадным

способом, последовательно проверяется каждый каскад. Затем проверке подвергается регистровая память объекта с произвольным доступом. Для микроинструкций с одинаковой (У, Н) – доступностью учитывается их упорядоченность в обобщённой операторной модели.

Применение метода раскрутки позволяет обнаружить неисправности дешифрации управляющих кодов за меньшее число шагов, что при условной процедуре диагностирования позволяет сократить длину теста и продолжительность тестирования. Этот метод обладает также большими диагностическими свойствами, так как позволяет указывать на неисправность при первом не сравнении реакций с ожидаемыми реакциями, что согласуется с подходом, предложенным в [53] для комбинационных схем.

Гарантиями обнаружения неисправностей дешифратора микроинструкций являются:

- контроль всех выходов объекта на всех тестовых воздействиях;
- применение полных тестов на константные неисправности для различных выражений, активируемых микроинструкциями μ_i, μ_j, $i \neq j$;
- применение кодовых методов для регистровой памяти с произвольным доступом.

Минимизация теста возможна лишь за счёт сокращения числа тестовых данных (операндов), поскольку на верхнем уровне проверяются все микроинструкции. Сокращение объёма тестовых данных достигается за счёт использования части теста для ранее проверенных операционных цепей. При этом проверяются не менее двух воздействий для различения двух микроинструкций, причём для тех переменных, на которые различаются активируемые выражения. Для проверки цепей передачи данных на взаимовлияние соседних разрядов используются коды типа 52 и 25. Общий алгоритм синтеза теста регистровой микропрограммно - управляемой структуры представлен на рис. 6.8.

Рассмотренные модель и метод были применены для построения теста центрального микропроцессорного элемента ЦПЭ Х589ИК02.

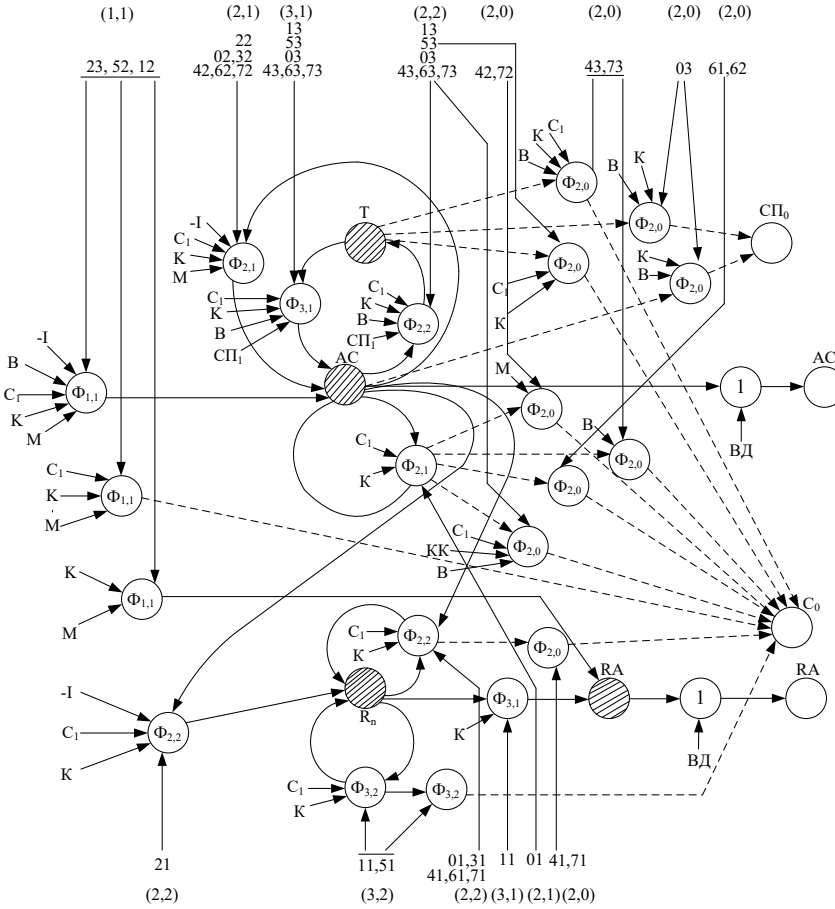

Рис. 6.7. Операторная модель ЦПЭ К529 ИК02

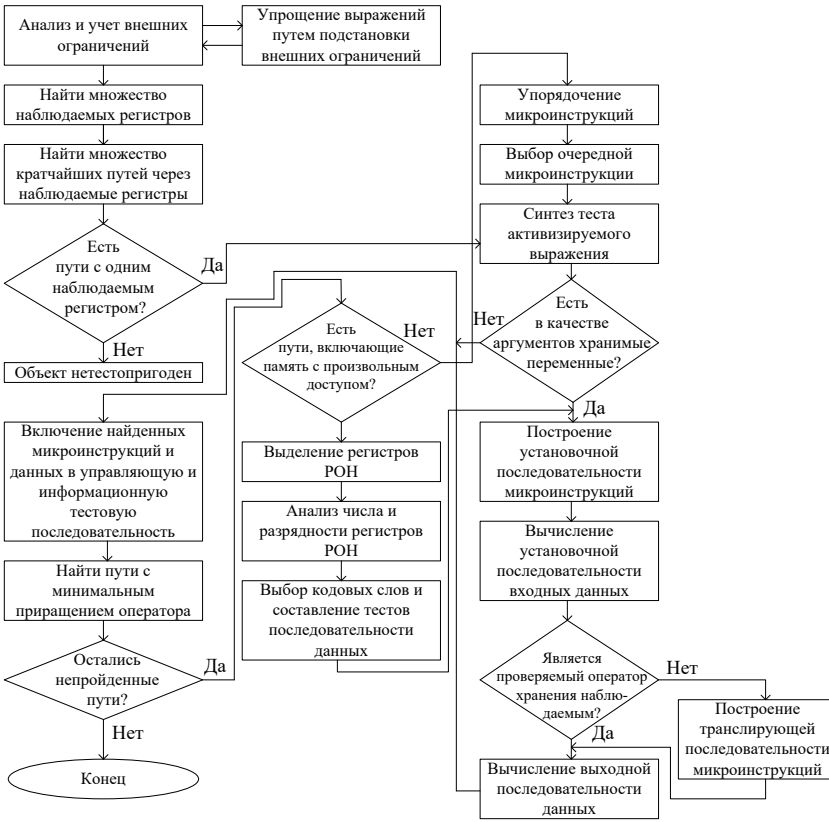

Рис. 6.8. Алгоритм построения теста МПУУ

Глава 7. КОНСТРУИРОВАНИЕ БАЗЫ ЗНАНИЙ ПРОЦЕДУРНОГО ТИПА

7.1. Порождение вариантов знаний

Рассмотренные выше системы понятий, моделей и методов технического диагностировали ВС, построенные с использованием теоретических основ порождения понятий, не исчерпывают всех возможных вариантов понятий, моделей и методов. Полнота систем определяется состоянием предметной области и текущими потребностями практики. Поскольку оба эти фактора изменчивы, системы также подвержены изменениям, а, следовательно, не постоянны как сами варианты предметного знания, так и их предпочтения. С учётом этого полнота систем может быть обеспечена с одной стороны возможностью изменения совокупности базисных признаков и аксиом вывода из них производных вариантов, а с другой стороны – возможностью получения *любого* варианта знания. Эта задача разрешима с привлечением аппарата *перечислимых* множеств и формальных систем [70].

Существенной особенностью процедуры порождения каждого варианта в рассмотренных системах является последовательное приращение (конкатенация) составляющих его признаков. В качестве начального признака в системе понятий принимается родовое понятие подсистемы, в системе моделей – свойство объекта диагностирования, принятое за основное, в системе методов – ядро функционального базиса. Таким образом, l-й вариант представляется цепочкой признаков $\mathbf{c}_l$, начинающейся с начального признака C_0, а все возможнее варианты – множеством цепочек C [117].

Очевидно, что цепочки $\mathbf{c}_l$ и $\mathbf{c}_m$ различаются между собой либо количеством признаков – длиной L, либо признаками $C_{\rho l}$ и $C_{\rho m}$ в ρ-й позиции. Поскольку в содержании понятия не допускается повторений какого-либо существенного признака, каждый признак может входить в цепочку только *один* раз. Исходя из изложенного, для порождения всех вариантов, образующих систему, необходимо знать начальный признак C_0 и перечни дополнительных признаков C_ρ, $\rho=1,\ldots,L$. Они и образуют аксиомы формальной системы Ф.

Если дополнительные признаки неизвестны, то они должны порождаться в рамках других формальных систем или вручную пользователем.

Множество порождаемых на основе формальной системы Ф цепочек представляет собой дерево с общей вершиной C_0. Их количество определяется числом висячих (конечных) вершин дерева. Для N_ρ признаков на ρ-м ярусе дерева, $\rho = \overline{1, L}$, оно равно $\prod_{\rho=1}^{L} N_\rho$. При N_1=,…,= N_ρ=,…,=N_L=2 (дихотомия) оно равно 2^L. Трудоемкость перебора вариантов быстро растет с ростом L и N_ρ. Согласно [61] приемлемое число ярусов дихотомического дерева для его построения на современной ЭВМ за единицы секунд равняется примерно двадцати. Однако просмотр такого количества вариантов невозможен для пользователя.

В реальных условиях часть *синтаксически* правильно описанных признаков и цепочек не являются *семантически* правильными. Это означает, что они лишены смысла, поскольку не имеют аналогов в реальной действительности, например, в силу ограничений решаемой адачи. Семантически неправильный признак $C_{n\rho} \in C_\rho$ не обладает заданной совокупностью свойств $P=\{P_1,\ldots,P_t\}$ или одним из них. Семантически неправильная цепочка $\mathbf{c}_m$ не обладает заданной совокупностью интегральных свойств $P_c=\{P_{c1},\ldots,P_{ct}\}$ или хотя бы одним из них.

Интегральное свойство определяется на всех признаках цепочки. Количественно оно оценивается задаваемым в задаче функционалом. Относительно этой оценки цепочка $\mathbf{c}_l$ считается семантически правильной, если её функционал достигает заданного значения.

Задача построения семантически правильных цепочек относится к задачам выбора. Нахождение цепочек с экстремальным значением функционала является экстремальной задачей выбора. Критерием выбора признака $C_{n\rho}$ на ρ-м шаге процедуры синтеза цепочки $\mathbf{c}_l$ может являться достижение экстремального значения функционала – max $F(C_{n\rho})$ или min$F(C_{n\rho})$ с целью минимизации числа признаков в цепочке. Эта задача решается с помощью методов локальной оптимизации, не гарантирующих

нахождения глобального оптимума. В качестве примера сошлёмся на принцип выбора тестового воздействия в ТФН по максимуму приращения информации как функции разбиения её неисправностей.

Все названные задачи на дереве вариантов являются *NP*-полными. Сокращение перебора достигается за счёт исключения из него семантически неправильных цепочек в процессе их синтеза. С этой целью признак $C_{n\rho}$, не обладающий свойством $P_i \in P$ или не улучшающий интегральное свойство $P_{ci} \in P_c$, включается в список запрещенных признаков B. Поскольку он не включается в ρ-ю позицию синтезируемой цепочки, то и все следующие за ним признаки $C_{n\rho},\ldots,C_{nL}$ также исключаются из рассмотрения. Количество исключаемых из-за запрета признака $C_{n\rho}$ цепочек равно $\prod_{t=\rho}^{L} N_t$. Естественно, что чем ранее обнаружится запрет признака, там значительнее сокращение перебора.

Если запрет признака $C_{n\rho}$ относительно свойства $P_h \in P$ сохраняется на всё время порождения цепочек, то сохранение запрета относительно интегрального свойства $P_{sq} \in P_s$ зависит от постоянства признаков C_ρ, $\rho = \overline{1,L}$. Если список признаков C_ρ меняется в процессе порождения цепочек, то после очередного изменения признак $C_{n\rho}$ исключается из списка запрещённых. Это объясняется тем, что он может улучшить интегральное свойство P_{sq} новой подцепочки: $C_0, C_{1h},\ldots, C_{\rho-1,q}$.

Метод сокращения перебора, подобный изложенному, назван в [118] методом фокусирования поиска. При использовании для отсечения неинформативных ветвей дерева функционалов цепочек этот метод трансформируется в метод ветвей и границ.

Итак, *l*-й вариант знания будем представлять цепочкой признаков

$$\mathbf{c}_l = (C_0, C_{n1},\ldots,C_{n\rho},\ldots,C_{nL}),$$

где $C_{n\rho} \in C_\rho$, $C_\rho = \{ C_{1\rho},\ldots,C_{n\rho},\ldots,C_{N\rho}\}$, $\rho = \overline{1,L}$.

Он синтезируется итеративно путем присоединения (конкатенации ○) на ρ-м шаге процедуры к цепочке $\mathbf{c}_{\rho-1,l}=(C_0, C_{n1},\ldots,C_{n\rho},\ldots,C_{n\rho-1})$ признака $C_{N\rho}$:

$$\mathbf{c}_{\rho,l}=(C_0, C_{n1},\ldots,C_{n\rho},\ldots,C_{n\rho-1}, C_{N\rho}).$$

Моделью порождения вариантов является формальная система общего вида Ф = <T, P, A, П>. Базовые элементы (алфавит) и синтаксические правила её предметно ориентированы, а аксиомы и правила вывода инварианты относительно различных конкретных формальных систем:

$$\mathrm{A} = \{ \mathrm{C}_0, \mathrm{C}_\rho \}, \qquad \rho = \overline{1, L}.$$

$$\Pi : \frac{\mathbf{c}_{\rho\text{-}1,l}, \bigwedge_{i=1}^{q} P_i(C_{n,\rho}), \bigwedge_{j=1}^{h} P_{cj}(\mathbf{c}_{\rho,l})}{\mathbf{c}_{\rho,l}}.$$

Цепочки, образующие множество вариантов знания С, удовлетворяют совокупности целевых свойств $\bigwedge_{j=1}^{h} P_{cj}(\mathbf{c}_{\rho,l})$ или заданной длине $s \le L$:

$$\mathrm{C} = \{\mathbf{c}_{\rho,l} \mid \bigwedge_{j=1}^{h} P_{cj}(\mathbf{c}_{\rho,l}) \vee (\rho = s)\}.$$

Варианты знания из С, удовлетворяющие k заданным свойствам, образуют множество селектированных (отобранных) вариантов:

$$\mathrm{C}_{\lim} = \{\mathbf{c}_l \mid \bigwedge_{t=1}^{k} P_t(\mathbf{c}_l)\, r^{(2)}\, P_{t,\lim}\}.$$

Двухместное отношение $r^{(2)}$ интерпретируется либо арифметическими предикатами, либо вербально, как соответствие или несоответствие t-му условию, $t = \overline{1, k}$.

Приведём алгоритм порождения вариантов предметного знания, общий для различных применений.

1. l:=1;
2. $\mathbf{c}_l = \varnothing$; $\forall B_r \in B$ (B_r:=true);
3. ρ:=0; $\mathbf{c}_l$:=C_0;
4. ρ:=ρ+1;
5. n_ρ:=1;
6. Если $\exists P_i \in P$ $P_i(\mathrm{C}_{n\rho})$=false, идти к 10;

7. $\mathbf{c}_{\rho l} := \mathbf{c}_{\rho-1,l} \circ C_{n\rho}$;
8. Если $\forall P_{cj} \in P_c\ (P_{cj}\ (\mathbf{c}_{\rho-1,l} \circ C_{n\rho})) = P_{cj}\ (\mathbf{c}_{\rho,l})$, идти к 10;
9. Если $\rho < L$ или $\exists P_{cj} \in P_c\ \ (P_{cj}(\mathbf{c}_{\rho-1,l} \circ C_{n\rho}) \neq P_{gj}$, идти к 4, иначе к 11 (цепочка завершена);
10. $B(n_\rho)$:=false;
11. $n_\rho := n_\rho + 1$;
12. Если $B(n_\rho)$=false, идти к 11;
13. Если $n_\rho \le N_\rho$, то $l := l+1$;
14. $\rho := \rho - 1$;
15. Если $\rho=0$, то конец, иначе идти к 5.

Алгоритм реализует обход дерева цепочек «сверху-вниз» и «слева-направо» относительно перечней признаков C_ρ, $\rho = \overline{1, L}$. Признаки, не удовлетворяющие условиям P и P_c, маскируются булевым вектором B и исключаются из последующего перебора.

Применительно к конкретным знаниям формальная система конкретизируется следующим образом: формируется алфавит T и синтаксические правила P, определяются аксиомы A, интерпретируются правила вывода П и условия P, P_c и P_{lim}. На этой основе уточняется алгоритм порождения вариантов знания.

7.2. Оценивание вариантов

Проиллюстрируем её на примере сопоставления методов синтеза теста поисков дефектов. В качестве оцениваемых критериев примем длину теста поиска дефектов L, объем памяти V, требуемый для запоминания тестовой информации, и трудоемкость дешифрации, выражаемую числом n элементарных (побитных) сравнений экспериментальной и заданной диагностической информации [54].

В качестве диагностической модели ОД примем таблицу функций неисправностей (ТН). Она универсальна по отношению к объектам диагностирования и методам поиска. Кроме того, она удобна для получения верхних и нижних оценок перечисленных выше критериев.

Строкам ТН соответствуют векторы значений $\mathbf{u}_i=(u_{i1},\ldots,u_{im})$ входных переменных $x_1,\ldots,x_m$ а столбцам – модификации ОД – исправная M_0 и неисправные $M_0,\ldots,M_j,\ldots, M_N$.

Каждая неисправная модификация M_j порождается любым дефектом r_{jk} из класса эквивалентности R_j. Содержимым таблицы функций неисправностей, как известно, являются значения выходной переменной, определенные на входных наборах переменных. Это создает определенные неудобства для анализа. Поэтому будем использовать таблицу различения неисправностей (ТРН) [52]. Её содержимым являются предикаты неравенства $p_{ij},\ i=\overline{1,L},\ j=\overline{1,N+1}$. Значение p_{ij} вычисляются следующим образом:

$$p_{ij}=\begin{cases}1, \text{ если } M_j(\mathbf{u}_i)\neq M_0(\mathbf{u}_i);\\ 0, \text{ если } M_j(\mathbf{u}_i)=M_0(\mathbf{u}_i).\end{cases}$$

Согласно этим условиям исправной модификации M_0 в ТРН соответствует нулевой столбец, отражающий совпадение реакций исправного объекта с его эталонными реакциями.

Сопоставим три базовых метода поиска, использующие в качестве модели ТРН: комбинационный поиск (КП), последовательный поиск с пошаговым *разбиением* модификаций (ПРМ) и последовательный поиск с пошаговым *выделением* модификаций (ПВМ). Для иллюстрации метода поиска на примере в качестве общей для них модели будем использовать ТРН, представленную в табл. 7.1. Она отражает различие реакций пяти неисправных модификаций с реакциями исправного объекта на последовательности входных воздействий *a*, *b*, *c*, *d*.

Таблица 7.1.

Входные воздействия	M_0	M_1	M_2	M_3	M_4	M_5
a	0	1	0	0	0	0
b	0	0	1	0	1	1
c	0	0	0	1	1	1
d	0	0	0	0	0	1

Выведем оценки выбранных параметров для каждого из названных методов.

7.2.1. Метод ПРМ

Тест, реализующий поиск с помощью *равномерного разбиения модификаций* на каждом входном наборе, строится с использованием следующего информационного критерия локальной оптимизации [50].

$$I_i = \sum_{i=1}^{k_i} \frac{N_i}{N+1} \log_2 \frac{N_i}{N+1} \qquad (7.2)$$

где I_i – приращение информации о модификациях после применения i-го входного воздействия, k_i – число подмножеств модификаций, различаемых с помощью i-го воздействия, N_i – число модификаций в i-ом подмножестве.

На каждом шаге построения тестовой последовательности выбирается входное воздействие, обладающее наибольшим значением I_{max}. Построение теста завершается, когда в каждом различимом подмножестве оказывается по одной модификации ОД.

При построении теста можно воспользоваться более простым – весовым критерием [119].

$$w_i = \sum_{j=1}^{k_i} n_j^0 \cdot n_j^1 \qquad (7.2)$$

где n_s^0 и n_s^1 – соответственно число нулей и единиц в j-м подмножестве модификаций, а k_j – число подмножеств на j-м уровне.

Для рассматриваемого примера (табл. 7.10) подсчёт весов воздействий на каждом шаге приведен в табл. 7.2.

Таблица 7.2.

Входные воздействия	Весовые категории на i-ом шаге		
	w_1	w_2	w_3
a	5·1=5	2·1+3·0=2	1·1+2·0=**1**
b	3·3=9	2·1+1·2=**4**	
c	3·3=**9**		
d	5·1=5	3·0+2·1=2	2·0+1·1=**1**

Из двух воздействий *b* и *c* с весом 9 на первом шаге выбрано *c*, на втором

– b с w_{2max}=4. Из двух равноценных воздействий a и d на третьем шаге выбрано a. Последним включено в тестовую последовательность воздействие d. При условном поиске в тестовую последовательность включается либо воздействии a, либо d, в зависимости от исхода тестирования на втором воздействии. Дерево поиска, соответствующее полученной тестовой последовательности $\mathbf{x}_T$=(c, b, a, d) изображено на рис. 7.1.

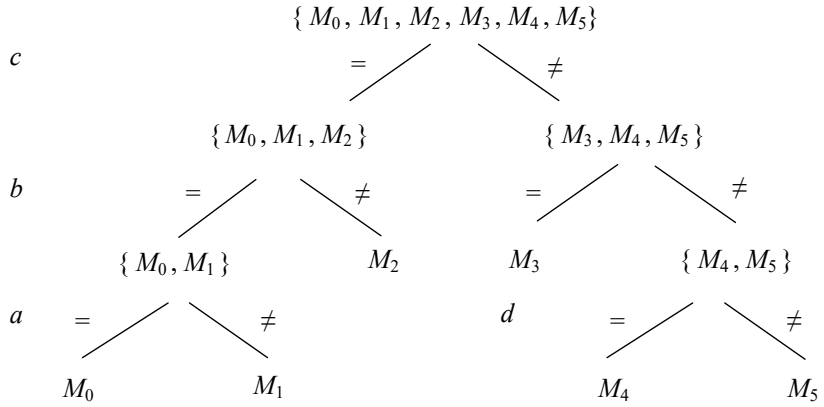

Рис. 7.1. Дерево поиска с равномерным разбиением модификаций

Длина полученной в примере тестовой последовательности соответствует её минимальной оценке [52]:

$$L_{min} =]\log_2(N+1)[\qquad (7.3)$$

Максимальная длина теста имеет место для ТРН, представляющей собой единичную матрицу с дополнительным нулевым столбцом:

$$L_{max} = \begin{cases} N \text{ при } N \le 2^m; \\ 2^m \text{ при } N > 2^m. \end{cases} \qquad (7.4)$$

Для N=1, 2, $2^{2^m-1} - 2^{2^m} - 1$ формулы (7.3) и (7.4) дают одинаковые оценки.

Объём памяти, требуемый для реализации ПРМ, определяется размерностью ТРН:

$$V = m + N + 1 \tag{7.5}$$

Эта формула характеризует полный объём ТРН, включая занимаемый самим тестом.

Количество сравнений, требуемое для различения всех модификаций методом ПРМ, определяется следующей общей формулой:

$$n_{\text{ПРМ}} = \sum_{i=1}^{L} N_i, \tag{7.6}$$

где N_i – количество модификаций, подлежащих разделению на i-м шаге поиска. Если начальное количество модификаций равно N+1 и на каждом шаге поиска оно уменьшается в среднем вдвое, то при использовании теста с минимальной длиной $L_{\min}$ минимальное количество сравнений определится следующей разновидностью формулы (7.6):

$$n_{\text{ПРМ min}} = \sum_{i=1}^{]\log_2(N+1)[}]\frac{N+1}{2^{i-1}}[,$$

Для условия $2^{L_{min}} = N + 1$, полученного потенцированием выражения (7.3), эту формулу можно рассматривать как сумму членов убывающей геометрической прогрессии со знаменателем ½:

$$n_{\text{ПРМ min}} = \frac{N+1-2\cdot 1/2}{1-1/2} = 2\cdot N. \tag{7.7}$$

Максимальное количество сравнений имеет место при максимальной длине теста и сокращении числа сравнений на каждом шаге на единицу. С учётом (7.4) формула (7.6) для случая $N \le 2^m$ преобразуется к виду:

$$n_{\text{ПРМ max}} = \sum_{i=1}^{L}(N+2-i) = N/2 \cdot (N+3). \tag{7.8}$$

Первая часть этой формулы определена как сумма членов убывающей арифметической прогрессии с разностью i=1.

Для случая $N > 2^m$ вместо (7.8) имеет место неравенство:

$$n_{\text{ПРМ max}} < N/2 \cdot (N + 3).$$

При $N = 2^{2^m} - 1$ и $m>3$, используя формулу (7.7) получим

$$n_{\text{прм}} = 2^{2^{m+1}}.$$

Метод КП отличается от ПРМ лишь способом дешифрации результатов тестирования. Поэтому длина реализующего его теста также оценивается формулами (7.3) и (7.4), а объём занимаемой памяти – формулой (7.5).

Поскольку алгоритм КП характеризуется сканированием столбцов ТРН до положительного исхода сравнения одного из них со столбцом, полученным при тестировании, максимальное количество сравнений определяется перебором всех столбцов ТРН:

$$n_{\text{КП}} = L \cdot (N + 1). \qquad (7.9)$$

7.2.2. Метод ПВМ

Тест, реализующий поиск путём *последовательного выделения модификаций*, строится на основе отношения доминирования. *j*-я и *k*-я модификации объекта, $j,k = \overline{0,N}$, $j \neq k$, представленные в ТРН столбцами q_j и q_k, находятся в отношении доминирования первая над второй ($M_j \succ M_k$), если $q_j \vee q_k = q_j$. Этому выражению соответствует покрытие столбца q_k единицами столбца q_j. Если $q_j \vee q_k \neq q_j$ и $q_j \vee q_k = q$, то модификации M_j и M_k не находятся в отношении доминирования. Порядок их выделения безразличен.

Алгоритм упорядочения модификаций заключается в выявлении отношения доминирования между ними путём их попарного сравнения. Для приведенного в табл. 7.1 примера имеет место следующее бинарное отношение доминирования:

$$r_{\succ} = \{(M_5, M_4), (M_5, M_3), (M_5, M_2), (M_4, M_3), (M_4, M_2), (M_5, M_0), (M_4, M_0), (M_3, M_0), (M_2, M_0), (M_1, M_0)\}.$$

Одним из вариантов цепи модификаций, построенной на основе отношения $r_{\succ}$, является: M_5, M_4, M_3, M_2, M_1, M_0. Поскольку модификация M_1 не находится в отношении доминирования ни с одной из предшествующих ей в цепочке модификаций, она может занимать любую

позицию в цепочке до M_0. Тестовая последовательность, выделяющая модификации в установленном порядке, состоит из N фрагментов (идентификаторов [53]), j-й фрагмент, $j = \overline{1,N}$, включает подмножество входных воздействий, отличающих j-ю модификацию от последующих j+1, j+2,…,N. Минимизация j-го фрагмента осуществляется путём преобразования выражения ΠΣ в ΣΠ [51].

Для рассматриваемого примера минимальные совокупности воздействий, выделяющие модификации в заданной последовательности, находятся следующим образом:

M_5: $\Pi\Sigma_5 = d\cdot(b \vee d)\cdot(c \vee d)\cdot(b \vee c \vee d)\cdot(b \vee c \vee d) \rightarrow \Sigma\Pi_5 = d$;

M_4: $\Pi\Sigma_4 = b \cdot c \cdot(b \vee c)\cdot(b \vee c) \rightarrow \Sigma\Pi_4 = b \cdot c$;

M_3: $\Pi\Sigma_3 = (c \vee b)\ )\cdot(c \vee a)\cdot c \rightarrow \Sigma\Pi_3 = c$;

M_2: $\Pi\Sigma_2 = (b \vee a)\cdot b \rightarrow \Sigma\Pi_2 = b$;

M_1: $\Pi\Sigma_1 = a \rightarrow \Sigma\Pi_1 = a$.

Результирующая минимальная тестовая последовательность, позволяющая выделить модификации в заданном порядке, имеет вид: $\mathbf{x}_{r1}$=(d; b,c; c; b; a). В неё входят 5 фрагментов, разделённых точкой с запятой, причем второй из них состоит из двух воздействий. Дерево поиска, соответствующее этой последовательности, приведено на рис. 7.2. Выделение модификаций в нём осуществляется по не сравнению получаемых реакций объекта с ожидаемыми на каждом фрагменте тестовой последовательности. Упорядочение модификаций для их поиска может осуществляться также на основе отношения обратного доминирования, устанавливаемого между j-й и k-й модификациями, $j,k = \overline{0,N}$, $j \neq k$, с помощью выражения $d_j \wedge d_k = d_j$, для которого $M_j \succ M_k$. Ему соответствует покрытие столбца d_k нулями столбца d_j.

Для рассматриваемого примера цепь модификаций, построенная на основе отношения доминирования нулей, имеет обратную направленность: M_0, M_1, M_2, M_3, M_4, M_5.

Тестовая последовательность для выделения модификаций в заданном порядке, построенная вышеизложенным способом, также состоит из пяти фрагментов.

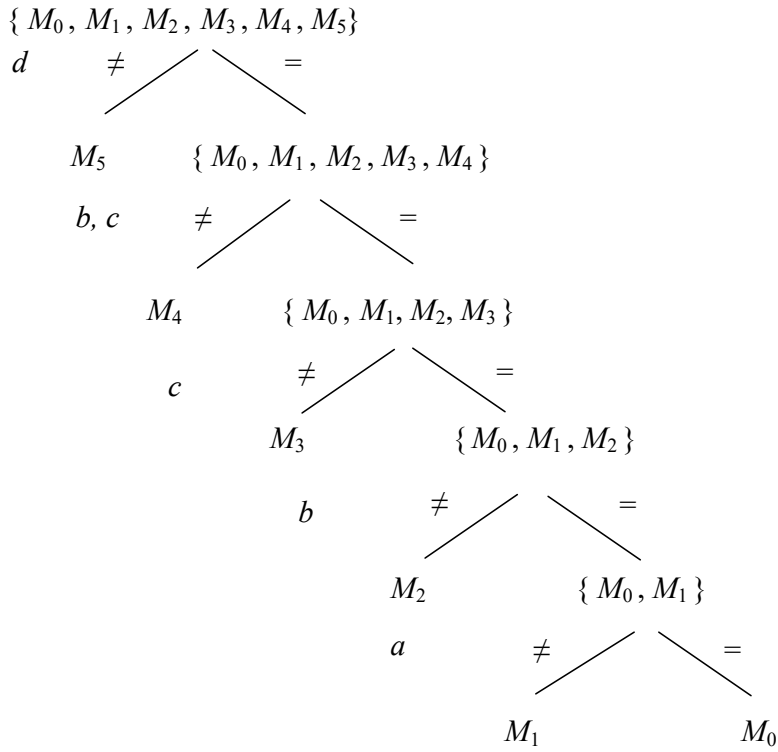

Рис. 7.2. Дерево поиска с выделением модификаций
по не сравнению

Первый фрагмент, выделяющий исправную модификацию M_0, представляет собой тест контроля. Его длина равна трем воздействиям. Общая длина тестовой последовательности $\mathbf{x}_{T2}$ на 2 воздействия больше, чем у $\mathbf{x}_{T1}$.

Дерево поиска, соответствующее $\mathbf{x}_{T2}$, приведено на рис. 7.3. Выделение модификаций в нем осуществляется по положительному исходу сравнения получаемых реакций с ожидаемыми на каждом воздействии тестовой последовательности.

Из сопоставления последовательностей xT1 и xT2 следует, что их длина зависит от выбора типа отношения доминирования (единиц или нулей в столбцах ТРН). С целью минимизации тестовой последовательности необходимо анализировать соотношение единиц и нулей в ТРН объекта. Если единиц меньше, выбирается отношение доминирования единиц, а в противном случае – нулей. Учитывая это, для получения минимальной и максимальной оценок длины теста ПВМ, будем использовать предельные соотношения единиц и нулей объекта. Минимальное количество единиц содержится в булевой матрице A1 размерности N×(N+1) следующего вида:

$$\mathbf{A}_1 = \begin{vmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix}$$

Она состоит из нулевого столбца, соответствующего исправной модификации M_0, и единичной матрицы порядка $N \times N$, представляющей N неисправных модификаций.

Минимальное количество нулей содержится в булевой матрице $\mathbf{A}_2$ размерности $N \times (N+1)$, включающей матрицу порядка $N \times N$ с нулевой диагональю:

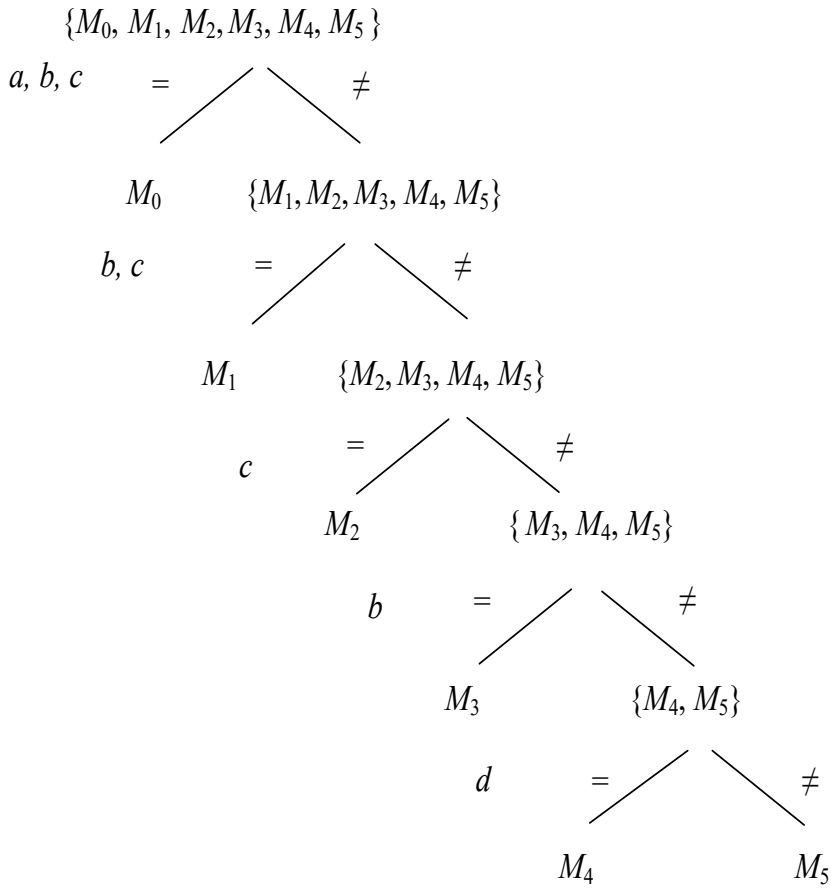

Рис. 7.3. Дерево поиска с выделением модификаций по сравнению

$$\mathbf{A}_2 = \begin{vmatrix} 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \end{vmatrix}$$

При доминировании единиц минимальная оценка длины теста $L_{\text{ПВМ1}}$ определяется по матрице $\mathbf{A}_1$. Поскольку в ней все столбцы диагональной матрицы не находятся между собой в отношении доминирования, для различения каждого из них от других достаточно одно воздействие. Следовательно, $L_{\text{ПВМ1,min}}=N$.

Минимальная оценка $L_{\text{ПВМ}}$ при доминировании нулей определяется по матрице $\mathbf{A}_2$ следующим образом. При выделении нулевого столбца необходимо и достаточно использовать любые два воздействия, отличающие его от всех других столбцов. Каждый из столбцов диагональной матрицы отличается от других одним воздействием. Таким образом, общее количество воздействий $L_{\text{ПВМ0,min}}=2+N-1=N+1$.

Максимальная оценка $L_{\text{ПВМ0}}$ находится по матрице $\mathbf{A}_2$ с учетом того, что все её столбцы, кроме второго, находятся в отношении доминирования единиц. Для выделения каждого столбца, начиная с последнего, необходимо использовать все единицы, находящиеся под нулем. Они определяют необходимое количество воздействий, отличающих рассматриваемый столбец от последующих столбцов. Общее количество воздействий, различающих все столбцы, кроме второго, определяется количеством единиц в треугольной матрице, равным $N\cdot(N-1)/2$.

Поскольку второй столбец различается с помощью одного последнего воздействия, общее число их равно:

$$L_{\text{ПВМ0 max}} = \frac{N(N-1)}{2}+1. \qquad (7.10)$$

Для нахождения максимальной оценки $L_{\text{ПВМ0,max}}$ следует использовать матрицу $\mathbf{A}_1$. Все столбцы в ней, начиная с первого, находятся в отношении доминирования нулей. Учитывая то, что количество различающих воздействий для выделения каждого столбца определяется количеством нулей, находящихся под единицей, общее количество

воздействий тестовой последовательности равно количеству нулей в нулевом столбце и треугольной матрице, входящей в $\mathbf{A}_1$. Отсюда

$$L_{\text{ПВМ0,max}} = N\cdot(N+1)/2. \qquad (7.11)$$

Из сопоставления полученных нижних и верхних оценок следует большая длина теста ПВМ, реализующего доминирование нулей. Это объясняется вхождением в него теста контроля, отличающего исправную модификацию от всех остальных. Тестовая последовательность, реализующая доминирование единиц, решает задачу контроля в процессе поиска неисправных модификаций объекта – в случае отрицательного его исхода. Совмещение задач поиска и контроля и обеспечивает её меньшую длину.

Для комбинационных схем число строк ТРН ограничивается длиной тривиального теста $L=2^m$. При этом оценки, полученные на основе матриц $\mathbf{A}_1$ и $\mathbf{A}_2$, справедливы для $N=2^m$. При $N>2^m$ $L_{\text{ПВМ min}}>N+1$.

Оценим $L_{\text{ПВМ}}$ для полной ТРН, состоящей из 2^m строк и $N+1=2^{2^m}$ столбцов. Вследствие того, что она содержит равные количества единиц и нулей, оценки $L_{\text{ПВМ max}}$ и $L_{\text{ПВМ min}}$ совпадают и равны числу единиц (нулей) в ТРН:

$$L_{\text{ПВМ}} = 2^m \cdot 2^{2^m} \cdot \frac{1}{2} = 2^{2^m+m-1}. \qquad (7.12)$$

Обобщением этой оценки для ТРН с $\log_2(N+1)$ строками и столбцами является:

$$L_{\text{ПВМ}} = \frac{1}{2}\cdot(N+1)\cdot\log_2(N+1) \qquad (7.13)$$

В оценках (7.13) и (7.11) $\log_2(N+1)/2$ и $N/2$ определяют половину строк ТРН, используемых для различения модификаций.

Когда число строк ТРН, необходимых для различения модификаций, лежит между $\log_2(N+1)/2$ и $N/2$ и модификации выделяются по преобладающим в ТРН значениям, максимальная длина тестов может превышать половину общего числа нулей и единиц в ТРН. Поэтому в общем случае максимальная длина теста определяется неравенством:

$$L_{\text{ПВМ1 max}} \le (L-1)\cdot(N+1) \qquad (7.14)$$

Объём памяти, требуемый для реализации метода ПВМ, оценивается

произведением объёма памяти, занимаемого элементарным тестом, на длину теста:

$$V_{\text{ПВМ}} = (m + 1)\cdot L_{\text{ПВМ}}. \quad (7.15)$$

Количество сравнений при поиске методом ПВМ равно:

$$n_{\text{ПВМ}} = L_{\text{ПВМ}}. \quad (7.16)$$

Таким образом, получены минимальные и максимальные оценки параметров L, V и n для всех методов поиска: ПРМ, КП и ПВМ.

7.2.3. Сопоставление методов ПВМ с методами ПРМ и КП

Для сопоставления методов применяются коэффициенты α, β, γ, имеющие при предпочтительности метода ПВМ значение, меньше 1:

$$\alpha = \frac{L_{\text{ПВМ}}}{L}, \quad \beta = \frac{V_{\text{ПВМ}}}{V}, \quad \gamma = \frac{n_{\text{ПВМ}}}{n_{\text{прм (кп)}}}. \quad (7.17)$$

Воспользовавшись полученными выше оценками, найдём минимальные и максимальные оценки коэффициентов α, β, γ.

Для этого применим следующие соотношения:

$$\alpha_{\text{мин}} = \min(\alpha) = \frac{L_{\text{ПВМ}_{\text{мин}}}}{L_{\text{макс}}},$$

$$\alpha_{\text{макс}} = \max\ (\alpha) \le \frac{(L-1)(N+1)}{L} = \left(1 - \frac{1}{L}\right)(N+1),$$

$$\beta = \frac{m+1}{m+N+1}\alpha, \quad \gamma_{\text{кп}} = \frac{\alpha}{N+1}, \ \gamma_{\text{ПРМ}} = \frac{L_{\text{ПВМ}}}{n_{\text{ПРМ}}}.$$

При этом

$$\gamma_{\text{ПРМ}_{\min}} = \min\left(\frac{L_{\text{ПВМ}}}{n_{\text{ПРМ}}}\right) = \frac{L_{\text{ПВМ}_{\min}}}{n_{\text{ПВМ}_{\max}}},$$

$$\gamma_{\text{ПРМ}_{\text{макс}}} = \max\left(\frac{L_{\text{ПВМ}}}{n_{\text{ПРМ}}}\right) < \frac{L_{\text{ПВМ}_{\max}}}{n_{\text{ПРМ}_{\min}}},$$

где $n_{\text{ПРМ}_{\max}}$ определяется выражением (7.7).

Пример. Сопоставим методы ПРМ, КП и ПВМ по параметрам L, V и n применительно к КС, реализующей функцию «сложение по mod 2» (рис. 7.4). ТРН схемы представлена в табл. 7.3. Она является полной, так как включает все $2^{2^m} = 16$ модификаций КС, одну исправную M_0 и 15 неисправных. Модификации M_1-M_{10} порождаются одиночными неисправностями, M_{11}-M_{14} – кратными неисправностями, а M_{15} соответствует инвертированию выхода КС.

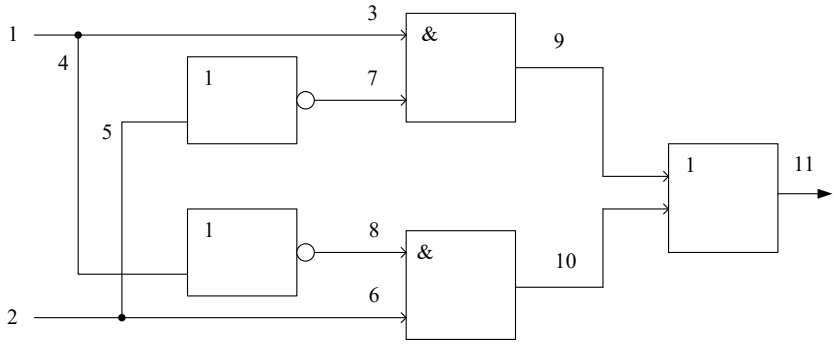

Рис. 7.4. Схема «сложения по mod 2»

Тест, реализующий метод ПРМ, включает все наборы a, b, c, d, причём их последовательность для данной ТРН безразлична. Следовательно, $L_{\text{ПВМ}} = 4$.

Тестовая последовательность, выделяющая модификации в таблице 7.5 в порядке убывания номеров, имеет вид: $\mathbf{x}_T$= (a, b, c, d; b, c, d; a, c, d; a, b, d; a, b, c; c, d; b, d; b, c; a, d; a, c; a, b; d; c; b; a).

Таблица 7.3.

x_1	x_2		1	2	3	4	5	6	7	8	9	10	11	12	13	14	0
0	0	a	1				1	1	1				1	1	1		1
0	1	b		1			1			1	1		1	1		1	1
1	0	c			1			1		1		1	1		1	1	1
1	1	d				1			1		1	1		1	1	1	1

Тестовая последовательность $\mathbf{x}_T$ включает 32 входных набора, что соответствует оценке вычисленной по формуле (7.12) для m=2.

Объём памяти, требуемый для реализации методов ПРМ и ПВМ, соответствует оценкам, определенным по формулам (7.5) и (7.15): $V_{\text{ПРМ}} = 72$, $V_{\text{ПВМ}} = 96$.

Количество сравнений, выполняемых при реализации методов ПРМ, ПВМ и КП, соответствует оценкам, вычисленным по формулам (7.8), (7.16) и (7.9): $n_{\text{ПРМ min}}$=30, $n_{\text{ПВМ}}$=32, $n_{\text{КП min}}$=64.

Соотношение вычисленных показателей:

$$\frac{L_{\text{ПВМ}}}{L_{\text{ПРМ}}} = \frac{32}{4} = 8\,,\ \frac{V_{\text{ПВМ}}}{V_{\text{ПРМ}}} = \frac{96}{72} = 1.33\,,\ \frac{n_{\text{ПВМ}}}{n_{\text{ПРМ}}} = \frac{32}{30}.$$

характеризует метод ПРМ как более предпочтительный для случая $N+1=2^{2^m}$.

Другому крайнему случаю $N=2^m$ – диагональной матрице, включающей модификации M_1, M_2, M_3, M_4, M_5, а также M_0, соответствуют следующие оценки параметров:

$$L_{\text{ПРМ max}}=4,\ L_{\text{ПВМ min}}=4,$$
$$V_{\text{ПРМ max}}=28,\ V_{\text{ПРМ min}}=12,$$
$$n_{\text{ПРМ max}}=14,\ n_{\text{ПВМ max}}=4,\ n_{\text{КП max}}=20.$$

При равных длинах тестов по параметрам V и n метод ПВМ имеет предпочтение перед методом ПРМ:

$$\beta_{\min} = \frac{V_{\text{ПВМ}}}{V_{\text{ПРМ}}} = \frac{3}{7}\,,\ \gamma_{\text{ПРМ}_{\min}} = \frac{2}{7}\,,\ \gamma_{\text{КП min}}=0{,}2.$$

Полученные оценки показывают, что для диагностической модели комбинационной схемы, представляющей собой полную ТРН с максимально возможным количеством неисправностей, поиск с равномерным разбиением модификаций является более экономичным по сравнению с поиском путей выделения модификаций. Последний не имеет преимущество для ТРН с $N<N_{\max}$, столбцы в которых не находятся в отношении доминирования единиц (нулей). Таким образом, выбор более экономичного метода поиска должен предваряться вычислением параметров L, V и n.

Для полной ТРН ($2^m \cdot N_{max}$) и ТРН, представляющей собой диагональную матрицу ($N \cdot (N+1)$ при $N=2^m$), оценки параметров L, V и n инварианты по отношению к функциям комбинационных схем.

7.3. Архитектура базы знаний процедурного типа

Система, выполняющая функции порождения вариантов предметного знания, относится к классу баз знаний, поскольку она оперирует с интенсионалами понятий. Согласно [63] интенсионал «как правило, задаёт некоторую процедуру, позволяющую определять принадлежность того или иного конкретного факта к некоторому понятию. Он выделяет *знания*, отделяет их от данных, которые всегда задаются экстенсионально (перечислением)». База знаний этого типа отличается от баз знаний экспертных систем тем, что в ней объектом формализации является не опыт [120], а аксиомы предметной области, она продуцирует не заключения, а факты, выполняя роль не принятия решений, а подготовки для этой цели исходных данных.

По способу хранения вариантов знания эта база знаний может быть отнесена к *процедурному* типу [121], по виду процедурного знания может быть названа *фактуальной*, а по назначению – *вариантной*. В порождаемых ею цепочках признаки интенсионалов могут находиться в различных отношениях – родо-видовом, партитивном, собирательном и каузальном. В последнем случае базы знаний относится к продукционному типу, так как цепочки признаков совпадают с путями в дереве вывода заключений.

Обобщённая блок-схема БЗ процедурного типа изображена на рис. 7.5. Первый блок БЗ предназначен для формирований списка и значений параметров, определяющих режим порождения цепочек признаков. В нём также задаются *изменяемые* требования к свойствам признаков и цепочек в отличие от постоянных требований, фиксируемых в системе. Это позволяет считать рассматриваемую базу знаний *адаптивной* по отношению к условиям её применения [98].

Генератор признаков БЗ выполняет выбор незапрещённых признаков из списков и включает их в синтезируемую цепочку, обеспечивая последовательный перебор всех списков, и на этой основе порождение

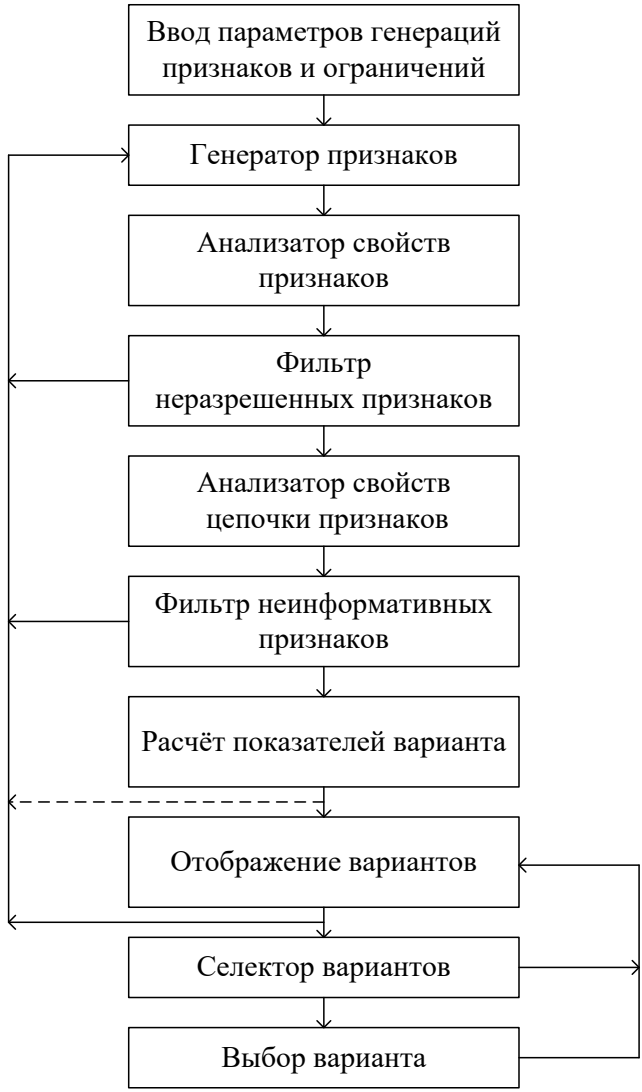

Рис. 7.5. Блок-схема базы знаний процедурного типа

всевозможных цепочек признаков. Включение признака в цепочку выполняется с использованием синтаксических правил формальной системы – модели генератора признаков. Если списки признаков не задаются извне, они формируются специальным блоком системы или вводятся пользователем в процессе порождения цепочек.

Каждый вновь выбранный признак анализируется на удовлетворение заданному требованию к определенному свойству или группе свойств. При отрицательном результате анализа признак фильтруется, т.е. не включается в синтезируемую цепочку, и заносится в список запрещённых признаков.

После индивидуального анализа признак оценивается в составе цепочки признаков. Если целевое свойство цепочки при его включении не улучшается, то он, как неинформативный, исключается из цепочки (фильтруется) и также заносится в список запрещённых признаков.

Каждая цепочка синтезируется до тех пор, пока она не удовлетворит заданному целевому свойству – назначению варианта. Семантически правильная цепочка, удовлетворяющая назначению, рассматривается как *допустимый* вариант. Он визуально отображается в процессе или по окончанию его синтеза, либо в составе группы допустимых вариантов после завершения их порождения. В этом случае база знаний выполняет роль информационно-справочной системы.

В том случае, когда к варианту предъявляются некоторые специальные требования, генерируемые варианты подвергаются селекции в процессе их порождения, либо после неё. При этом отсеиваются все варианты, не удовлетворяющие этим требованиям. Оставшиеся варианты характеризуются выбранными показателями. Если ни один из вариантов не отвечает предъявленным требованиям, последние ослабляются. В этом качестве база знаний выполняет роль информационно-советующей системы. Из оставшихся вариантов пользователь выбирает наиболее подходящий на основе неформальных предпочтений.

Как следует из изложенного, получение допустимого варианта представляет собой многоступенчатый процесс. Он управляется предъявляемыми к вариантам требованиями в широком диапазоне – от отнесения к допустимым всех генерируемых вариантов до их полной недопустимости.

Архитектурно база знаний процедурного типа представляет собой совокупность следующих подсистем (рис. 7.6):

- генерации и анализа вариантов;
- селекции вариантов;
- отображения и просмотра вариантов;
- настройки режимов.

Подсистема генерации и анализа вариантов разделяется на две соответствующие подсистемы. Из всех подсистем наибольшей независимостью от назначения базы знаний обладают подсистемы генерации и селекции вариантов. Они адаптируются к различным знаниям путем означивания соответствующих переменных. Остальные подсистемы имеют большую специализацию и разрабатываются для каждого типа знаний.

Подсистема отображения и просмотра вариантов представляет варианты в адекватной их содержанию форме – табличной, аналитической (символьной), графической (графы, блок-схемы, диаграммы, циклограммы и т.д.), а также текстами на естественном языке.

Варианты могут сохраняться временно в оперативной памяти ЭВМ на время их селекции и просмотра, а также постоянно – в соответствующих файлах.

Помимо целевой информации, отображающей варианты знания, в системе используется управляющая и справочная информация. Первая служит для настройки режимов функционирования системы, а вторая – для информирования пользователя о режимах порождения вариантов – штатных и нештатных.

7.4. Диагностическая база знаний общего назначения

Она схватывает системы понятий, диагностических моделей и методов диагностирования.

7.4.1. Система понятий

Она порождается базой знаний процедурного типа следующим образом. В качестве начального признака поочередно принимаются родовые понятия системы: *техническое диагностирование*, *дефект*, *диагностическая информация*, *диагностируемость*, *тест* и др.

Рис. 7.6. Архитектура базы знаний процедурного типа

Порождение видовых понятий Таблица 7.4

Д	К, П	С, ДН	Т, Р	ВН, ВШ
Д	Д К	Д К С	Д К С Т	Д К С Т ВН
				Д К С Т ВШ
			Д К С Р	Д К С Р ВН
				Д К С Р ВШ
		Д К ДН	Д К ДН Т	Д К ДН Т ВН
				Д К ДН Т ВШ
			Д К ДН Р	Д К ДН Р ВН
				Д К ДН Р ВШ
	Д П	Д П С	Д П С Т	Д П С Т ВН
				Д П С Т ВШ
			Д П С Р	Д П С Р ВН
				Д П С Р ВШ
		Д П ДН	Д П ДН Т	Д П ДН Т ВН
				Д П ДН Т ВШ
			Д П ДН Р	Д П ДН Р ВН
				Д П ДН Р ВШ

В качестве последующих признаков принимаются видовые отличия, сгруппированные относительно оснований деления – мета признаков понятий. Группа признаков, соответствующая *i*-му основанию деления, образует *i*-й список признаков БЗ.

Если основания делания независимы – для случая фасетной (параллельной) классификации, списки признаков распределяются относительно ярусов дерева порождения вариантов случайным образом. В противном случае над ними устанавливается отношение порядка. Порядок над списками является *линейным*, если каждому ярусу дерева сопоставляется один список признаков, или *частичным*, если – более одного. Линейный порядок может иметь место и для фасетной классификации, если списки видовых отличий сгруппированы по важности для пользователя. Пример порождения видовых понятий фасетной классификации относительно родового понятия *диагностирование* (Д) приведен в таблице 7.4.

Порядок оснований деления выбран произвольный. Соответствующие им списки видовых отличий содержат по два противопоставляемых друг другу признака. Ни один из них не относится к запрещённым понятиям. В примере дерево порождения вариантов имеет четыре яруса. Максимальное число его конечных вершин равно 16-ти. Для компактности в таблице применены следующие аббревиатуры (сокращения) видовых отличий:
К – контроль, П – поиск, С – статическое, Д – динамическое, Т – тестовое, Р – рабочее, ВН – внутреннее, ВШ – внешнее.

Каждый ярус дерева порождения понятий наращивается очередными видовыми отличиями, помещёнными в верхней строке таблицы. Понятия, подлежащие употреблению и содержащие более двух слов, обычно обозначаются специальными терминами. Принятые обозначения понятий сводятся в словари терминов и сокращений, открытые для новых обозначений. Например, понятие Д П – (диагностирование-поиск) обычно называют одним из этих признаков, т.е. диагностированием или поиском, а Д К – *контролем технического состояния*. Совокупность признаков Д К Ф называют *функциональным контролем*, а Д К Р В Н (диагностирование, контроль, рабочее, внутреннее) называют *контролем функционирования* и т.д.

Аналогичным образом порождаются межвидовые и собирательные

понятия. В этом случае ярусам дерева ставятся в соответствие видовые понятия из различных фасет – для порождения межвидовых понятий и из одной фасеты – для порождения собирательных понятий. Комбинируя списки понятий для ярусов дерева порождения вариантов, можно получать сетевые классификация предметной области.

Для установления отношений между известными понятиями база знаний дополняется подсистемой, реализующей соответствующий алгоритм, приведенный во второй главе. Существенные признаки понятий, используемые для установления отношений, извлекаются из их определений с использованием формального языка определения понятий.

Анализ различных сочетаний признаков, порождаемых базой знаний, позволяет порождать на концептуальном уровне новые методы и средства диагностирования и сопоставлять совокупностям признаков известные понятия.

7.4.2. Система диагностических моделей

Эта система, представленная в главе 4, порождается аналогично системе понятий. В качестве начального признака принимается некоторое основное свойство объекта диагностирования, например, *функция.* В качестве последующих признаков принимаются другие аспекты, подлежащие сравнению в диагностической модели ВС, начиная с общих и кончая частными признаками. Например, *управление* ЭВМ может реализовываться *схемными*, *микропрограммными* и *программными* средствами. Соответствующему ярусу дерева ставится в соответствие список этих признаков. Таким же образом, детализируются любые другие характеристики ЭВМ – архитектурные, функциональные, структурные, электрические, конструктивные и др.

Как следует из изложенного, основную рожь в порождении моделей играет формирование и упорядочение признаков, характеризующих вычислительные системы. Поэтому база знаний должна включать базу *характеристик* ВС. Входящие в неё признаки группируются в кластеры, внутри которых они упорядочиваются в отношении «общее-частное». При анализе конкретной ВС из базы характеристик выбираются подлежащие рассмотрению группы признаков и упорядочиваются относительно ярусов дерева порождения моделей.

Помимо признаков, характеризующих ВС, база данных содержит

метапризнаки, характеризующие сами модели, такие как *однородность* (гомогенные, гетерогенные модели), *соответствие задаче* (адаптированные, неадаптированные модели), *форма представления* (аналитическая, графовая, табличная) и др. Они также образуют списки сканируемых генератором признаков.

Сопоставление порождаемых моделей с известными обеспечивается с помощью базы *стандартных моделей.* Она содержит интерпретации совокупностей отражаемых моделями признаков ВС. Например, СФ-модель вентильного уровня, адаптированная к активизации путей, имеет аналитическую интерпретацию «эквивалентная нормальная форма» булевой функции. Совокупность признаков моделей может иметь более одной интерпретации, а может не иметь ни одной. Последнее означает, что порожденная модель ранее не рассматривалась.

Для диагностических моделей существенными являются включаемые в них модели неисправностей. Списки последних составляются относительно отражаемых моделью аспектов ВС. Например, для СФ-модели вентильного уровня список неисправностей включает следующие их виды: *константную, замыкание статическое* (короткое замыкание) и *динамическое* (взаимовлияние), а для Ф-модели дешифратора – не выбор ни *одного* из выходов, выбор *другого* выхода, выбор *наряду* с назначенным другого выхода и т.д. Эти списки составляются на основе анализа искажений моделируемых свойств ВС.

Для отображения свойств ВС и их искажений в базе знаний используются естественный и математический языки. Соответствие между ними устанавливается словарем *обозначений.*

Генерируемые модели находятся между собой в родовидовом, либо партитивном отношении, образуя решетки моделей (см. главу 4). На основе содержательной модели могут строиться также модели конкретных объектов путём построения отношений над её признаками.

База знаний диагностических моделей позволяет рассматривать различные варианты совокупностей различных моделируемых свойств ВС и их искажений, устанавливать связи между ними, оценивать существующие модели и предлагать новые.

7.4.3. Система методов диагностирования

Эта система, рассмотренная в главе 5, может быть построена формально с помощью базы знаний переборного типа. В качестве начального признака базового метода принимается ядро функционального базиса, а в качестве последующих признаков – входящие в него остальные функции. Функции упорядочиваются по спискам в соответствии с ограничениями на порядок их следования. На основе этих списков и порождаются различные методы диагностирования и построения тестов. Функции и ограничения размещаются в базе данных, откуда они извлекаются для порождения различных вариантов методов.

Так же, как и для моделей, база знаний методов включает базу стандартных методов. Она позволяет порождаемым совокупностям признаков сопоставлять известные методы диагностирования и построения тестов.

База знаний, порождающая базисные методы, вместе с подсистемой анализа методов реализует назначение информационно-советующей системы. Сопоставительная оценка методов на примере разбиения множества модификаций ОД рассмотрена в разделе 7.2. Реализующая её подсистема анализа является внешней по отношению к базе знаний переборного типа, поскольку она не фильтрует признаки в процессе порождения цепочек. Внутренняя система анализа признаков и вариантов рассматривается в следующей главе.

Наиболее адекватной формой отображения методов являются блок-схемы. Эту функцию реализует подсистема отображения методов.

Рассмотренные базы знаний охватывают концептуальный уровень диагностического обеспечения вычислительных систем и предназначаются для ранних этапов его разработки – формирования технического задания, разработки эскизного проекта и подготовки соответствующей документации.

Помимо порождения базисных методов база знаний процедурного типа может порождать параметрические методы с ограниченным числом признаков такие, например, как методы генерации тестов для ЗУ [122], методы диагностирования на сетевом уровне ВС. Система порождения последних подробно рассматривается в следующей главе.

Глава 8. БАЗА ЗНАНИЙ «МЕТОДЫ ДИАГНОСТИРОВАНИЯ ВЫЧИСЛИТЕЛЬНЫХ СЕТЕЙ»

8.1. Метод диагностирования вычислительной сети

Под методом диагностирования вычислительной сети будем понимать совокупность операций вычисления, пересылки и сопоставления результатов решения задачи (или её фрагмента) в вычислительной сети с целью рабочего диагностирования её вычислительных модулей (ВМ) [92]. В процессе вычислений ВМ может находиться в трех состояниях: исправном G (Good), сбоя M (Malfunction) и отказа F (Failure). Неизвестное состояние ВМ помечается знаком «?». Для постановки диагноза ВМ будем использовать оценки результатов вычислений (РВ) на ВМ: правильный *t* (true), ложный *f* (false), «сбойный» *m* (malfunction).

В случае отказа с различающимися РВ состояние ВМ помечается символом D (Dynamic), а соответствующие ему оценки РВ – *d* и *e* (при контроле повтором). Этот случай будем рассматривать как дополнительный по отношению к отказу с постоянными РВ.

Сформулируем вербально аксиомы диагностирования ВМ.

1. В течение интервала времени решения задачи (вычисления оператора) и диагностирования ВМ в задействованных для этих целей вычислительных модулях может произойти только один отказ, либо сбой.
2. Отказ ВМ проявляется двумя или более следующими подряд неверными результатами вычислений.
3. Состояние ВМ оценивается по большинству одинаковых РВ.
4. ВМ не обладает свойством самовосстановления (после двух или более неверных РВ не может быть получен правильный РВ).
5. Оценку собственного состояния и состояний вспомогательных ВМ выполняет ВМ, формирующий результат вычисления.

Выводимыми на основе перечисленных аксиом являются следующие утверждения.

Утверждение 8.1. Для обнаружения сбоя необходимо и достаточно выполнить *одно* сравнение результатов, вычисленных на одном ВМ.

Утверждение 8.2. Для обнаружения отказа необходимо и достаточно выполнить *одно* сравнение результатов, вычисленных на разных ВМ.

Утверждение 8.3. Для диагностирования сбоев и отказа основного и дублирующего ВМ необходимо выполнить *не менее четырёх* сравнений РВ.

Докажем это утверждение. Пусть неисправен один из ВМ. Согласно утверждению 8.2 для обнаружения отказа нужно сравнить его РВ и РВ другого ВМ. Но не сравнение этих РВ не позволяет указать, чей РВ неверен. Для этого согласно аксиоме 3 необходимо привлечь для арбитра РВ третий ВМ. РВ, не сравнивающийся с двумя другими, согласно аксиоме 1 считается неверным. Но при этом по одному РВ невозможно определить состояние ВМ с неверным РВ – отказ или сбой ВМ. Для различения отказа от сбоя согласно утверждению 8.1 требуется повторить вычисление на ВМ с неверным РВ. Таким образом, для диагностирования состояния отказа или сбоя ВМ необходимо вычислить три РВ на трёх разных ВМ и повторить вычисление на двух из них, выполнив не менее четырёх сравнений РВ, что и требовалось доказать.

Для диагностирования отказов с разными РВ необходимо дополнительно выполнить ещё два сравнения – для каждой тройки РВ, используемой для диагностирования основного и дублирующего ВМ (всего 6 сравнений).

Результаты вычислений делятся на *оцениваемые* (ОРВ) и *базы сравнения* (БС). Состояние ВМ оценивается по исходу сравнения ОРВ и БС. Исход сравнения согласно утверждениям 8.1 и 8.2 различен для случаев повтора и дублирования вычислений.

Выразим аналитически условия реализации изложенных способов. Обозначим через y_{ij} результат вычисления (ОРВ), выполненного на i-м модуле в j-й раз, а через y_{kl} – базу сравнения, вычисленную на k-м модуле в l-й раз.

Условием обнаружения *сбоя* ВМ является несовпадение результатов двух вычислений на *одном* (положим первом) модуле: $y_{11} \neq y_{12}$.

Условием обнаружения *отказа* ВМ является несовпадение результатов, вычисленных на *разных* модулях, и совпадение результатов двух вычислений на *одном* модуле: $(y_{11} \neq y_{21})$ & $(y_{11} \neq y_{12})$.

Условием различения отказа от сбоя является несовпадение результатов, вычисленных на разных модулях, совпадение результатов двух вычислений на одном модуле и несовпадение результатов вторых вычислений на разных модулях:

$$(y_{11} \neq y_{21}) \,\&\, (y_{11} = y_{12}) \,\&\, (y_{11} \neq y_{12}).$$

Условием нахождения (диагностики) модуля отказавшего или предотказного (выдающего сбои), является несовпадение результатов основного и контрольного модулей и несовпадение (совпадение) результатов основного и арбитражного модулей:

$$(y_{11} \neq y_{21}) \,\&\, (y_{11} \neq y_{31}) \vee (y_{11} = y_{31}).$$

Условием нахождения (диагностики) отказавшего модуля, и различения его от предотказного (выдающего сбои), является несовпадение результатов основного и контрольного модулей, несовпадение (совпадение) результатов основного и арбитражного модулей и несовпадение (совпадение) повторно вычисленных результатов с предыдущими:

$$(y_{11} \neq y_{21}) \,\&\, ((y_{11} \neq y_{31}) \vee (y_{11} = y_{31}) \,\&\, (y_{11} \neq y_{12}) \vee (y_{11} = y_{12})).$$

Представим все возможные варианты пар ОРВ-БС оцененных РВ, в табличном виде, сведя их в матрицы равенства $\mathbf{R}_{=}$ и неравенства $\mathbf{R}_{\neq}$ при повторе и $\mathbf{D}_{=}$ и $\mathbf{D}_{\neq}$ – при дублировании. В матрицах $\mathbf{R}_{=}$ и $\mathbf{R}_{\neq}$ всевозможные пары оценок «основной РВ-повторной РВ» приведены построчно:

$$\mathbf{R}_{=} = \begin{array}{|cc|} \text{ОРВ} & \text{БС} \\ t & t \\ f & f \end{array} \qquad \mathbf{R}_{\neq} = \begin{array}{|cc|} \text{ОРВ} & \text{БС} \\ t & m \\ m & t \\ t & f \end{array}$$

Из 3^2=9 возможных пар в матрицы включены только 5 пар. Четыре пары – (m, m), (m, f), (f, m), (f, t) являются недопустимыми: три – на основании аксиомы 1 (единственности отказа или сбоя), а четвертая – на основании аксиомы 4 (невозможности самовосстановления ВМ).

Матрицы $\mathbf{D}_{=}$ и $\mathbf{D}_{\neq}$, включающие варианты сравнений и не сравнений РВ при дублировании, представляют собой следующие совокупности столбцов, состоящих из пар РВ:

$$\mathbf{D}_{=} = \begin{array}{l|c|} \text{ОРВ} & t \\ \text{БС} & f \end{array} \qquad \mathbf{D}_{\neq} = \begin{array}{|cccc|} t & t & m & f \\ m & f & t & t \end{array}$$

Матрица $\mathbf{D}_{=}$ включает единственный столбец, так как пара (f,f) является недопустимой согласно аксиоме 1. В то же время, допустимой в матрице $\mathbf{D}_{\neq}$ по сравнению с матрицей $\mathbf{R}_{\neq}$ является пара (f, t) поскольку РВ вычисляется разными (а не одним) ВМ.

При учёте отказа с разными РВ матрица $\mathbf{R}_{=}$ дополняется парами (f, d) и (d, e), а матрица $\mathbf{D}_{\neq}$ – парами (f, d) и (d, f).

Матрицы $\mathbf{R}_{=}$, $\mathbf{R}_{\neq}$ и $\mathbf{D}_{=}$, $\mathbf{D}_{\neq}$ представляют собой области определения предикатов не сравнения $P_{\neq \mathrm{R}}(y_{ij}, y_{kl})$ и $P_{\neq \mathrm{D}}(y_{ij}, y_{kl})$. Последние принимают значение истина на множестве пар из $\mathbf{R}_{\neq}$ и $\mathbf{D}_{\neq}$ и ложь – на множестве пар из $\mathbf{R}_{=}$ и $\mathbf{D}_{=}$.

Задаче метода диагностирования вычислительной сети – постановке диагноза ВМ по результатам сравнения их РВ, соответствует нахождение оценок всех РВ, привлекаемых для постановки диагноза. Поскольку согласно утверждению 8.3. максимальное их число равно пяти, для постановки диагноза основного и дублирующего ВМ необходимо знать значения трёх двухместных предикатов ($]\log_2 5[= 3$). Таким образом, процедуру диагностирования можно представить трёх-ярусным дихотомическим деревом. В нём переход от верхнего яруса к нижнему осуществляется с помощью обратного отображения предиката не сравнения. Последнее однозначно только для случая $P_{\neq \mathrm{D}}^{-1}(t)=(t,t)$, поскольку матрица $\mathbf{D}_{=}$ содержит всего один столбец. Это означает, что диагноз исправности при дублировании вычислений устанавливается сразу же по положительному исходу сравнения параллельно полученных результатов. Распознавание других состояний требует большего числа сравнений.

Дерево диагностирования ВМ строится в следующей последовательности [92]:

1. В корневой вершине дерева помещается базовый результат y_{00}.
2. Выполняется ветвление вершины дерева относительно пары сравниваемых переменных. Одна из ветвей помечается знаком =, соответствующим значению $P_{\neq}(y_{ij}, y_{kl})$=л (ложь), а противоположная ей – знаком ≠.

3. Значениям предикатов $P_{\neq R}$ и $P_{\neq D}$ сопоставляются подмножества пар из вышеприведенных матриц, которые используются для формирования кортежей состояний ВМ для данного ветвления.
4. Если ветвлению соответствует всего один кортеж, ему ставится в соответствие диагноз одного из ВМ.
5. Если ветвлению соответствует более, чем один кортеж, то разбиение подмножества кортежей выполняется с помощью последующих сравнений.

Пример построения дерева диагностирования приведен на рис. 8.1. Против каждого кортежа значений сравниваемых РВ в скобках помещен диагноз состояний ВМ. Он сопровождается вопросительным знаком при числе кортежей более одного.

Структурно m-й метод диагностирования описывается ориентированным графом $G_m=(Y_m, R_m)$. Вершина $y_v \in Y$ интерпретируется как результат вычисления одного или группы операторов (программы) на одном из ВМ, участвующих в диагностировании. Все множество вершин Y упорядочено на плоскости относительно вертикальной и горизонтальной осей. Вершина y_{00} представляет собой базовый РВ, подлежащий оценке. Остальные вершины графа играют по отношению к ней роль баз сравнения [8], хотя и сами могут использоваться в качестве ОРВ. Их номера по вертикали $v=\overline{1,n_в-1}$ и по горизонтали $h=\overline{1,n_г-1}$ характеризуют соответственно модульные и временные ресурсы, привлекаемые для диагностирования ВМ. Таким образом, процесс диагностирования осуществляется над полем результатов вычислений (РВ) размерностью $n_г \times n_в$.

Исходный граф $G=(Y, R)$ является нуль-графом: $R \subset Y \times Y = \varnothing$. Его множество дуг находится в процессе синтеза m-го метода, $m=\overline{1,N}$.

Множество дуг графа G_m соответствует операторам сравнения ОРВ и ВС. Дуга (y_{ij}, y_{kl}), соединяющая вершины y_{ij} и y_{kl} графа, интерпретируется сравнением ОРВ и ВС. Граф сравнений позволяет компактно описать метод диагностирования сети. На рис. 8.2. приведен пример такого описания для метода, изображенного на рис 8.1. Назовем граф сравнений схемой метода диагностирования сети (СМД).

На рисунке 8.2 дуги графа изображены двойными линиями – символами сравнения. Цифра, помечающая дугу, означает очерёдность

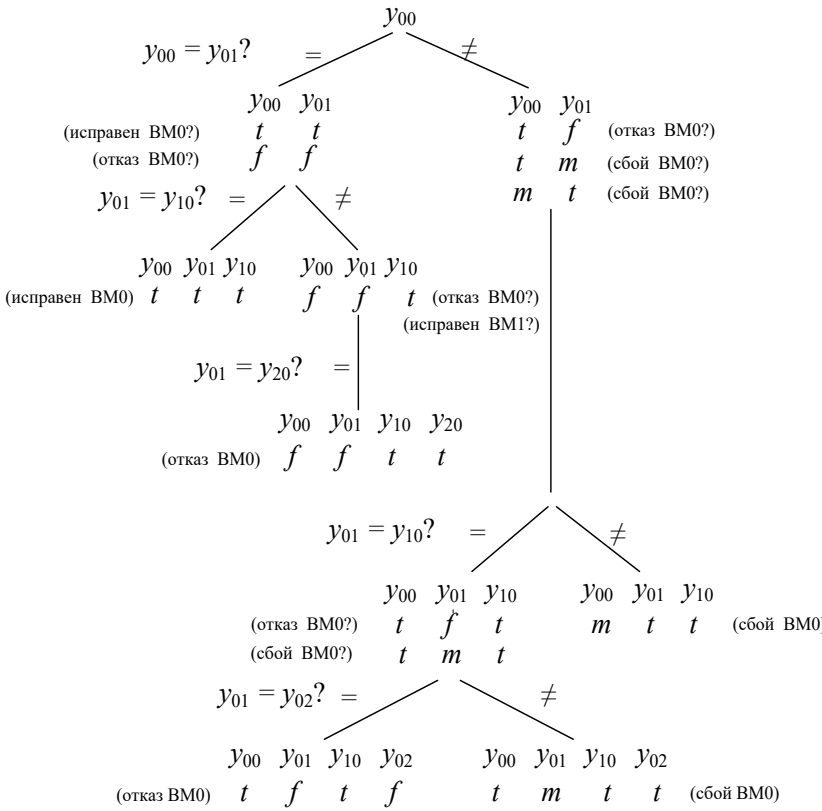

Рис. 8.1. Дерево диагностирования методом сравнения РВ

сравнения, а буква – его цель (**к** – контроль, **а** – арбитраж, **д** – диагностика). Заметим, что в строгом смысле все сравнения относятся к процедуре диагностирования, ибо диагноз состояний ВМ может устанавливаться даже по результату первого сравнения (положительный исход контроля дублированием означает исправность обоих ВМ).

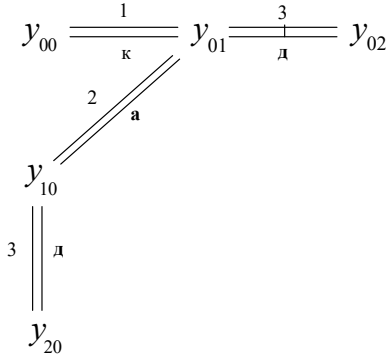

Рис. 8.2. Схема метода диагностирования сети

Наличие на сетевом уровне ВС общих средств для реализации процедур обнаружения и различения сбоев и отказов, диагностики отказавшего модуля и формирования правильного результата, позволяет синтезировать методы, обладающие различной совокупностью соответствующих свойств [123].

Принцип восстановления результата при использовании методов диагностирования сети для обеспечения отказоустойчивости ВС основан на выявлении большинства одинаковых результатов, принимаемых за истинные. Это соответствует принципу восстановления «вперед» (forward error recovery). Способу восстановления «назад» (backward error recovery) соответствует возврат к началу вычисления в случае отрицательного исхода контроля.

8.2. Нахождение разрешённых операций

Исходя из вышеизложенного, в перечень операций метода диагностирования сети входят:

- сравнение результата вычисления (РВ) задачи или её фрагмента с *повторным* РВ;

- сравнение РВ, полученного на одном вычислительном модуле (ВМ) с *дублирующим* РВ, полученном на другом ВМ.

Таким образом, признаки цепочки $\mathbf{c}_l = C_0, C_1,\ldots,C_L$, представляющей метод диагностирования сети, интерпретируется парами ОРВ-БС, в которых БС выполняют роль повторных или дублирующих РВ. Начальный признак C_0 интерпретируется основным РВ, подлежащим оценке.

Количество операций определяется требуемой *глубиной* диагностирования. В том случае, когда количество операций недостаточно для диагностирования всех участвующих в этом процессе ВМ (при ограничении числа сравнений $s<L$), сетевое диагностирование дополняется *тестовым*. С этой целью применяется запрос на тестирование ВМ, состояние которого неизвестно. При отсутствии средств рабочего диагностирования для контроля и различения отказов и сбоев применяется периодическое тестирование ВМ. Одна из этих операций завершает цепочку сравнений с фиксированной длиной $s<L$:

$$\mathbf{c}_l = C_0, C_1,\ldots,C_s, C_{s+1}.$$

Синтаксические правила формальной системы из всех возможных пар ОРВ-БС должны выделять разрешённые или *правильные*.

Правильной считается s-я пара РВ (y^s_{ij}, y^s_{kl}), удовлетворяющая условиям:

1) одного конечного результата: $s=0$, $i=0$, $j=0$;
2) несовпадения координат ОРВ и БС: $i\neq k$; $j\neq l$;
3) предшествования ОРВ базе сравнения $i<k$ или $j<l$;
4) смежности ОРВ $(\bigcup_{\sigma=1}^{s-1} y^{\sigma}_{mv}) \cap y^{s}_{ij} \neq \varnothing$;
5) новизны БС $(\bigcup_{\sigma=1}^{s-1} y^{\sigma}_{mv}) \cap y^{s}_{kl} = \varnothing$;
6) отсутствия перескоков через модуль или повтор:
   $k - m_{\max} \le 1$ и $l - v_{\max} \le 1$.

Каждая новая пара РВ проверяется на обладание перечисленными свойствами из множества $\boldsymbol{\rho}$. Если она не обладает каким – либо из этих свойств, то не включается в цепочку операций, а входящая в неё БС y_{kl} маскируется с целью запрета на дальнейшее применение.

Операция «запрос на тестирование» является разрешённой в том случае, когда состояние ВМ не распознано по результатам сетевого диагностирования.

Операция «периодическое тестирование» используется в том случае, когда метод диагностирует только сбои ВМ.

Поиск разрешённых операций сводится к решению задачи построения множеств s-достижимых вершин [124]. (s соответствует рангу ρ цепочки признаков).

Вершина y_{kl}, достижимая из y_{00} s_k шагов через вершину y_{ij}, характеризуется $s_п$-достижимостью y_{ij} из y_{00} s_k–$s_п$-достижимостью y_{kl} из y_{ij} [92]. Примеры s_k–$s_п$-достижимых из y_{11} вершин приведены на рис 8.3 (s_k–$s_п$=1, $s_п$=2, s_k=3) и на рис. 8.4 (s_k–$s_п$=2, $s_п$=2, s_k=4). Им соответствуют множества Y(3)={y_{12}, y_{20}, y_{01}} и Y(4)={y_{13}, y_{21}, y_{02}}.

Из рис. 8.3. и 8.4. следует, что s_k–$s_п$-достижимые из y_{ij}, i>0, j>0, вершины в графе данного вида находятся справа и внизу от y_{ij} (y_{01}, y_{02}). Очевидно, что движение от y_{ij} направо ограничивается числом $n_г$ вершин в горизонтальном ряду, а наверх и вниз – диапазоном от 0 до $n_в$. Кроме того, движение оказывается возможным при s_k–$s_п$>1.

На основе изложенного сформулируем следующие утверждения.

Утверждение 8.4. Координаты s_k–$s_п$-достижимой из y_{ij} вершины y_{kl} при движении вправо находятся из выражений:

$k := i$;

$l := j+s_k–s_п$ при условии, что $j+s_k–s_п < n_г$.

Утверждение 8.5. Координаты s_k–$s_п$-достижимых из y_{ij} вершин y_{kl} при движении вниз и влево находятся из выражений:

$k := i + d$ при условии, что $i + d < n_в$ и $s_k–s_п – d > 0$;

$l := s_k–s_п– d$ при условии, что $(j+s_k–s_п)/n_г \geq 1$.

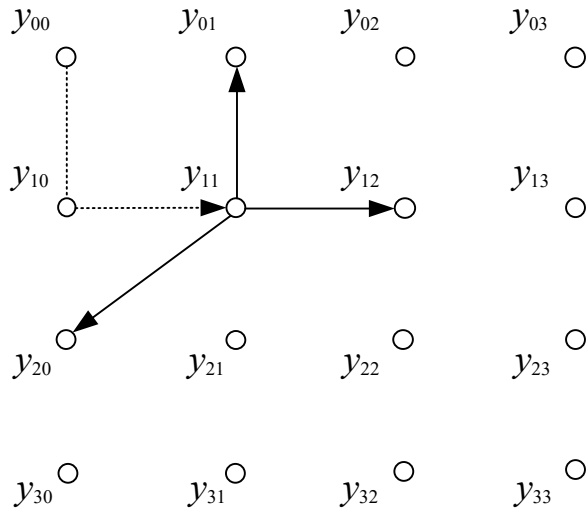

Рис. 8.3. 2-3 - достижимые вершины из y_{00} через y_{11}

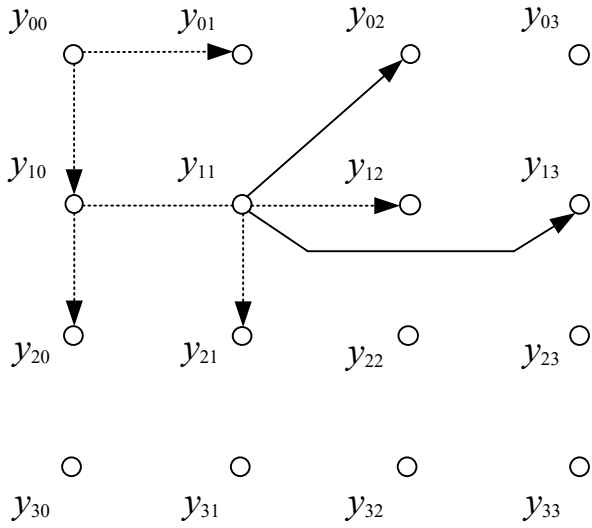

Рис. 8.4. 2-4 - достижимые вершины из y_{00} через y_{11}

Количество вершин y_{kl}, $k = i,\ldots,i + d$, при движении вниз (d=1,2,…) ограничивается условием $i + d < n_в$. Другим ограничением является $s_k–s_п – d > 0$, обуславливаемое уменьшением l вследствие сдвига влево на 1 при каждом приращении d. Выход за предел $n_г$ при $j+s_k–s_п > n_г$ ограничивается условием $(j+s_k–s_п)/n_г \geq 1$.

Утверждение 8.6. Координаты $s_k–s_п$-достижимых из y_{ij} вершин y_{kl} (d=1,2,…) при движении вверх и вправо находятся из выражений:

$k := i – d$ при условии, что $i – d \geq 0$;

$l := s_k – s_п$ при условии, что $j + s_k – s_п < n_г$.

Справедливость приведённых утверждений достаточно очевидна и легко проверяется на примерах, приведенных на рис. 8.3 и 8.4. На этих утверждениях основан следующий алгоритм построения множеств s-достижимых вершин:

1. Устанавливаются начальные данные: вершина y_{00} и s_k=1.
2. Пока $s \leq s_k$ и не назначены в качестве промежуточных вершин все вершины $y_{ij} \in \bigcup_{s_k=1}^{s} \mathrm{Y}(s_k - 1)$ для y_{ij} находится множество $\mathrm{Y}(s_k)$ при движении по графу вправо, вниз-влево, вверх-вправо;
3. $s_k := s_k + 1$;
4. Выбор следующей вершины $y_{ij} \in \bigcup_{s_k=1}^{s} \mathrm{Y}(s_k - 1)$;
5. Конец.

Оценим множества s-достижимых вершин.

Утверждение 8.7. s_k-достижимость из y_{00} вершины $y_{kl} \in \mathrm{Y}(s_k)$ равна $k+l$.

Справедливость утверждения следует из способа нумерации вершин графа ОВ и добавления по единицы к индексам вершины y_{00} при движении от неё вправо и вниз на каждом шаге процедуры.

Следствие. Максимальная достижимость в графе с $n_г$ вершинами по горизонтали и $n_в$ – по вертикали равна $n_г + n_г – 2$.

Наиболее удалённой от y_{00} является вершина $y_{nг–1,nв–1}$. Согласно утверждению 8.7 $s(y_{nг–1,nв–1}, y_{00})=(n_г–1)+(n_в–1)=n_г + n_г – 2$, что и требовалось доказать.

Приведём оценки мощностей множеств s-достижимых вершин.

Утверждение 8.8. Мощность множества вершин $\Upsilon(s_k,\ y_{ij}(s_п))$ s_k-достижимых из y_{00} через $s_п$-достижимую вершину y_{ij} максимальна при $s_k - s_п \geq n_в - 1 - i$, $s_k - s_п \geq n_г - 1 - j$ и равна $n = n_в = n_г$.

Поскольку величина $|\Upsilon(s_k,\ y_{ij}(s_п))|$ определяется суммой дуг, проходимых при движении по графу вниз-влево и вверх-вправо, она становится максимальной в тех случаях, когда это движение не ограничивается размерами графа. Это имеет место при заданных в утверждении 8.8 условиях. Первое условие характеризует возможность достижения из y_{ij} последнего $n_в - 1$-го ряда при движении по графу вниз – влево, для чего требуется $s_k - s_п$ дуг. Второе условие характеризует возможность достижения из y_{ij} начального 0-го ряда при движении по графу вверх-вправо. Условие $n_в = n_г = n$ делает движение вниз-влево и вверх-вправо взаимно дополнительными относительно координат i и j вершины y_{ij}. При выполнении первого условия утверждения 8.5 оказывается возможным движение вниз на $n_в - 1 - i$ шагов, а при выполнении второго условия – вверх на l шагов и вправо на 1 шаг. Суммируя, получаем максимальное количество шагов n, что и требовалось доказать.

Утверждение 8.9. Мощность множества вершин $\Upsilon(s_k)$ s_k-достижимых из y_{00} при условии $n_в = n_г = n$ равна:

$$|\Upsilon(s_k)| = \begin{cases} s_k + 1 & \text{при } s_k < n - 1; \\ 2 \cdot n - (s_k + 1) & \text{при } s_k > n - 1; \\ n & \text{при } s_k = n - 1; \end{cases}$$

При $s_k > n - 1$ количество элементов множества $\Upsilon(s_k)$ возрастает на единицу при каждом увеличении шага достижимости s_k на единицу вплоть до максимума n при $s_k = n - 1$ ($s_п = 1$, $i = 0$, $j = 0$), что следует из утверждения 8.8. При $s_k < n - 1$ количество элементов множества $\Upsilon(s_k)$ уменьшается на единицу в силу ограничений со стороны размерности матрицы $n \times n$. Это утверждение легко доказывается по матрице вершин $Y \times Y$.

Множествам s_k-достижимости, $s_k = \overline{0,s,}$ в ней соответствует совокупность диагоналей, перпендикулярных направлению от начальной вершины y_{00} к граничной $y_{n_г-1,n_в-1}$. При этой главной диагонали соответствует множество вершин $\mathrm{Y}(s_k)$, $s_k = n - 1$ максимальной мощности n (см. рис. 8.3 и 8.4).

Утверждение 8.10. Мощность множества всех вершин, s_k-достижимых из y_{00}, $s_k = \overline{0,s,}$ при условии $n_в = n_г = n$ равна:

$$\left| \bigcup_{s_k=0}^{s} \mathrm{Y}(s_k) = n^2 \right|.$$

Это утверждение доказывается на основе предыдущего. Аналитическое доказательство заключается в выводе из предыдущих формул следующего тождества:

$$\left| \sum_{s_k=0}^{n} (s_k + 1) + \sum_{s_k=n}^{2n-2} 2 \cdot n - (s_k + 1) = n^2 \right|.$$

Другое доказательство этой оценки основывается на анализе способа построения s_k-достижимых из y_{00} множеств вершин, $s_k = \overline{0,s,}$. Исходя из него в $\bigcup_{s_k=0}^{s} \mathrm{Y}(s_k)$ входят все вершины матрицы $Y \times Y$, начиная с начальной y_{00} и кончая граничной $y_{n_г-1,n_в-1}$, причём ни одна из них не повторяется ни разу: $\bigcap_{s_k=0}^{s} \mathrm{Y}(s_k) \neq \varnothing$. Последнее свойство основывается на том, что s_k- достижимые из y_{00} вершины находятся только при движении от неё вправо и вниз и поэтому не повторяются на каждом новом s_k-м шаге. Повторение осуществляется только при движении вверх-вправо, которое имеет место при формировании множеств вершин, s_k-достижимых из промежуточных вершин y_{ij}, i=0, j=0. Таким образом, общее количество s_k-достижимых вершин, включая исходную вершину y_{00}, s_k=0, равно размерности матрицы вершин n^2, что и требовалось доказать.

Утверждение 8.11. Множество s_k-достижимых из y_{00} через промежуточную вершину $y_{ij} \in \mathrm{Y}(s_п)$ вершин, $i>0$, $j>0$, $s_п>0$, имеет верхнюю границу $2 \cdot n^2 - 5 \cdot n + 1$.

Доказательство этого утверждения проведём для наихудшего случая. Ему соответствует построение множества $\bigcup_{s_k=s_п+1}^{s}\Upsilon(s_k)$, $s_п=1$, включающего максимальное количество подмножеств $\Upsilon(s_k)$, $s_k=\overline{2,s}$, а именно, s–2. Для них подмножество промежуточных вершин $\Upsilon(1)=\{y_{01}, y_{10}\}$. Из них через y_{10} возможно достижение граничной вершины $y_{nг-1,nв-1}$ за s шагов. Следовательно, для неё мощность подмножества $\Upsilon(y_{10}, s)=1$.

Максимальная мощность остальных подмножеств согласно утверждению 8.8 равна n. Таким образом, максимальная мощность всего множества равна $n\cdot(s - 3) + 1$. После подстановки вместо s её максимальной величины $2\cdot n - 2$ из следствия утверждения 8.7, получаем искомое выражение $2\cdot n^2 - 5\cdot n + 1$. Эта оценка является точной для случая n=3, поскольку максимальная мощность множества $\Upsilon(s_k)$, равная трём, дости-гается уже при s_k=2. При n>3 мощность множества $\Upsilon(2)$ меньше n, в силу того, что вершины последнего ряда оказываются недостижимыми из $\Upsilon(1)$.

Полученные оценки используются при назначении диапазонов переменных в программе нахождения множеств s_k-достижимых вершин. Результаты расчётов для случая $n_в = n_г = n = 4$ приведены в табл. 8.1. В её клетках вершины помечены только индексами (опущены символы y). Верхняя строка индексов вершин соответствует множеству s_k-достижимых из y_{00} вершин $\bigcup_{s_k=0}^{s}\Upsilon(s_k)$. Столбец, соответствующий каждому элементу этого множества, представляет собой множество s_k-достижимых из y_{00} через y_{ij} вершин $\bigcup_{s_k=s_п+1}^{s}\Upsilon(s_k)$, $s_п=\overline{0,s-1}$. Увеличение $s_п$ отражено в табл. 8.1 нисходящими ступенями вершин. Размерность таблицы согласуется с оценками, полученными выше.

Таблица 8.1 используется в качестве исходной информации для генерации всех графов, характеризующих методы диагностирования сети. Задача заключается в нахождении на s_k-м шаге построения графа $G_m(s_k)$ концов дуг, исходящих из вершин графа $G_m(s_k–1)$, полученного на

предыдущем шаге, $s_k = \overline{1,s}$. Шаг процедуры построения графа $G_m(s_k)$ соответствует s_k-достижимости из y_{00} вершин, претендующих на роль конца очередной дуги.

Таблица 8.1

Множества s-достижимости вершин

$s_п$	0	1		2			3				4			5	
y_{ij} / s_k	00	01	10	02	11	20	03	12	21	30	13	22	31	23	32
1	01 10														
2	02 11 20	02 10	11 01 20												
3	03 12 21 30	03 11 20	12 21 30 02	03 10	12 20 01 01	21 30 11 01									
4	13 22 31	12 21 30	13 22 31 03	11 20	13 21 30 02	22 31 12 02	10	13 20	22 30 11 01	31 21 11 01					
5	23 32	13 22 31	23 32	12 21 30	31 03	23 32 13	11 20	01 21 30	23 31 12	32 22 12	20 01	23 30 11 01	32 21 11 01		
6	33	23 32	33	13 22 31	23 32	33	12 21 30	02 22 03 03	02 32 03 03	33 23 03 03	21 30 02	31 12 02	33 22 12 02	30 11 01	33 21 11 01

Обозначим множество вершин $\Upsilon(G_m(s_k-1))$, графа $G_m(s_k-1)$ через $V(s_k-1)$. Для каждой вершины $y_{ij}\in V(s_k-1)$, $i\geq 0$, $j\geq 0$, существует ранее построенное (см. табл. 8.1) множество $W(y_{ij}, s) = \bigcup_{s_k=s_п+1}^{s} \Upsilon(s_k)$, $s_п=f(y_{ij})$.

Для нахождения концов дуг на s_k-ом шаге используется его часть, оканчивающаяся подмножеством $\Upsilon(s_k)$. Множество концов дуг $F(s)$ находится на основании следующего утверждения.

Утверждение 8.12. Концы дуг, характеризующих варианты s_k-го сравнения и имеющих начало в вершинах $V(s_k-1)$ графа $G_m(s_k-1)$, содержатся в множестве $F(s)$, вычисляемом по формуле:

$$F(s_k) = \bigcap_{y_{ij}\in V(s_k-1)} W(y_{ij}, s_k)$$

Доказательство основано на том условии, что в качестве базы сравнения на s-ом шаге могут использоваться любые варианты, входящие в множества $W(y_{ij}, s_k)$, $y_{ij}\in V(s_k-1)$, $s=\overline{0, s_k}$, при условии, что они 1-достижимы из вершин $y_{ij}\in V(s_k-1)$. Этому условию отвечают множества $W(y_{ij}, s_k)$, содержащие только по одному подмножеству $\Upsilon(s_k)$. Наличие $\Upsilon(s_k)$ среди членов рассматриваемого выражения и обеспечивает 1-достижимость вершин по отношению к последней вершине графа $G_m(s_k-1)$, полученной на последнем шаге его генерации.

На утверждения 8.12 базируется алгоритм генерации вариантов сравнения.

1. Для $s_k=\overline{0, s}$ выполняется: $G(s_k):=\varnothing$, $V(s_k):=\varnothing$, $F(s_k):=\varnothing$;
2. $s_k:=0$;
3. $V(s_k):= y_{00}$;
4. $F(s_k):= W(y_{ij}, s_k)$;
5. Выбор y_{00} из $F(s_k)$;
6. $G_m(s_k):= G(s_k) \cup (y_{00}, y_{01})$;
7. $s_k:= s_k +1$;
8. $V(s_k):= \Upsilon(G_m(s_k))$;

9. $F(s_k) = \bigcap_{y_{ij} \in V(s_k - 1)} W(y_{ij}, s_k)$;

10. Если выбраны все y_{ij} из $V(s_k)$, то идти к 14, иначе
    Выбор y_{ij} из $V(s_k)$;
    Если выбраны все y_{kl} из $F(s_k)$, то идти к 14, иначе
    Выбор y_{kl} из $F(s_k)$;
    $G_m(s_k) := G_m(s_k) \cup (y_{ij}, y_{kl})$;
11. $s_k := s_k + 1$;
12. $V(s_k) := \Upsilon(G_m(s_k))$;
13. $F(s_k) = \bigcap_{y_{ij} \in V(s_k - 1)} W(y_{ij}, s_k)$
    Если $s_k < s$, то идти к 8, иначе
    Пока не выбраны все $y_{kl} \in F(s_k)$ выполнять
    Выбор y_{kl} из $F(s_k)$;
    $G_m(s_k) := G_m(s_k) \cup (y_{ij}, y_{kl})$;
14. $s_k := s_k + 1$;
    Если $s_k > 1$, то идти к 8, иначе
    Если не выбрана $y_{01} \in F(s_k)$, то
    Выбор $y_{10} \in F(s_k)$, идти к 6, иначе конец.

На первом этапе алгоритма строится начальная дуга (y_{00}, y_{01}) графа $G_m(s)$, s=1, (п. п. 2÷6). Затем находятся остальные дуги за исключением последней (s_k=s). На основе графа $G_m(s_k-1)$ строится семейство графов $\{G_m(s)\}$ путём перебора всех концов дуг на последнем шаге. После этого осуществляется возврат на предыдущий уровень, выбор новой дуги (если не все варианты использованы) и снова переход на высший уровень с формированием множеств начал $V(s_k)$ и концов $F(s_k)$ дуг. Если использованы все варианты дуг, то осуществляется переход на предыдущий уровень. Если s_k=1 вся процедура повторяется для дуги (y_{00}, y_{10}).

Утверждение 8.13. Верхняя граница M числа частичных графов сравнений, включающих s дуг и генерируемых на основе полного графа размерностью $n \times n$ равна $M=(s+1)! \cdot (n-1)^{s-1}$.

Выражение имеет вид произведения, поскольку задача генерации вариантов носит переборный характер. Она обосновывается следующим образом. На s-м шаге процедуры граф $G_m(s)$ состоит из s дуг и $s+1$ вершин. Для каждой вершины находится множество $F(s)$. Исходя из утверждения 8.12, $|F(s)| \le |W(s_k)|$. А согласно утверждению 8.8 $|W(s_k)| \le n$. Полагая, что множества $W(y_{ij}, s_k)$ и $W(y_{rt}, s_k)$ для вершин $y_{ij}, y_{rt} \in V(s_k-1)$ различаются хотя бы на один элемент, принимаем максимальное количество концов дуг $\max|F(s)| = n-1$. При $s=1$ для одной начальной вершины y_{00} имеется всего два конца дуг – y_{01} и y_{10}. При $s>1$ $s+1$-й вершине соответствует $n-1$ концов дуг. С учетом особого случая ($s=1$) выражение для M и приобретает вид $M=(s+1)! \cdot (n-1)^{s-1}$

Таким образом, нахождение разрешённых пар ОРВ-БС осуществляется в два этапа.

На первом этапе находятся множества эквивалентных по достижимости вершин $\Upsilon(0), \Upsilon(1),\ldots,\Upsilon(s_k),\ldots,\Upsilon(s)$, где s_k – суммарное количество шагов по горизонтали (вправо) и вертикали (вниз), которое требуется для достижения вершины $y_{kl} \in V(s_k)$ из начальной вершины $y_{00} \in \Upsilon(0)$, $|\Upsilon(0)| = 1$.

На втором этапе строится совокупность из s дуг, концы которых принадлежат множествам $\Upsilon(s_п)$ и $\Upsilon(s_k)$, $s_п = \overline{0, s-1}$, $s_k = \overline{1, s}$, $s_k - s_п \ge 1$.

8.3. Анализ свойств цепочки операций

Целевым свойством цепочки сравнений, называемой методом диагностирования сети, является постановка диагнозов всех возможных состояний ВМ, участвующих в диагностировании. Каждая разрешённая пара ОРВ-БС подвергается анализу в составе цепочных сравнений на приращение числа диагнозов состояний ВМ. Пара РВ признается

неинформативной, если она не увеличивает количество диагнозов состояний ВМ. Её база сравнения при этом маскируется, т.е. заносится в список запрещенных.

Диагноз состояний ВМ ставится на основе совокупности оцененных РВ. Всевозможные сочетания оцененных РВ удобно представлять в матрицах состояний (МС) ВМ, отражающих накрытие графа G_m значениями оценок РВ $Z=\{t, f, m\}$.

Утверждение 8.14. Для моделирования различных вариантов сбоев и отказов ВМ требуется 5 исходных матриц состояний.

Число 5 соответствует количеству различных пар значений переменных в матрицах $\mathbf{R}_=$, $\mathbf{R}_\neq$ и $\mathbf{D}_=$, $\mathbf{D}_\neq$. Оно и определяет максимальное количество возможных вариантов диагностики состояния ОВ.

С учётом отказа с *разными* РВ число матриц увеличивается до 6 при контроле *повтором* и до 7 – при контроле *дублем*.

Матрицы состояний $\mathbf{M}1$-$\mathbf{M}5$ формируются при первом сравнении, выполняющем функцию контроля вычислений. Если ОРВ и БС принадлежат одному ВМ, в исходные матрицы заносятся пары оценок РВ из $\mathbf{R}_=$ и $\mathbf{R}_\neq$, а в противном случае – из $\mathbf{D}_=$ и $\mathbf{D}_\neq$. При последующих сравнениях матрицы состояний пополняются оценками баз сравнений тех пар из матриц $\mathbf{R}_=$ и $\mathbf{R}_\neq$, $\mathbf{D}_=$ и $\mathbf{D}_\neq$, для которых первая оценка совпадает с оценкой соответствующего РВ в МС.

После каждого означивания матрицы $\mathbf{M}_q^{(s)}$, $q=\overline{1,5}$, выполняется анализ входящих в неё оценок РВ на предмет постановки диагноза состояний ВМ. При положительном исходе диагностирования знак «?» заменяется значением диагноза (G, M или F).

Обозначим через $\mathbf{e}_i$ вектор оценок i-го ВМ (для каждого РВ своя оценка), а через D_i – диагноз состояния ВМ. Здесь $i\in I$, где I – множество задействованных в диагностировании ВМ. Компоненты $\mathbf{e}_i$ принимают значения из $Z=\{t, f, m\}$, а также неопределенное значение x, где $x\neq\varnothing$, если есть РВ и $x=\varnothing$, если нет РВ. $|\mathbf{e}_i|$ – число непустых оценок в векторе $\mathbf{e}_i$. Если рассматриваются несколько векторов, они разделяются символом | – «или» в нотации Бэкуса-Наура.

Определим операцию пересечения $\cap$ оценок РВ. Для двух одинаковых оценок $x\in\mathbf{e}_i$ и $x\in\mathbf{e}_j$, $i\neq j$, $x(\mathbf{e}_i)\cap x(\mathbf{e}_j)\neq\varnothing$, а для разных оценок – $x(\mathbf{e}_i)\cap x(\mathbf{e}_j)=\varnothing$.

Постановка диагнозов основывается на аксиомах 1÷5 и утверждениях 8.1÷8.3 и представляет собой систему продукций, выражающих условия «если-то». В зависимости от совокупности оценок РВ, используемых для постановки диагноза, она делится на три блока продукций:

1) оценки РВ в анализируемом ВМ – *разные*, либо *одинаковые* или одиночная оценка;
2) учёт РВ в другом ВМ – с *учётом* и *без учёта*;
3) местонахождение ОРВ – в *основном* или *дополнительном* ВМ.

В свою очередь, диагноз ВМ с учётом РВ в других ВМ *использует*, либо *не использует* оценки этих РВ. Первый вариант различается количеством используемых оценок в другом ВМ – *одной* или *более*.

Приведём продукции диагнозов ВМ, относящиеся к первому блоку – с разными оценками РВ в анализируемом ВМ.

1. Если $\mathbf{e}_i$ = <*tmt*>, то D_i:=’M’.
2. Если $\mathbf{e}_i$ = <*tm*>|<*tf*> и |I|=1, то D_i:=’?’ и требуется либо РВ другого ВМ, либо запрос теста.
3. Если $\mathbf{e}_i$ = <*tm*>|<*tf*> и |I|>1, (есть РВ в другом ВМ) и D_j:=’?’, то D_j:=’G’.
4. Если $\mathbf{e}_i$ = <*tff*> и |I|>1, то D_i:=’F’ и если D_j:=’?’, то D_j:=’G’.
5. Если $\mathbf{e}_i$ = <*tff*> и |I|=1, то требуется либо РВ другого ВМ, либо запрос теста.
6. Если $\mathbf{e}_i$ = <*mt*> и |I|>1, то D_i:=’M’ и если D_j:=’?’, то D_j:=’G’.
7. Если $\mathbf{e}_i$ = <*mt*> и |I|=1, то требуется либо РВ другого ВМ, либо запрос теста.

Продукция 1 не учитывает наличия РВ в другом ВМ. Остальные шесть продукций устанавливают диагноз ВМ с учетом РВ в другом ВМ. Однако они не используют оценки этих РВ. Отметим, что помимо диагнозов состояний ВМ заключения продукций содержат рекомендации (подсказки) по выбору последующей операции цепочки.

Приведённые продукции представлены в алгебраической записи. Более наглядной является их табличная запись. В таблице 8.2 приведены продукции, использованные для диагноза ВМ при заданных ограничениях. Первые семь продукций таблицы 8.2 являются табличными аналогами продукций, рассмотренных выше. Те из них, для которых имеет место условие $|I|=1$, занимают одну строку таблицы, а остальные – две строки. Диагностируемым состояниям ВМ соответствуют символы ‘G’, ‘F’, ‘M’ в столбце диагнозов. В двух первых столбцах таблицы приводятся рекомендуемые операции и базы сравнений для тех оценок РВ, которые не позволяют найти ни одного диагноза состояний ВМ.

Продукции 8÷18, размещённые в таблице 8.2, описывают постановку диагноза на основе одинаковых или одной оценки РВ в анализируемом ВМ. Продукция 8 является промежуточной, так как не анализирует оценок дополнительных ВМ, а для различения оценок f и m основного РВ не имеет достаточной информации. Начиная с продукции 10, для постановки диагноза используются оценки РВ дополнительных ВМ. Факт сравнения оценок РВ различных ВМ помечаются вертикальными стрелками ↓ – одной или двумя, по количеству сравнений между ВМ. Острие стрелки адресуется к ВМ с базой сравнения. Заметим, например, что наличие двух межмодульных сравнений в посылке продукции 17 позволяет поставить диагноз модулям ВМ1 и ВМ2, в отличие от продукции 8 с теми же исходными данными.

8.4. Программная реализация системы

8.4.1. Архитектура системы

Система генерации методов диагностирования сети реализована в форме интеллектуального справочника. Он состоит из трех подсистем [125] – генерации, селекции и просмотра методов. Наиболее сложной из них является подсистема генерации, теоретические основы которой изложены выше. Она включает следующие блоки:

- задания режимов и ручного ввода сравнений;
- автоматической генерации сравнений;

Таблица 8.2

Таблица диагнозов ВМ

NN п/п	ВМ	Оценки РВ	Диагноз ВМ	Требуемая для диагностики БС	
				Операция сравнения	Координаты БС
1	ВМО	*tmt*	‘M’	–	–
2	ВМО	*tm*, *tf*	–	с повт. БС	02
3	ВМО	*tm*, *tf*	–	с повт. БС	02
	ВМ1	*t* *t*	‘G’	–	–
4	ВМО	*tff*	‘F’	–	–
	ВМ1	*t*	‘G’	–	–
5	ВМО	*tff*	–	с дубл. БС	10
6	ВМО	*mt*	‘M’	–	–
	ВМ1	*t*	‘G’	–	–
7	ВМО	*mt*	–	с дубл. БС	10
8	ВМО	*f* *m*	–	с повт. БС	01
	ВМ1	*t* *t*	–	–	–
	ВМ2	*t* *t*	–	–	–
9	ВМО	*ff*	‘F’	–	–
	ВМ1	*t*	‘G’	–	–
	ВМ2	*t*	‘G’	–	–
10	ВМО	$\downarrow \begin{matrix} t \\ tx \\ tt \\ tx \end{matrix} \downarrow$	‘G’	–	–
	ВМ1			–	–
11	ВМО	$\downarrow \begin{matrix} t \\ f \end{matrix} \downarrow \begin{matrix} f \\ t \end{matrix}$		–	–
	ВМ1			с дубл. БС	20

Таблица 8.2
Продолжение

Таблица диагнозов ВМ

NN п/п	ВМ	Оценки РВ	Диагноз ВМ	Требуемая для диагностики БС	
				Операция сравнения	Координаты БС
12	ВМО	$\downarrow \begin{matrix} tx \\ mt \end{matrix} \downarrow$	‘G’	–	–
	ВМ1		‘M’	–	–
13	ВМО	$\downarrow \begin{matrix} tx \\ ff \end{matrix} \downarrow \quad \downarrow \begin{matrix} fx \\ tt \end{matrix} \downarrow$	–	с дубл. БС	20
	ВМ1		–	с дубл. БС	20
14	ВМ0	$\begin{matrix} tx \\ \downarrow\downarrow f \\ t \end{matrix} \quad \begin{matrix} tx \\ \downarrow\downarrow m \\ t \end{matrix}$	‘G’	–	–
	ВМ1		–	с повт. БС	11
	ВМ2		‘G’	–	–
15	ВМ0	$\begin{matrix} tx \\ \downarrow\downarrow ff \\ t \end{matrix}$	‘G’	–	–
	ВМ1		‘F’	–	–
	ВМ2		‘G’	–	–
16	ВМ0	$\begin{matrix} tx \\ \downarrow\downarrow mt \\ t \end{matrix}$	‘G’	–	–
	ВМ1		‘M’	–	–
	ВМ2		‘G’	–	–
17	ВМ0	$\begin{matrix} f \\ \downarrow\downarrow t \\ t \end{matrix} \quad \begin{matrix} m \\ \downarrow\downarrow t \\ t \end{matrix}$	–	с повт. БС	01
	ВМ1		‘G’	–	–
	ВМ2		‘G’	–	–
18	ВМ0	$\begin{matrix} ff \\ \downarrow\downarrow t \\ t \end{matrix}$	‘F’	–	–
	ВМ1		‘G’	–	–
	ВМ2		‘G’	–	–

- моделирования состояний ВМ;
- анализа сравнений;
- оценки свойств и ресурсов методов;

Архитектура подсистемы генерации методов диагностирования сети приведена на рис. 8.5. Рассмотрим ее подробнее.

8.4.2. Блок задания режимов и ручного ввода сравнений

В соответствии с названием он, в свою очередь, состоит из двух блоков. Первый из них предназначен для выбора и настройки всех режимов, начиная с основных – генерация, селекция или просмотр методов.

Генерация методов осуществляется вручную или автоматически. Ручная генерация осуществляется путем последовательного ввода сравнений РВ. В обоих режимах выбираются варианты синтеза метода с сетевой или тестовой диагностикой и завершение синтеза по количеству сравнений или до полной диагностики всех ВМ. Задаётся также число генерируемых методов.

При автоматической генерации методов задается пространство параметров – либо стандартное ($n_г$=$n_в$=4, s=6), либо его подпространство ($n_г$<4) $\vee$ ($n_в$<4) $\vee$ (s<6). Варьируя эти параметры, а также выбирая типы ОРВ (основной РВ или любой) и БС (повторный или дублирующий РВ), можно существенно менять количество генерируемых методов – от десятков до единиц.

8.4.3. Блок автоматической генерации сравнений.

Он, в свою очередь, состоит из двух блоков – построения множеств s-достижимостей и цепочек сравнений. Множества s-достижимости строятся сразу после установки параметров автоматической генерации сравнений с помощью алгоритма, приведенного в разделе 8.2. Результаты работы подсистемы со стандартными значениями параметров ($n_г$=$n_в$=4, s=6) сведены в табл. 8.1.

Генерация цепочек сравнений осуществляется на основе алгоритма порождения вариантов знания, приведенного в разделе 6.1. В нем каждому признаку цепочки $\mathbf{c}_l$ ставится в соответствие пара ОРВ-БС:

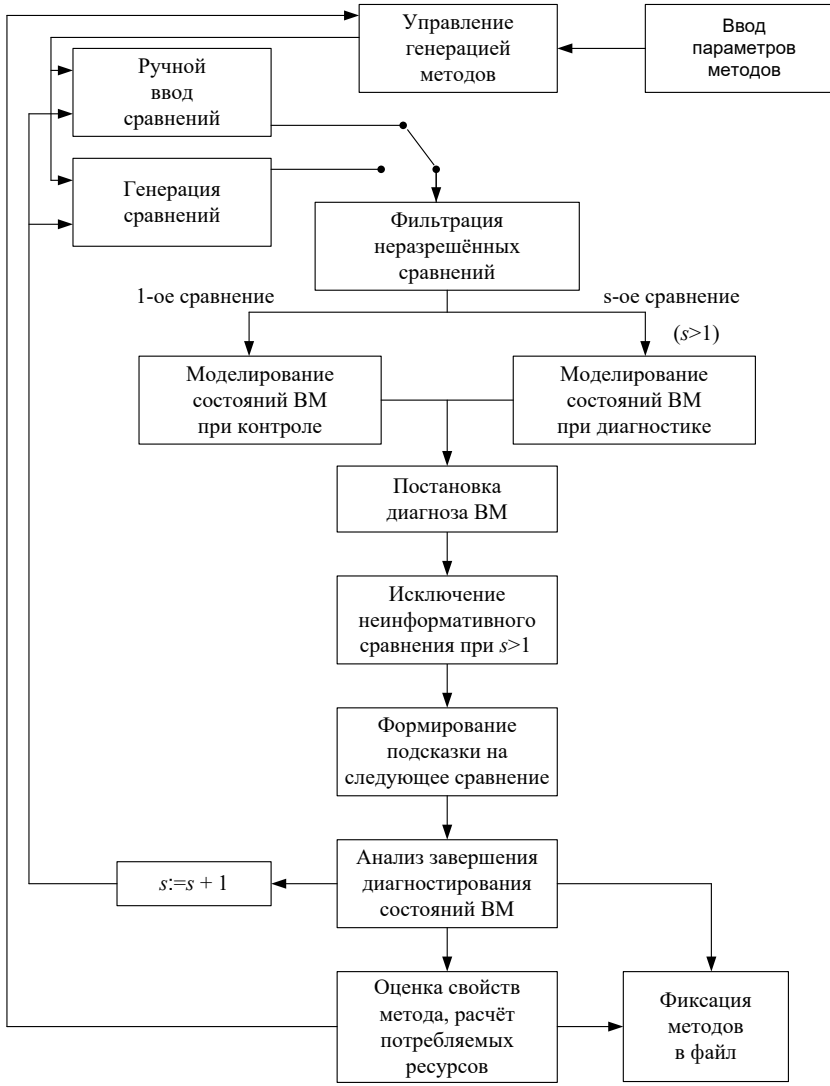

Рис. 8.5. Архитектура подсистемы генерации методов диагностирования

$C_{n\rho}=(y_{ij}, y_{kl})$. Множество C_ρ ρ-го яруса дерева генерации вариантов состоит из двух подмножеств – $V_\rho(s_k–1)$ (ОРВ или начал дуг сравнений) и $F_\rho(s_k)$ (БС или концов дуг сравнений). Оба подмножества не являются постоянными в процессе генерации цепочек сравнений кроме подмножеств $V_1(s_1–1)$ и $F_1(s_1)$ первого яруса.

Подмножества $V_\rho(s_k–1)$ и $F_\rho(s_k)$ формируются на основе множеств *s*-достижимости. Процедура формирования начинается с включения в множество $V_1(s_1–1)$ единственного элемента 00 (y_{00}) множества Y(1) (из верхней строки таблицы 8.1). Этот элемент принимается за начальный ОРВ. 1–достижимые по отношению к нему вершины 01 и 10 (см. первые две строки таблицы 8.1) включаются в множество $F_1(s_1)$. Состав последующих множеств $V_\rho(s_k–1)$ и $F_\rho(s_k)$, ρ>1, определяется *выбором* БС из множества $F_\rho(s_k)$.

При выборе в качестве БС первого элемента 01 формируется первая пара 00-01 и в множество следующего уровня $V_2(s_2–1)$ заносится выбранный элемент 01: $V_2(s_2–1)$=(00, 01). Далее, с помощью формулы утверждения 8.12 определяются элементы множества $F_2(s_2)$: ({01, 10} ∪ {02, 11, 20}) ∩ {02, 10}={02, 10}. Здесь элементы первых двух (объединяемых) множеств представляют собой 1- и 2-достижимые вершины по отношению к вершине 00 (пять первых элементов первого столбца таблицы 8.1), а элементы третьего множества – 1-достижимые вершины из выбранной вершины 01 (2 верхних элемента второго столбца таблицы 8.1). При выборе в качестве ОРВ и БС первых элементов множеств $V_2(s_2–1)$ и $F_2(s_2)$ формируется пара 00-02 и в множество следующего уровня $V_3(s_3–1)$ заносится выбранный элемент 02. Далее процесс продолжается вышеизложенным образом до тех пор, пока не будет достигнут заданный ярус *L* дерева генерации вариантов или подсистемой анализа цепочка не будет признана запрещенной.

После достижения последнего яруса дерева и формирования последней пары цепочки генерируется новая цепочка сравнений, отличающаяся только последней парой от предыдущей цепочки. С этой

целью в множестве $F_L(s_L)$ выбирается следующий элемент в качестве БС. Если все элементы этого множества выбраны и допускается в качестве ОРВ не только вершина 00, то выбирается следующий элемент множества $V_L(s_L–1)$ и для него повторяется перебор БС из множества $F_L(s_L)$. После завершения перебора множеств $V_L(s_L–1)$ и $F_L(s_L)$ и генерации $|V_L(s_L–1)| \cdot |F_L(s_L)|$ цепочек с разными последними парами ОРВ-БС, осуществляется переход на предыдущий ярус, для которого повторяется вышеизложенный процесс и так – вплоть до возврата к первому ярусу дерева. После завершения перебора цепочек с начальной парой 00-10 процесс генерации цепочек завершается.

Каждая сгенерированная пара ОРВ-БС подвергается синтаксическому контролю, условия которого сформулированы в разделе 8.2. Если она удовлетворяет этим условиям, то проверяется подсистемой анализа на приращение диагнозов. Если пара признается синтаксически неправильной, либо неинформативной, БС этой пары маскируется путем присвоения значения false соответствующему ей номеру элемента из $F_\rho(s_k)$: $B(n_{\rho БС})$:= false. В дальнейшем этот элемент в переборе не участвует. Если этот элемент содержится в множестве $F_{\rho-1}(s_k–1)$, то при переходе на предыдущий ярус на него передается и маска этого элемента.

На рис. 8.6 в качестве примера приведен фрагмент дерева генерации цепочек сравнений. Слева для каждого яруса приведены соответствующие множества $V_\rho(s_\rho–1)$ и $F_\rho(s_\rho)$, из которых выбираются пары сравнений, а также в последней колонке – запрещённые в количестве БС вершины из $F_\rho(s_\rho)$. Вершинам дерева соответствуют пары ОРВ-БС (сверху-вниз). Дерево имеет 4 яруса. Последовательности вершин от первого до четвертого яруса и представляют собой цепочки, состоящие из четырёх сравнений. В силу запрета элемента 03 на третьем ярусе первой, третьей и пятой его вершинам соответствуют неинформативные пары сравнений. Эти вершины является тупиковыми. На четвертом ярусе к запрещённым относится и элемент 11. По этой причине информативными признаются только подчеркнутые вершины. Поскольку они являются терминальными, фрагмент дерева генерации содержит 4 приемлемых цепочки сравнений:

$\mathbf{c}_1$= ((00, 01), (00, 02), (00, 10), (00, 20) ;
$\mathbf{c}_2$= ((00, 01), (00, 02), (00, 10), (01, 20));
$\mathbf{c}_3$= ((00, 01), (00, 02), (00, 10), (02, 20));
$\mathbf{c}_4$= ((00, 01), (00, 02), (00, 10), (10, 20)).

№ срв ρ	Координаты РВ		
	ОРВ $V_\rho(s_k - 1)$	БС $F_\rho(s_k)$	запр. БС
1	00	01 10	
2	00 01	02 10	
3	00 01 02	03 10	03
4	00 01 02 10	03 11 20	03 11

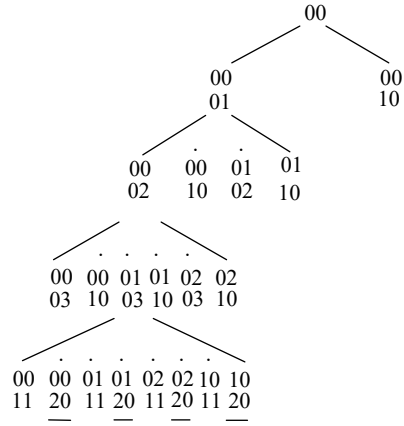

Рис. 8.6. Дерево генерации цепочек сравнений

8.4.4. Блок моделирования состояний ВМ

Он заполняет матрицы состояний на каждом шаге синтеза цепочки сравнений оценками пар ОРВ-БС. Начальное занесение оценок выполняется для первой пары ОРВ-БС, выполняющей функцию контроля. Согласно разделу 8.1 число таких пар равно пяти. Они извлекаются из матриц равенства и неравенства $\mathbf{R}_{=}$ и $\mathbf{R}_{\neq}$ – при повторе и $\mathbf{D}_{=}$ и $\mathbf{D}_{\neq}$ – при дублировании.

Для последующих пар в каждую матрицу заносится одна оценка, соответствующая БС. Условиями её занесения являются отсутствие диагнозов для *всех* строк матрицы и непустое пересечение *любой* первой оценки из матриц $\mathbf{R}_{=}$, $\mathbf{R}_{\neq}$ ($\mathbf{D}_{=}$, $\mathbf{D}_{\neq}$) с оценкой в матрице, соответствующей координатам ОРВ пары. Очевидно, что пересечение никогда не будет пустым, поскольку в матрицах $\mathbf{R}_{=}$, $\mathbf{R}_{\neq}$,$\mathbf{D}_{=}$, $\mathbf{D}_{\neq}$ первая оценка принимает все три возможных значения – *t, m, f*. Занесение оценок не требуется, если все состояния матрицы диагностированы.

Занесение новых оценок создаёт различные сочетания оценок РВ во всех пяти матрицах, что имитирует различные варианты состояний ВМ. При этом могут возникнуть сочетания оценок, неудовлетворяющие аксиомам диагностирования ВМ. К ним относятся сочетания $mt\bar{t}$ и $t\bar{f}\,\bar{f}$. Такие сочетания выявляются и заменяются на допустимые – *mtt* и *tff*.

Занесение оценок в матрицы прекращается, когда поставлены диагнозы всем имитируемым ими состояниям.

8.4.5. Блок анализа сравнений

Этот блок выполняет функции постановки диагноза ВМ и формирования рекомендаций (подсказок) по следующему сравнению. Для постановки диагноза используется система продукций, приведённая в таблице 8.2. Продукции, описывающие транзитные переходы через ВМ, не требуются, поскольку основной ВМ в матрицах состояний является *виртуальным*. Это означает, то строки каждой матрицы по очереди анализируются как основные.

В принципе система продукций может быть реализована средствами стандартных экспертных систем продукционного типа. Однако с целью экономичности и увеличения быстродействия продукции упорядочиваются в деревья диагнозов.

Блок анализа состоит из четырех секций, выделенных по количеству и результатам сравнений в одном ВМ: одно не сравнение, два не сравнения, одно сравнение, 0 сравнений (один РВ в ВМ).

Диагнозы состояний ВМ, участвующих в диагностировании, помечаются символами G, M, F, D справа от соответствующей строки МС. В том случае, когда фиксированное количество сравнений не позволяет диагностировать все состояния ВМ, соответствующие строки помечаются символом R, обозначающим необходимость тестирования ВМ с целью постановки диагноза. При сочетании оценок основного ВМ tt при s=1 и ttt при s=2 первая строка помечается символом T, обозначающим необходимость периодического тестирования основного ВМ.

На рис. 8.7 приведены распечатки матриц состояний и диагностического дерева метода, рассмотренного выше. Его дерево и схема диагностирования изображены соответственно на рис. 8.1 и 8.2.

На распечатках слева над каждой группой МС размещаются предикаты сравнения $y(ij)=y(kl)$, а над каждой матрицей – значения предикатов ‘=‘ (РВ равны) и ‘#’ (РВ не равны). Слева от каждой строки МС приводится относительные адреса ВМ, участвующих в диагностировании, начиная с нулевого – основного ВМ. Справа для каждой значащей строки МС указывается диагноз соответствующего состояния (G, M, F). Элементы МС представляют собой оценки РВ. Их количество обуславливается постановкой диагноза для всех строк матрицы. Поскольку диагнозы для всех строк матрицы определены, методы содержат по четыре сравнения для распознавания всех возможных состояний ВМ (в отсутствие отказа с разными РВ).

На рис 8.8 приведены распечатки матриц состояний и диагностического дерева метода, диагностирующего дополнительно отказ с разными РВ. Они включают дополнительно к предыдущим оценкам f и d для РВ и оценку D для состояний ВМ. Поскольку разные РВ рассматриваются только для основного ВМ, диагностирование всех его состояний требует еще одного сравнения, т.е. всего пять сравнений вместо четырех в предыдущем случае.

МЕТОД 1

Y(00) = Y(01) ?	=	=	#
Y(01) = Y(10) ?	=	#	#
Y(01) = Y(02) ?		=	#
Y(00) = Y(20) ?		#	

BM0	t t – | G	f f f | F	t m t |
BM1	t – – | G	t – – | G	t – – | G
BM2	– – – |	t – – | G	– – – |

Y(00) = Y(01) ?	#	#	
Y(01) = Y(10) ?	=	#	
Y(01) = Y(02) ?		=	
Y(00) = Y(20) ?			

BM0	m t – | M	t f f | F
BM1	t – – | G	t – – | G
BM2	– – – |	– – – |

ДИАГНОСТИКА BM0 И BM1 МЕТОДОМ 1

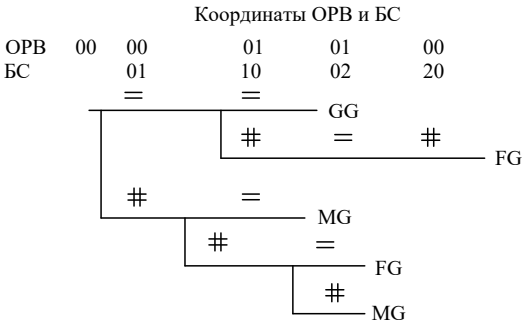

Рис. 8.7. Матрицы состояний и дерево метода диагностирования с одинаковыми РВ при отказе

МЕТОД 2

Y(00) = Y(01) ?	=	=	#
Y(01) = Y(10) ?	=	#	#
Y(00) = Y(10) ?		#	=
Y(01) = Y(02) ?		=	#
Y(00) = Y(20) ?		#	

BM0	t t – | G	f f f | F	t m t | M
BM1	t – – | G	t – – | G	t – – | G
BM2	– – – |	t – – | G	– – – |

Y(00) = Y(01) ?	#	#	#
Y(01) = Y(10) ?	=	#	#
Y(00) = Y(10) ?		=	#
Y(01) = Y(02) ?		=	
Y(00) = Y(20) ?			

BM0	m t – | M	t f f | F	f – | D
BM1	t – – | G	t – – | G	t – – | G
BM2	– – – |	– – – |	– – – |

ДИАГНОСТИКА BM0 И BM1 МЕТОДОМ 2

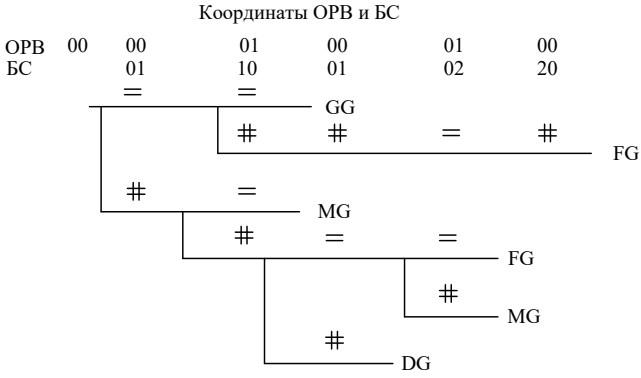

Рис. 8.7. Матрицы состояний и дерево метода диагностирования с разными РВ при отказе

8.4.6. Блок оценки свойств и ресурсов методов

К оцениваемым свойствам методов относятся:

- обнаружение сбоев ВМ;
- обнаружение отказов ВМ;
- различение сбоев и отказов.

Первые два свойства определяются относительно типа первого сравнения. Повтор вычислений позволяет обнаруживать только сбои ВМ, а дублирование – и сбои и отказы. Различение сбоев и отказов достигается с помощью либо сравнений РВ, либо вызовом теста.

Потребляемые методами ресурсы оцениваются количеством:

- ВМ, используемых для диагностирования;
- вычислений РВ;
- повторов РВ на одном ВМ;
- межмодульных пересылок;
- сравнений РВ,

а также потребностью в тестировании по запросу и периодическом тестировании. Вся эта информация при необходимости накапливается в файле.

Значения переменных, характеризующих перечисленные ресурсы определяются после синтеза каждого очередного метода. Их вычисления не представляет каких-либо затруднений. Количество пересылок N между ВМ с номерами i, j, k определяется исходя из следующих условий:

если $i=j$, то $N=0$;

если $i\neq j$, то $N=2$;

если $i\neq j$ и $k\neq l$ то $N=4$.

Все данные о методах сводятся в единую таблицу.

8.4.7. Подсистема селекции

Сгенерированные методы могут подвергаться селекции относительно способности их к обнаружению сбоев и отказов и использованию для их диагностики тестов, а также относительно ограничений на перечисленные выше ресурсы. При этом селекция осуществляется относительно четырех из семи возможных признаков. Задание режима селекции методов выполняется в два этапа – вначале задается совокупность селектирующих

признаков, а затем вводятся ограничения на признаки, характеризующие ресурсы.

Подсистема выдает номера методов, отвечающих предъявленным требованиям. При необходимости отобранные методы фиксируются в файл селектированных методов.

8.4.8. Подсистема просмотра методов

Она реализует просмотр как сгенерированных, так и селектированных методов из соответствующих файлов. Меню просмотра включает матрицы состояний и диагностическое дерево метода, таблицы обнаружения сбоев и отказов, способов диагностики и потребляемых ресурсов, причем таблицы выводятся для групп методов (по 12).

8.4.9. Характеристики интеллектуального справочника «Методы диагностирования вычислительной сети»

Интеллектуальному справочнику «Методы диагностирования вычислительной сети» присущи следующие свойства:

Построение сетевых и смешанных (с тестовыми) методов диагностирования вычислительных сетей.

Диагностирование одиночных сбоев, постоянных и перемежающихся отказов ВМ (аппаратуры и системного обеспечения).

Полнота семейства методов диагностирования вычислительных сетей.

Возможность выбора любого метода диагностирования сети в пространстве «количество ВМ – количество вычислений»: 3×3 (до 6-ти сравнений) и смешанного (с тестовым) в пространстве 3×3×2, где 2 – количество тестируемых ВМ.

Ручной и автоматический способы порождения методов.

Настройка параметров генерации методов.

Ручной режим:

Возможность пошагового построения метода с индикацией промежуточных ступней;

Переход от тестовой диагностики к сетевой;

Подсказки и выдача диагностической информации;

Обучение способам построения методов.

Автоматический режим:

Выбор оцениваемого РВ и способов контроля (повторном и дублированием вычислений);

Быстрота построения методов.

Селекция методов по одному – четырём из семи характеризующих их признаков, приведённых в меню.

Фиксация сгенерированных и селектированных методов соответствующие файлы.

Просмотр методов (матриц состояний и деревьев диагностирования) и сводных таблиц, характеризующих их свойства.

Вывод на принтер нужной информации с экрана.

8.4.10. Область применения

Проектирование подсистемы контроля и диагностики операционной системы ЭВМ сетевой архитектуры;

Планирование отказоустойчивых вычислений для ЭВМ с сетевой архитектурой и вычислительных сетей;

Исследование оптимальных методов диагностирования;

Применение при поиске отказов в вычислительных сетях;

Обучение специалистов;

Справочное пособие.

Тип ЭВМ. Персональный компьютер типа IBM PC (XT-AT).

Объём программы. 230 кбайт.

Язык программирования. TurboPascal, версия 5.0.

ЗАКЛЮЧЕНИЕ

Принятый подход к решению поставленных в монографии задач можно охарактеризовать следующими ключевыми словами: *многообразие*, *генерация* и *выбор*. Примечательно, что они согласуются с ключевыми словами эволюции Природы по Н. Моисееву: *изменчивость*, *наследственность* и *отбор*.

Многообразие характеризует сложность системы, внутренней или внешней. Примером сложной внутренней системы является машина с динамической архитектурой (МДА), характеризуемая совокупностями реализуемых уровней аппаратного и программного обеспечения. Многообразие вычислительной системы влечет многообразие применяемых для её диагностирования моделей и методов, которые играют ведущую роль в обеспечении отказоустойчивости ВС.

Второй принцип – *генерация* – характеризует возможность порождения любого варианта диагностического знания (понятия, модели, метода) на основе их элементов, принятых за начальные, и правил вывода.

Третий принцип – *выбор* – определяет целесообразность практического применения того или иного варианта диагностического знания и реализуется с применением, в общем случае, многокритериальной оптимизации. Вес критериев оценки вариантов знания меняется вместе с развитием предметной области и изменением потребностей практики.

Строгое применение изложенных принципов обеспечивает их формализация, за основу которой приняты теоретические основы порождения понятий, представляющие собой интерпретацию булевой решётки. Теоретико-множественное представление элемента знания через совокупность отражаемых им существенных признаков позволяет выразить любой элемент знания через некоторое понятие, характеризуемое своим содержанием (интенсионалом) и объёмом (экстенсионалом). Эта модель и была использована для систематизации знания предметной области.

Сопоставляя название монографии и её содержание, читатель отметит предпочтение, отданное в ней рассмотрению системного и языкового аспектов предметной области. Это сделано не случайно. Во время участия

в многочисленных научных мероприятиях у автора зачастую создавалось впечатление, что докладчики говорят почти об одном и том же, используя различные выразительные средства. Невольно зарождалась мысль о степени оригинальности излагаемого знания, его связи с известным знанием. Потребность в решении этой проблемы стимулировалась необходимостью разработки комплекса нормативно-технической документации, в котором нуждалась промышленность. Позднее возникла задача разработки эффективной информационной технологии для представления и выбора вариантов знаний при проектировании и использовании диагностического обеспечения ВС.

Вместе с тем в докладах и в литературе весьма тщательно и квалифицированно рассматривались конструктивные диагностические модели и методы диагностирования вычислительных систем. Поэтому автор ограничился кратким изложением своих оригинальных результатов в этой области, отослав читателя к многочисленным публикациям, не пытаясь дублировать их.

Насколько в монографии достигнута главная цель – доказательство полезности и эффективности применения формального подхода к систематизации предметной области – судить читателю. Но необходимо отметить следующее. Изложенное не следует воспринимать как завершённое исследование. За рамками монографии остались такие интересные и частично исследованные вопросы как применение полученных результатов к обеспечению отказоустойчивости вычислительных систем. Это, в частности, касается применения методов диагностирования сетей к проектированию подсистемы контроля ЭВМ с сетевой архитектурой и планированию отказоустойчивых вычислений на ней в зависимости от требований, предъявляемых к надежности получаемых результатов вычислений.

Не следует также воспринимать результаты систематизации как окончательные и неизменные. Развитие предметной области стимулирует появление новых противоречий, разрешение которых влечёт модификацию системы знания. Важно только, чтобы при этом сохранялись неизменными основные его фрагменты.

В практическом плане, как надеется автор, монография будет способствовать решению проблемы создания информационного обеспечения автоматизированных рабочих мест САПР ВС.

ЛИТЕРАТУРА

1. Верзаков Г.Ф., Киншт Н.В., Рабинович В.И., Тимонен Л.С. Введение в техническую диагностику. Под ред. К.Б. Карандеева. – М.: Энергия, 1968. – 224 с.
2. Клямко Э.И. Схемный и тестовый контроль автоматических цифровых вычислительных машин. – М.: Сов. радио. 1963. – 192 с.
3. Волков А.Ф., Ведешенков В.А., Зенкин В.Д. Автоматический поиск неисправностей в ЦВМ. – М.: Сов. радио, 1968., – 148 с.
4. Чжен Г., Мэннинг М., Метц Г. Диагностика отказов цифровых вычислительных систем. – М.: Мир, 1972. –232 с.
5. Основы технической диагностики. В 2-х кн.// Карибский В.В., Пархоменко П.П., Согомонян Е.С, Халчев В.Ф. Под ред. П.П. Пархоменко – М.: Энергия, 1976. кн.1. – 464 с.
6. Гуляев В.А. Организация систем диагностирования вычислительных машин. – Киев: Наукова думка, 1979. – 115 с.
7. Убар Р. Тестовая диагностика цифровых устройств, I. – Таллин: Таллинский политехнический ин-т, 1980. – 112 с.
8. Микони С.В. Систематизация методов обеспечения надёжных вычислений // Препринт № 66. Ленинградский ин-т информатики и автоматизации АН СССР, 1988. – 41 с.
9. Каляев А.В. Многопроцессорные системы с программируемой архитектурой. – М.: Радио и связь, 1984. – 240 с.
10. Avizienis A. Architecture of fault-tolerant computing systems //Fault tolerant computing symposium (FTCS) 5. Paris, 1975. – P. 3-15.
11. Авиженис А. Отказоустойчивость – свойство, обеспечивающее постоянную работоспособность цифровых систем // ТИИЭР, -1978, 66 №10. – С. 5-25.
12. Согомонян Е.С., Шагаев И.В. Аппаратурное и программное обеспечение отказоустойчивости вычислительных систем // А и Т. 1988. №2 – С. 3-39.
13. Отказоустойчивые ЭВМ за рубежом // Обзорная информация. ТС-2. Средства вычислительной техники и оргтехники. –М.: Информприбор, – Вып. 6. – 33 с.
14. Гусейнова А.А., Никитин А.И. Вычислительные системы повышенной отказоустойчивости // УС и М, 1987. №5. – С. 25-30.

15. Гуляев В.А., Додонов Л.Г., Пелехов С.П. Организация живучих вычислительных структур. – Киев: Наумова думка, 1982. – 137 с.

16. Торгашёв В.А., Плюснин В.У., Пономарёв В.М. Мультипроцессоры с динамической архитектурой // Электронно-вычислительная техника. – М.: Радио и связь, 1968. – С.173-182.

17. Торгашёв В.А. РЯД – развивающийся язык динамического типа для распределённых вычислений // Информационно-вычислительные проблемы автоматизации научных исследований. – М.: Наука, 1983. – С. 215-233.

18. Городецкий Б.Ю. Лингвистика и искусственный интеллект // Всесоюзная конференция по искусственному интеллекту. Тезисы докладов, т.1, Переславль-Залесский, –М.: ВИНИТИ. 1988. – С.13-17.

19. Лотте Д.С. Основы построения научно-технической терминологии. – М.: Изд. АН СССР. 1961. –155 с.

20. Никифоров А.Л. Научный факт и научная теория // В кн. Творческая природа научного познания. – М.: Наука, 1984, – С.150-172.

21. Целищев В.В., Карпович В.Н., Поляков И.В. Логика и язык научной теории. – Новосибирск: Наука, Сиб. отд. 1982. – 189 с.

22. Микони С.В. Построение функциональных тестов для больших схем // Электронная техника. Сер. Управление качеством и стандартизация, –1973. –Вып. 6(16). – С. 32-38.

23. Marinos P.N. Derivation of Minimal Complete Sets of Test-Input Sequences Using Boolean Differences // IEEE Transactions on Computers. 1971. vol. C-20. No 1. pp. 25-32.

24. Бессонов А.А., Стешкович Н.Т., Турчина Е.Д. Автоматизация построения контролирующих тестов. Под. ред. А.А. Бессонова, – Л.: Энергия. 1976. – 224 с.

25. Armstrong D.B. On finding a nearly minimal set of fault detection teats for combinational logic nets / D.B. Armstrong // IEEE Transactions on Electronic Computers. 1966. vol. EC-15. No 1. pp. 66-73.

26. Гольдман Р.С., Чипулис В.П. Техническая диагностика цифровых автоматов. – М.: Сов. радио. 1975. – 256 с.

27. Поспелов Д.А. Логические методы анализа и синтеза логических схем. – М.: Энергия. 1968. – 327 с.

28. Коган И.В. Построение тестов для контактных схем // Материалы научных семинаров по теоретическим и прикладным вопросам кибернетики. – Киев: РДНТП. 1963. – 46 с.

29. Данилов В.В., Карповский М.Г., Москалев Э.С. Тесты для не направленных графов // А и Т. – 1970. №4. – С.160-168.

30. Убар Р. Тестовая диагностика цифровых устройств. II. – Таллин: Таллинский политехнический ин-т. 1961. –110 с.

31. Roth, J.P. Diagnostic of automata failures: a calculus and a method / J.P. Roth // IEEE Transactions on Computers. 1966. – Vol. 10. – № 7. pp. 278-291.

32. Микони С.В. Метод построения тестов комбинационных автоматов // Автоматика и вычислительная техника. 1969. № 6, – С.24-29.

33. Микони С.В., Дубровский А.В. О контроле коротких замыканий в комбинационных автоматах // Применение ЭВМ при решении железнодорожных задач. Труды ЛИИЖТ. 1972. Вып. 335. – С.112-119.

34. Микони СВ., Дубровский А.Н. Замыкания в электронных схемах // Электронная техника. Сер. Управление качеством и стандартизация, 1976. Вып. 4(46). – С.15-23.

35. Burgess N., Damper R.I., Totton K.A.. Shaw S.J. Physical faults in MOS circuits and their coverage by different fault models // IEEE Proc. 1988. EC-135. No 1. pp. 1-9.

36. Вейцман И.Н. Диагностирование КМОП БИС // Теория, методы и средства диагностирования дискретных устройств и систем на современной элементной базе. Межвуз. н.-т. сб. ЛМИ. 1988. – С.10-12.

37. Bouicious W.G., Hsien E.P., Hoth J.P., ets. Algorithms for detection of faults in logic circuits // IEEE Trans, on Comp. 1971. No. 11. pp. 1258-1264.

38. Твердохлебов В.А. Логические эксперименты с автоматами. – Саратов: Изд-во СГУ, 1988. –183 с.

39. Тоценко В.Г. Распознавание автоматов Мура с конечной памятью и определение её глубины // В сб. Проектирование и диагностика дискретных устройств на интегральных схемах, – Киев: КВИРТУ, 1973.

40. Шаршунов С.Г. Построение тестов микропроцессоров // А и Т. 1985. № 11. – С.145-155.

41. Вийлуп А.А., Микони С.В. Построение тестов микропрограммно-управляемых устройств на основе обобщенной операторной модели // Межвуз. н.-т. сб. «Теория, методы и средства диагностирования дискретных устройств и систем на современной элементной базе. – Л.: ЛМИ, 1998. – С.15-17.

42. Thatte S.M., Abraham J.A. Test generation for microprocessors // IEEE Trans, on Comp. 1980. V. C-29. No 6. pp. 429-441.

43. Никонов Е.В., Подгурский Ю.Е. Применение сетей Петри // Зарубежная радиоэлектроника. 1966. № 11. – С. 17-37.

44. Данилов В.В. Конструктивное направление в тестовом диагностировании цифровых схем // Электронная техника. Сер. Управление качеством, стандартизация, метрология, испытания. 1983. Вып. 3(102), – С. 10-16.

45. Данилов В.В., Клюев И.Н., Петрова М.И., Тяжев В.Т. Модели и методы тестового диагностирования микропроцессорных БИС // Обзоры по электронной технике, сер. Управление качеством, стандартизация, метрология, испытания. – 1984. Вып. 2(1020). – 38 с.

46. Колмогоров А.Н., Драгалин А.Г. Введение в математическую логику, – М.: Изд-во МГУ, 1982. –120 с.

47. Трахтенброт Б.А., Бардзинь Я.М. Конечные автоматы (поведение и синтез). – М.: Наука. 1970. – 400 с.

48. Глушков В.М. Синтез цифровых автоматов. –М.: Изд-во физ.-мат. лит-ры. 1962. – 476 с.

49. Колесов Н.В. Поиск дефектов в распределённых вычислительных системах на сетевом уровне // Межвуз. н.-т. сб. «Методы и системы технической диагностики». – Саратов: Изд-во СГУ. 1990. Вып. 14. Ч.2. – С.73-74.

50. Карибский В.В., Пархоменко П.П., Согомонян Е.С. Техническая диагностика объектов контроля. – М.: Энергия, 1967. – 78 с.

51. Чегис И.А., Яблонский С.В. Логические способы контроля работы электрических схем // Труды Математ. ин-та им. В.А. Стеклова, т.51. Изд-во АН СССР. 1958. – С. 270-360.

52. Kautz W.H. Fault testing and diagnosis In combinational digital circuits // IEEE Trans. on El.. Comp. 1968. V. EC-16, No 4. pp. 352-366.

53. Данилов В.В., Тяжев В.Т. Тест поиска дефектов в комбинационной схеме // А и Т, 1981. № 8. – С. 152-158.

54. Дубровский А.В., Микони С.В. Сопоставительный анализ методов поиска дефектов в комбинационных схемах // Сб-к н. трудов «Проблемы обработки информации и интегральной автоматизации производства». – Л.: Наука, 1990. – С. 162-174.

55. Микони С.В., Поздняков Л.Н. Построение тестов и словарей для комбинационных схем методом поэлементной активизации путей, с коррекцией списков неисправностей // Электронная техника. Сер. Управление качеством и стандартизация. 1976. Вып. 4(46). – С. 15-23.

56. Микони С.В. Метод построения тестов для автоматов с памятью // Тезисы доклада 2-го Всес. совещания по теории релейных устройств и конечных автоматов. – Рига. 1971. – С.118-119.

57. Chrntal R., Gabrlel T. Microprocessor functional testing // Proc. IEEE Test Conference. Philadelfia. 1980. pp. 433-443.

53. Стёпин В.С. Идеалы в нормы в динамике научного поиска. Сб. трудов. –Минск: Изд. БГУ. 1981. –С.10-64.

59. Мальцев А.И. Алгебраические системы. – М.: Наука. 1970. – 348 с.

60. Воеводин В.В. Математические модели и методы в параллельных процессах. – М.: Наука. 1986. – 294 с.

61. Гордеев Э.Н. Задачи выбора и их решение // Сб. «Компьютер и задачи выбора. – М.: Наука. 1989. – С.5-48.

62. Статья «Понятие». Большая Советская Энциклопедия. – М.: Советская Энциклопедия. 3-е изд. т. 25. 1976. – С. 473-474.

63. Поспелов Д.А. Логико-лингвистические модели в системах управления. – М.: Энергоиздат. 1981. – 232 с.

64. Курош А.Г. Лекции по общей алгебре. – М.: Изд. физ.-мат. литры. 1962. – 396 с.

65. Фути К. Судзуки К. Языки программирования и схемотехника СБИС. – М.: Мир. 1988. – 220 с.

66. Кириллов В.И., Старченко А.Л. Логика. – М.: Высшая школа. 1982. – 262 с.

67. Шрейдер Ю.А., Шаров А.А. Системы и модели. – М.: Радио и связь. 1982. – 152 с.

68. Колмогоров А.В., Драгалин А.Г. Введение в математическую логику. – М.: Изд-во МГУ. 1982. – 118 с.

69. Вейль Г. Математическое мышление. – М.: Наука. 1989. – 400 с.

70. Смальян Р. Теория формальных систем. – М.: Наука. 1981. – 204 с.

71. Мостепаненко Н.В. Философия и методы научного познания. –Л.: Лениздат. 1972. – 272 с.

72. Котов Р.Г., Новиков А.И., Скокан Ю.П. Прикладная лингвистика и информационная технология. – М.: Наука. 1937. – 161 с.

73. Кузнецова Э.В. Лексикология русского языка. – М.: Высшая школа. 1982. – 151 с.

74. Мартынов В.Б. Категории языка. – М.: Наука. 1982. – 192 с.

75. Грамматика современного русского литературного языка. – М.: Наука. 1970. – 767 с.

76. Методика стандартизации сокращений русских слов и словосочетаний. – М.: Госкомитет стандартов СМ СССР. 1977. – 39 с.

77. Микони С.В., Чахирева А.Л. Формализованный язык для определения понятий // Научно-техническая информация. Сер.2. 1987. № 1. – С. 23-27.

78. Микони С.В. Основные принципы формализации систем понятий // Деп. рукопись. ЦНИИ Электроника. 1985. Р-3916. – 26 стр.

79. Микони С.В. Метод построения системы понятий предметной области // Тезисы докл. Всес. конференции по искусственному интеллекту. Том 1. – Переславль-Залесский: 1988. – С. 546-551.

80. Микони С.В. Генерация понятий предметной области // Сб. н.т. «Проблемы обработки знаний», – Л.: Наука. 1989. – С. 32-43.

81. Симаков Л.Л. Понятие «состояние» как философская категория. – Новосибирск: Наука, Сиб. отд. 1982. –124 с.

82. Микони СВ. Логико-семантический анализ понятия «техническое состояние» // Реферативный сб. «Научно-техническая терминология». – М.: Госкомитет стандартов СССР. 1984. Вып. 2. – С.7-12.

83. Микони С.В. Установление отношения между понятиями «техническое диагностирование» и «контроль технического состояния», сформулированных в государственных терминологических стандартах // Электронная техника. Сер. Управление качеством, стандартизация, метрология, испытания. 1984. Вып. 4. – С. 39-42.

84. Микони С.В. Систематизация понятий смежных дисциплин на основе логико-лингвистического подхода // Реферативный сб. «Научно-техническая терминология». – М.: Госкомитет стандартов СССР. 1986. Вып.1. – С. 6-11.

85. Микони С.В. Проблемы стандартизации в области диагностирования функционально-сложных изделий микроэлектроники //

Электронная техника. Сер. Управление качеством, стандартизация, метрология, испытания. 1983. Вып. 3. – С. 4-9.

86. Микони С.В. Нормативное обеспечение диагностирования и отладки микропроцессорных систем // Тезисы семинара «Средства диагностирования и отладки микропроцессорных систем». – Л.: ЛДНТП, 1984. – С. 12-15.

87. Микони С.В. Проблемы повышения познавательной функции терминологических стандартов // Научно-техническая информация. Сер.1. 1984. № 7.

88. Пятницын Б.Н. 0б активности модельного познания // Сб. «Творческая природа научного познания». – М.: Наука. 1984. – С.121-150.

89. Саркисян С.Д., Ахундов В.М., Минаев Э.О. Анализ и прогноз развития больших технических систем. – М.: Наука. 1983. – 273 с.

90. Дружинин В.В., Конторов Д.С. Проблемы системологии. – М.: Сов. радио. 1976. – 295 с.

91. Колесников Л.А. Основы теория системного подхода. – Киев: Наукова думка. 1988. – 171.

92. Микони С.В. Сетевые методы отказоустойчивых вычисления на ЭВМ с динамической архитектурой // Препринт №120. Ленингр. институт информатики и автоматизации АН СССР. 1963. – 48 с.

93. Микони С.В. Систематизация моделей, применяемых для построения тестов микропроцессорных БИС // Электронная техника. Сер. Микроэлектроника. 1966. –Вып. 3(119). – С. 62-70.

94. Микони С.В. Обоснование состава исходных данных для много-модельной системы моделирования микропрограммно-управляемых устройств // Электронная техника. Сер. Микроэлектроника. 1988. Вып. 3(127). – С. 51-55.

95. Микони С.В. О классификации моделей дискретных объектов диагностирования // Электронная техника. Сер. Управление качеством и стандартизация. 1976. Вып. 9(51). – С. 39-44

96. Микони С.В. О базовых критериях классификации моделей дискретных объектов диагностирования// Тезисы докл. 2-й Всес. конференции «Проблемы надёжности при проектировании систем управления». – Киев: РДНТП. 1976. Вып. 3. – С. 38-40.

97. Микони С.В., Дубровский А.В. О построении и применении руководящего материала «Классификация моделей цифровых

электронных схем и методов построения тестов» // Электронная техника. Сер. Управление качеством и стандартизация. 1978. Вып. 6(30). – С. 70-74.

98. Микони С.В. Процедурное представление методов решения задач в базах знаний // Тезисы семинара 5-го Ленинградского симпозиума до теории адаптивных систем. Ч.2. – Л.: ЛДНТП. 1991. – С.97-99.

99. Юдин Б.В., Микони С.В. Вопросы стандартизации в области функционального контроля БИС // Электронная техника. Сер. Управление качеством и стандартизация. 1975. Вып. 5(35). – С. 48-57.

100. Дадаев Ю.Г. Арифметические коды, исправлявшие ошибки. – М.: Сов. радио. 1969. – 167 с.

101. Блинов И.Н., Гаскаров Д.В., Мозгалевский А.В. Автоматический контроль систем управления. – Л.: Энергия. 1968. – 151 с.

102. Сидоров В.Н., Юсупов Р.М. Алгоритмическая надёжность цифровых систем управления. –Л.: ЛВИКА им. Можайского. 1969. –97 с.

103. Микони С.В., Дубровский А.В. О полноте тестов, синтезируемых методом поэлементной активизации путей // Тезисы докладов 3-го Всес. совещания по технической диагностике. – Минск. 1975. – С. 27-28.

104. Микони С.В., Поздняков Л.Н. Вопросы программирования алгоритма поэлементной активизации путей // Тезисы докл. III Всес. школы-семинара по технической диагностике. – М. 1975. – С. 28-32.

105. Микони С.В. О построении словаря неисправностей комбинационной логической схемы по её тесту // Сб. трудов ЛИИЖТ «Ж.д. автоматика, телемеханика и связь на бесконтактных элементах». 1969. Вып. 303. – С.151-162.

106. Микони С.В. Алгоритмический метод построения тестов логических схем // Сб. трудов ЛИИЖТ «Ж.д. автоматика, телемеханика и связь на бесконтактных элементах». 1969. Вып. 303. – С.173-179.

107. Микони С.В. Алгоритм построения тестов комбинационных автоматов // Сб. Техническая диагностика. –М.: Наука. 1972. – С. 179-161.

106. Микони С.В. Метод построения тестов комбинационных автоматов, синтезированных в произвольном базисе // Деп. рукопись. Автоматика и вычислительная техника, 1972. № 3. – 15 с.

109. Микони С.В., Переборов С.И. Алгоритм построения тестов для многозначных структур, синтезированных в базисе mmh // Автоматика и вычислительная техника, 1975. № 5. – С. 40-46.

110. Микони С.В. Динамический контроль комбинационных БИС // Мат. н.-т. конф-ции «Радиоизмерения», т. 2, –Каунас: 1973. – С. 10-14.

111. Микони С.В. Метод контроля автоматов с памятью // Сб. трудов ЛИИЖТ «Применение ЭВМ при решении ж.д. задач». 1972. Вып. 335. – С. 103-111.

112. Микони С.В. Метод построения тестов для больших интегральных схем с памятью // Электронная техника. Сер. Управление качеством и стандартизация. 1974. Вып. 12(30). – С. 15-28.

113. Ясинявичене Г.М., Бургис Б.В. Мецаев Е.А., Грейлинас И.А. Тестовый контроль микропроцессорных БИС на производстве. – М.: Радио и связь. 1989. – 120 с.

114. Микони С.В. Построение тестов некоторых специализированных однородных структур // Сб. трудов ЛИИЖТ «Вычислительная техника на ж. д. транспорте. 1971. Вып. 331. – С.140-145.

115. Микони С.В. Диагностика неисправностей двухмерной однородной структуры // Материалы семинара «Автоматизация и алгоритмизация проектирования цифровых устройств и систем». – Л.: ЛДНТП. 1971. – С. 3-6.

116. Микони С.В., Орурк Е.И. Тестовый контроль одной однородной структуры // Сб. трудов ЛИИЖТ «Вычислительная техника на ж.д. транспорте, 1971. Вып. 331. – С. 158-164.

117. Микони С.В. Базы знаний, генерирующие варианты знания для решения задач проектирования ЭВМ // Тезисы междунар. конференции «Автоматизированное проектирование радиоэлектронной аппаратуры». – Каунас: 1991. – С. 151-155.

118. Ермилов В.А. Фокусирование поиска терминальной базы данных среди порождённых баз. Автоматика и телемеханика. 1988. № 12. – С. 137-142.

119. Chang H. An algorithm for selecting an optimum set of diagnostic tests // IEEE Trans. on Comput. 1966. Vol. 15, No 1. pp. 3-29.

120. Городецкий В.И. Формирование понятийной структуры знаний на основе экспертной и экспериментальной информации // Тезисы докл. Всес. конференции по искусственному интеллекту. Том 1. – Переславль-Залесский. 1988. – С. 385-390.

121. Ивахненко А.Г., Ивахненко Л.Н. Индуктивные (переборные) системы искусственного интеллекта // Всес. конференции по искусственному интеллекту. Том 2. – Переславль-Залесский: 1988. – С.161-165.

122. Георгиев Н.В., Орлов Б.В. Функциональный контроль полупроводниковых ЗУ // Электронная промышленность. 1980. № 6. – С. 35-40.

123. Микони С.В. Сетевые методы отказоустойчивых вычислений // Материалы семинара «Надежность ЭВМ (аппаратуры и программного обеспечения), вычислительных сетей в процессе их разработки и эксплуатации». – Л.: ЛДНТП. – С. 51-52.

124. Евстигнеев В.А. Применение теории графов в программировании. – М.: Наука. 1985. –351 с.

125. Микони С.В., Платонов П.В. Автоматизация синтеза и анализа сетевых методов отказоустойчивых вычислений // Тезисы доклада на 15-ой Всес. школе-семинаре по вычислительным сетям. Ч.2. – Л. 1990. – С. 30-85.

126. ГОСТ 19919-74. Контроль автоматизированный технического состояния изделий авиационной техники. Термины и определения.

127. ГОСТ 20911-75. Техническая диагностика. Основные термины и определения.

128. ГОСТ 27002-83. Надёжность в технике. Термины и определения.

129. ГОСТ 15467-79. Управление качеством продукции. Основные понятия. Термины и определения.

130. ГОСТ 16504-81. Испытания и контроль качества продукции. Основные термины и определения.

131. ГОСТ 20911-89. Техническая диагностика. Основные термины и определения.

132. РД-50 14-83. Методические указания. Методика разработки стандартов на термины и определения.

133. ОСТ 11 305. 009-84. Микропроцессорные средства вычислительной техники. Контроль технического состояния и поиск дефекта. Термины и определения.

134. РМ 11 070.051-77. Классификация моделей цифровых интегральных схем и методов построения тестов.

ОГЛАВЛЕНИЕ

TABLE OF CONTENTS

Технический редактор Орехов Д.И.
Объём – 10 уч.-изд. л. Тираж 500 экз.
Типография ВМИУ
СПИИРАН. 199178, С-Петербург, 14 линия, д. 39.