АНАЛИЗ ТЕМПОРАЛЬНЫХ ДЕРЕВЬЕВ РЕШЕНИЙ С ПРИМЕНЕНИЕМ БУЛЕВЫХ ПРОИЗВОДНЫХ

УДК 519.873

КРИВУЛЯ Геннадий Федорович
д.т.н., профессор ХНУРЭ.
Научные интересы: автоматизация проектирования и диагностирование цифровых устройств.

ВЛАСОВ Илья Валерьевич
аспирант ХНУРЭ.
Научные интересы: автоматизация проектирования и диагностирование цифровых устройств.

ПАВЛОВ Олег Анатольевич
аспирант ХНУРЭ.
Научные интересы: автоматизация проектирования и диагностирование цифровых устройств.

ВВЕДЕНИЕ

В настоящее время интеллектуальные системы, использующие знания экспертов, стали неотъемлемыми компонентами современных автоматизированных диагностических систем для сложных объектов различного назначения – от медицинских до атомных электростанций. Эффективность таких систем, трудоемкость их проектирования, эксплуатации и развития, их устойчивость к изменению предметной области зависят от средств, использованных для представления знаний и методов обработки этих знаний. Основные трудности при проектировании интеллектуальных диагностических систем связаны с тем, что такие системы разрабатываются для плохо формализованных предметных областей, в которых знания неточны, неполны, противоречивы и изменчивы. Это делает необходимым разработку эффективных методов представления и обработки таких знаний. В частности, возникает необходимость пополнения, обобщения и классификации диагностической информации.

В качестве основных моделей представления знаний в интеллектуальных системах используются системы продукций, фреймовые структуры, семантические сети и логические системы.. При этом наиболее удобной формой представления знаний для компьютерной обработки являются логические системы в виде деревьев решений [1]. Прогресс в методах сбора, хранения и обработки данных позволил пользователям собирать огромные массивы данных, которые необходимо анализировать для диагностирования сложных объектов. Объемы этих данных настолько велики, что возможности их оперативной обработки экспертами – диагностами недостаточно эффективны, в результате чего возникла необходимость в методах автоматизации анализа и обработки диагностической информации.

Основные трудности при проектировании интеллектуальных диагностических систем связаны с тем, что такие системы разрабатываются для плохо формализованных предметных областей, в которых знания неточны, неполны, противоречивы и изменчивы. Это делает необходимым разработку эффективных методов представления и обработки таких знаний. В частности, возникает необходимость пополнения, обобщения и классификация диагностической информации. При этом наиболее удобной формой представления знаний для компьютерной обработки являются логические системы в виде деревьев решений [1].

Деревья решений (Decision Trees) являются хорошим инструментом в системах поддержки принятия

решений интеллектуального анализа данных (Data Mining) и являются одним из наиболее популярных средств для решению задач классификации в диагностических системах Они создают иерархическую структуру классифицирующих правил типа "если … то …" (If -Then), имеющую вид дерева. Конечными узлами дерева являются "листья", соответствующие найденным решениям и объединяющие некоторое количество объектов классифицируемой выборки.

ТЕМПОРАЛЬНЫЕ ДЕРЕВЬЯ РЕШЕНИЙ

Деревья решений в их классическом виде имеют некоторые ограничения – в частности, отсутствие учета поведения объекта или системы во времени, невозможность принятия решений с течением времени. Без учета фактора времени не удается проследить динамику изменения состояния системы. Поэтому актуальной задачей является расширение признакового описание объектов с использованием понятия «время», как один из атрибутов для построения дерева решений [2].

Для анализа поведения и диагностирования сложных технических объектов обычно используются сенсоры, от которых в определенные промежутки времени поступает информация о состоянии объекта. Оператор в режиме реального времени на основе текущих показаний сенсоров определяет вид неисправности и формирует управляющее действие для устранения возникшей неисправности. Исходными данными для принятия решения является база данных, в которой записаны модели возможных реальных неисправностей объекта и управляющие воздействия для их локализации или компенсации. Информация в базе данных формируется на основе анализа показания сенсоров, полученных от объекта в течение некоторого интервала времени наблюдения при наличии возможных реальных неисправностей. При этом возникает задача интеллектуального анализа данных, суть которой заключается в извлечении знаний из базы данных (Knowledge Discovery in Databases). Большой объем базы данных и наличие информации о поведении объектов во временной координате значительно усложняет задачу диагностирования, поэтому возникает необходимость структуризации диагностической информации в базах данных.

Одним из возможных методов решения поставленной задачи является использование темпоральных деревьев решений, которые в отличие от обычных деревьев решений содержат дополнительную информацию о времени срабатывания соответствующих сенсоров для диагностируемых неисправностей объекта. Применение темпоральных деревьев позволяют в значительной степени ускорить принятие решений в условиях, когда время является критическим фактором для принятия решений. Для построения исходного темпорального дерева решений обычно используется таблица с исходными данными всех атрибутов следующего вида:

Таблица .1

Таблица атрибутов для построения исходного темпорального дерева

Неисправность	Сенсор 1			Сенсор 2			Управляющее воздействие	Время принятия решения
	t0	t1	t2	t0	t1	t2		
N1	L	L		L	N		УД 1	T1
N 2	L	N	N	N	H	H	УД 2	T2

В таблице 1 для двух сенсоров и для неисправностей N1 и N2 приведены значения показаний сенсоров в виде нечетких значений (N-норма, L-ниже нормы, H-выше нормы) в моменты времени t0 , t1 , t2 . Для каждой неисправности вырабатывается управляющее воздействие УД1 или УД2, и вводятся временные ограничения T1 и T2 для принятия решения.

ПОСТАНОВКА ЗАДАЧИ

Процедуры построения и использования темпоральных деревьев решений для диагностирования сложных объектов имеют значительные трудности, связанные с большой размерностью деревьев. Необходимость принятия решений в реальном времени приводит к тому, что число деревьев, работающих с поступающими данными, должно быть равно числу отсчётов. Время принятия решения в общем случае будет различным для каждой ситуации. Поэтому при построении дерева накладывается ограничение на уменьшения временных меток при обходе дерева от корня к концевой вершине. В связи с этим важной задачей является минимизация темпоральных деревь-

ев решений с возможностью анализа их поведения во временном пространстве.

ПРИМЕНЕНИЕ БУЛЕВЫХ ПРОИЗВОДНЫХ ДЛЯ АНАЛИЗА ТЕМПОРАЛЬНЫХ ДЕРЕВЬЕВ РЕШЕНИЙ

Для анализа и выявления противоречий в логических продукциях на основе дерева решений целесообразно использовать булевы производные. Если система продукций имеет правила, которые вызывают противоречивые выводы, возможно появление конкурентных гипотез и заключительные рекомендации системы будут зависеть от стратегии выбора правил (от способа разрешения коллизии). В результате система может сделать не те выводы, которые ожидал бы от нее эксперт, или вообще не сформулировать решение. Последнее наиболее вероятно в том случае, когда кроме противоречий в базе знаний имеется также и неполнота. К категории противоречий можно отнести ошибки зацикливания, когда условие и заключение одного правила меняются местами в другом правиле. В логических продукционных системах основным видом противоречий между двумя правилами является случай, когда при одинаковых ситуациях, описываемых одним и тем же словом состояния, делаются противоречивые выводы. Для данного вида продукций логический вывод, представленный двоичной функцией n-входных переменных, поиск противоречивых правил может быть сведен к поиску несущественной (фиктивной) двоичной переменной. Изменения значений такой переменной из 1 в 0 или 0 в 1 не вызывают изменения значения исходной функции, т.е. булева производная функции по переменной равна нулю. Поскольку сложные продукционные системы имеют значительное количество условий для каждого правила, есть большое число входных переменных, то возникает необходимость в методах автоматизированного вычисления булевых производных.

Определение 1. *Булевой разностью функции* $f(x_1, x_2, ..., x_n)$ *называется булева функция, имеющая вид*

$$\frac{\partial f(X)}{\partial x_i} = f(x_1, x_2, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n) \oplus$$
$$\oplus f(x_1, x_2, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n),$$

или

$$\frac{\partial f(X)}{\partial x_i} = f(x_1, x_2, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \oplus$$
$$\oplus f(x_1, x_2, ..., x_{i-1}, 1, x_{i+1}, ..., x_n),$$

где $\oplus$ — сумма по модулю 2.

Существующие методы получения булевых производных основаны на известных методах минимизации переключательных функций. Например, для небольшого числа аргументов пользуются картами Карно. Поскольку при автоматическом получении булевой разности *n* аргументов возникает необходимость в многократном решении задачи минимизации переключательной функции, то требуется применение эффективных с точки зрения машинной реализации алгоритмов минимизации булевых функций. Одним из таких алгоритмов является алгоритм получения минимальной скобочной формы с использованием БДР в виде регулярных граф-схем.

Далее описан процесс автоматического получения булевых разностей переключательной функции *n* аргументов с использованием минимальных скобочных форм. Запись переключательной функции в скобочной форме с использованием граф-схем основана на принципе функциональной декомпозиции функций $f(x_1, ..., x_n)$. При минимизации в классическом базисе И-ИЛИ-НЕ функциональная декомпозиция представляет собой разложение функции по 1-му аргументу в следующей форме:

$$f(X) = \overline{x_i} f(x_1, x_2, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \vee$$
$$\vee x_i f(x_i, x_2, ..., x_{i-1}, 1, x_{i+1}, ..., x_n) = \overline{x_i} f^{\overline{x_i}} \vee x_i f^{x_i}.$$

Выполнив операцию сложения по модулю два между значениями функции на тех наборах, где аргумент x_i принимает соответственно значения 0 и 1, получим значение булевой разности, функции $f(x_1, x_2, ..., x_n)$ по переменной x_i:

$$(1 f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \vee 0 f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n)) \oplus$$
$$\oplus (0 f(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n) \vee$$
$$1 f(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_n)) = \frac{\partial f(X)}{\partial x_i}.$$

Исходя из этого, представление функции в виде граф-схем можно использовать для получения булевых производных. Скобочные формы переключательной функции имеют различную сложность в зависимости от расположения переменных в ярусах графа. В общем

случае количество различных граф-схем для булевой функции *n* аргументов равно *n*! и для числа переменных $n > 6$ простой перебор при выборе оптимального графа весьма затруднителен.

Для целенаправленного перебора переменных введем количественную оценку оптимальности граф-схем с помощью понятия их сходности [3].

Определение 2. *Сходностью граф-схемы переключательной функции назовем следующую сумму*: $S = \sum_{i=1}^{n} k_i 2^{i-1}$, где k_i — количество сходных вершин в *i*-ом ярусе:

$k_i = \sum_{j=1}^{2^{n-i}} z_j$, где $z_i = \begin{cases} 1, f^{\bar{x}_i} = f^{x_i}; \\ 0, f^{\bar{x}_i} \neq f^{x_i}. \end{cases}$ k_i также называют сходностью в ярусе, а z_j –сходностью вершины графа.

Для получения булевой производной по переменной x_i необходимо выполнить суммирование по модулю 2 функций $f^{\bar{x}_i}$ и f^{x_i}. Удобнее всего такое суммирование выполнять в том случае, если переменная x_i стоит в нижнем ярусе графа, а функции $f^{\bar{x}_i}$ и f^{x_i} равны 0 или 1. Поэтому при вычислении булевой производной по x_i данную переменную размещаем в нижнем ярусе графа, а остальные переменные расставляем в ярусах графа таким образом, чтобы обеспечить максимальную сходность граф-схемы для функции *n*-1 аргументов. Затем записывают скобочную форму булевой производной функции $f(x_1, x_2, ..., x_n)$ по переменной x_i.

Рассмотрим алгоритм получения булевых производных на основе минимальной скобочной формы переключательной функции. Булевы производные по всем *n* переменным переключательной функции строятся на основе ее закона функционирования и могут быть использованы для любой схемной реализации.

Допустим, задана переключательная функция в виде таблицы истинности:

Таблица 2.

Таблица истинности переключательной функции

x_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
x_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
x_3	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
x_4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
f	1	1	0	1	0	0	0	1	1	0	0	0	0	1	0	1

Алгоритм получения булевых производных представлен ниже.

1. Помещаем в нижний ярус графа первую по порядку переменную булевой функции. Расположение остальных переменных в ярусах графа определяется в зависимости от сходности *S* граф-схемы оставшихся *(n*-1) аргументов. Допустим, в нижний ярус графа помещается переменная x_1.

2. Выполняем попарно операцию $\oplus$ между четными и нечетными значениями булевой функции. В результате получим новую булеву функцию от *(n*-1) аргументов. Проиллюстрируем данный этап алгоритма с использованием диаграмм Венна. Характерная особенность ее в том, что наборы располагаются последовательно по возрастанию номера набора. Диаграмма Венна для булевой функции, представленной в таблице истинности таблицы 2, представлена ниже.

Таблица 3.

Таблица истинности для диаграммы Венна

x_1x_2	x_3x_4			
	00	01	10	11
00	1	1	0	1
01	0	0	0	1
10	1	0	0	0
11	0	1	0	1

Ячейки диаграммы Венна, двоичные номера которых отличаются в одном разряде, соответствующему переменной x_i, будем называть взаимно дополнительными. Для определения сходности в ярусе графа k_i сравниваются взаимно дополнительные ячейки. Для переменной x_1 взаимно дополнительные ячейки образованы первой и третьей, второй и четвертой строками; для x_2 — первой и второй, третьей и четвертой строками; для x_3 — первой и третьей, а также вторым и четвертым столбцами для переменной x_4 — первой и второй, а также третьим и четвертым столбцами.

Операция $\oplus$ при помещении в нижний ярус переменной x_i, тоже выполняется для взаимно дополнительных ячеек по переменной x_i. В результате выполнения п. 2 алгоритма получим диаграмму Венна для функции *n* - 1 аргументов (таблица 4).

Таблица 4.

Таблица истинности для диаграммы Венна

x_2	x_3x_4			
	00	01	10	11
0	0	1	0	1
1	0	1	0	0

3. Определяем номер аргумента для следующего по порядку яруса графа. Подсчитываем сходность во втором ярусе графа, если туда поставить поочередно все аргументы, кроме x_i, который уже зафиксирован в первом ярусе. Размещаем в данном ярусе тот аргумент, для которого сходность максимальна. Если таких несколько, то выбираем любой аргумент.

Для нашего примера $k_2(x_2)=3; k_2(x_3)=3; k_2(x_4)=3$.

Помещаем во второй ярус переменную x_2.

4. Определяем номер аргумента для следующего яруса графа. Подсчитываем сходность графа для данного яруса с учетом того, что в предыдущих ярусах зафиксированы определенные аргументы. Сходность вычисляем для всех случаев, когда в этот ярус ставятся незафиксированные на предыдущих ярусах аргументы. Ставим сюда аргумент с максимальной сходностью.

Пункт 4 выполняется, пока все переменные в ярусах графа не расставлены.

Каждой входной функции следующего яруса графа соответствуют две взаимно дополнительных ячейки из диаграммы Венна для предыдущего яруса графа. В рассматриваемом примере диаграмма Венна для третьего яруса графа будет иметь следующий вид (табл. 5.).

Таблица 5.

Таблица истинности для диаграммы Венна

x_3x_4			
00	01	10	11
00	11	00	10

Из табл. 5 получим $k_3(x_3)=1$; $k_3(x_4)=0$. Помещаем в третий ярус x_3, а в четвертый — x_4. Расстановка переменных по ярусам графа закончена.

5. Записываем скобочную форму булевой производной как функцию $(n-1)$ аргументов. В результате получим $\frac{\partial f(x)}{\partial x_1}=x_4(x_2\vee x_3)$. .Данному примеру соответствует граф-схема (рис. 1).

6. Пункты 2—5 данного алгоритма повторяем n раз, причем каждый раз в нижнем ярусе графа размещаем следующую по порядку переменную данной булевой функции. После выполнения n шагов все булевы производные будут найдены. Для данного примера булевы производные имеют следующий вид:

$$\frac{\partial f(x)}{\partial x_2}=\bar{x}_3\vee x_1x_4;\qquad \frac{\partial f(x)}{\partial x_3}=\bar{x}_2\bar{x}_4\vee x_1x_2x_4;$$

$$\frac{\partial f(x)}{\partial x_4}=\bar{x}_1x_3\vee x_1(x_2\vee\bar{x}_4).$$

$$\frac{\partial \mathrm{f(x)}}{\partial \mathrm{x_4}}=\bar{\mathrm{x}}_1\mathrm{x}_3\vee \mathrm{x}_1(\mathrm{x}_2\vee\bar{\mathrm{x}}_4).$$

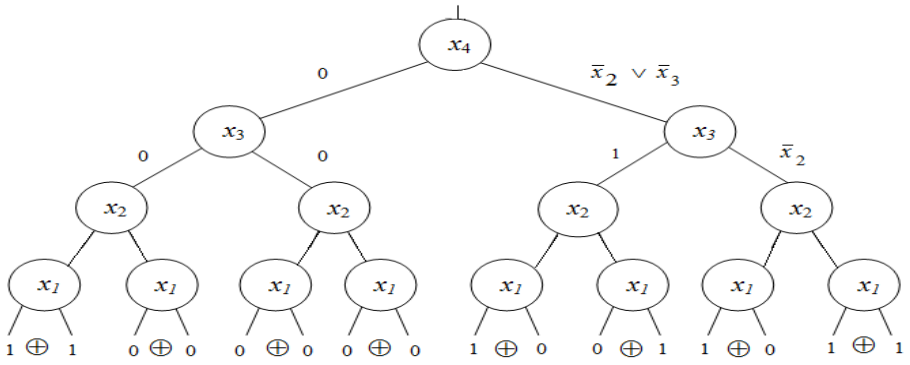

Рисунок 1– Граф-схема для производной

В тех случаях, когда требуется более детальное изучение взаимосвязи атрибутов дерева решений , возникает необходимость определения булевых производных высших порядков (кратные булевы производные).

Определение 3. *Кратной булевой производной функции* $f(X)$ *для* X_1, *где* $X_1=(x_1,x_2,\ldots,x_p)$ — *подмножество* $X=(x_1,\ldots,x_i,\ldots,x_p,\ldots,x_n)$, *т. е.* $X_1\in X$, *называется функция*

$$\frac{\partial f(X)}{\partial X_1} = \frac{\partial f(X)}{\partial x_1 \partial x_2 \ldots \partial x_p} = \frac{\partial}{\partial x_1}\left(\frac{\partial}{\partial x_2}\left(\ldots\left(\frac{\partial f(X)}{\partial x_p}\right)\ldots\right)\right).$$

В данном случае рассматривается булева производная *p*-го порядка. При вычислении кратных булевых производных алгоритм, изложенный выше, сохраняет свою работоспособность. Только в п. 1 алгоритма фиксируется в ярусах графа не одна переменная в нижнем ярусе, а *p*-переменных в нижних *p* ярусах графа, где *p* — порядок вычисляемой булевой производной. А в п. 2 алгоритма операция $\oplus$ выполняется не только для значений функции нижнего яруса графа, а для значений функции в *p* нижних ярусах графа.

На рис. 2 дан пример вычисления булевой производной второго порядка для функции, представленной в табл. 3:

$$\frac{\partial^2 f(X)}{\partial x_3 \partial x_4} = \overline{X}_1 \vee \overline{X}_2$$

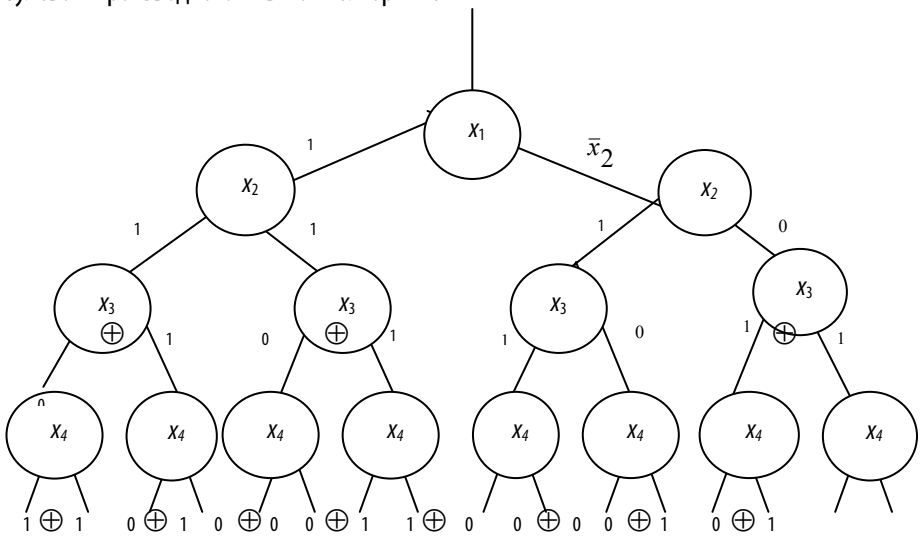

Рисунок 2 - Граф-схема для кратной производной

ВЫВОДЫ

Применение булевых производных для анализа темпоральных деревьев решений при оперативном функциональном диагностировании сложных технических объектов позволяет уменьшить объем обрабатываемой информации за счет минимизации продукционных правил в диагностической экспертной системе, а также упрощает процедуру регистрации диагностических процессов во временной области.

ЛИТЕРАТУРА:

1. Claudia Martins Antunes and Arlindo L. Oliveira. Temporal data mining: an overview //EPIA-2001 Workshop on Artificial Intelligence Techniques for Financial Time Series Analysis. 2001.
2. Luca Console, Claudia Picardi. Temporal Decision Trees: Model-based Diagnosis of Dynamic Systems On-Board //Journal of Artificial Intelligence Research. 2003. №19. P. 469-512.
3. Krivulja G.F.,A. A. Davidov.Optimizacija binarnih virishal'nih derev pri intelektual'nij diagnostici komp'juternih sistem // Radioelektronni i komp'juterni sistemi. 2010. № 6 (47).S. 260-265.
4. Krivulja G.F., I.V. Vlasov, O.A. Pavlov. Operativnoe funkcional'noe diagnostirovanie tehnicheskih ob#ektov s primeneniem temporal'nyh derev'ev reshenij.. Sb. nauchnyh trudov konferencii «Intellektual'nye sistemy prinjatija reshenij i problemy vychislitel'nogo intellekta». Evpatorija – 2013. S.193-195.
5. Krivulja G.F. , O.A. Pavlov, I.V. Vlasov. Primenenie temporal'nyh derev'ev reshenij dlja diagnostiki slozhnyh tehnicheskih ob#ektov. Tezi dopovidej 9-oї naukovoї konferencії Harkivs'kogo universitetu povitrjanih sil im..I. Kozheduba «Novitni tehnologії dlja zahistu povitrjanogo prostoru».
6. Harkiv –2013.S.213-214.
7. Krivulja G.F. , O.A. Pavlov, I.V. Vlasov. Optimizacija produkcionnyh pravil v diagnosticheskoj jekspertnoj sisteme. Materialy 13-oj mezhdunarodnoj nauchno-tehnicheskoj konferencii «Problemy informatiki i upravlenija». Har'kov-Jalta –2013. S. 5-7.