Math-Net.Ru

Общероссийский математический портал

Н. Н. Хрисанов, Д. Б. Фролагин, Метод решения задач поиска неисправностей,
Вестн. Сам. гос. техн. ун-та. Сер. Физ.-мат. науки, 2001, выпуск 12, 170–178

DOI: <https://doi.org/10.14498/vsgtu79>

Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и
согласны с пользовательским соглашением

<http://www.mathnet.ru/rus/agreement>

Параметры загрузки:

IP: 18.234.197.8

26 апреля 2023 г., 16:15:38

МЕТОД РЕШЕНИЯ ЗАДАЧ ПОИСКА НЕИСПРАВНОСТЕЙ

Описывается метод построения оптимальных стратегий поиска неисправностей в сложных технических объектах на основе решения задачи о наименьшем покрытии. Рассмотрено применение предложенного метода при решении различных задач, возникающих при контроле и диагностике сложных объектов.

Контроль и диагностика различного промышленного оборудования является неотъемлемой частью его серийного производства, эксплуатации, экспериментальных исследований. Применение оптимальных программ контроля и диагностики позволяет значительно сократить время и материальные затраты на определение истинного состояния контролируемого объекта, повысить достоверность получаемых результатов испытаний.

1 Постановка задачи. В настоящее время при постановке задач поиска неисправностей в качестве математической модели самого объекта контроля (ОК) обычно используется его функционально-логическая модель (ФЛМ) – некоторая совокупность блоков, соединенных между собой функциональными связями [1].

Р и с. 1. Функционально-логическая модель ОК

Пример ФЛМ, состоящей из 5 элементов, приведен на рис.1, где через X обозначены внешние входы ОК; Y – внешние выходы; Z – внутренние входы и выходы. Отдельные блоки (1...5) представляют собой элементарные звенья или элементы ОК. Совокупность всех элементов составляет множество Ω , причем $|\Omega| = n$. Каждый

элемент может находиться в одном из двух возможных состояний: работоспособности или неработоспособности (отказа). В состоянии работоспособности элемент отвечает требуемой реакцией на допустимую совокупность воздействий, в состав которых могут входить и реакция других элементов. В состоянии неработоспособности реакция элемента выходит за пределы допустимой. Отказы элементов возникают независимо.

Для контроля работоспособности и поиска мест отказов ОК имеется возможность проверки ряда в общем случае взаимосвязанных параметров, номинальное значение каждого из которых обеспечивается определенным подмножеством элементов Ω_i . Проверка каждого параметра осуществляется в результате проведения теста – ряда элементарных операций, в состав которых может входить подача необходимых входных воздействий, коммутация цепей ОК, измерение реакции в одной или нескольких контрольных точках и т.п. Обозначим через t_i тест, проверяющий работоспособность подмножества Ω_i , $i = \overline{1, m}$. В результате применения каждого теста можно получить только два исхода: “годен”, если работоспособны все элементы из Ω_i , и “не годен”, если по крайней мере один из элементов из Ω_i неработоспособен. Применение каждого теста связано с некоторыми затратами t_i , которые могут обозначать время, необходимое для проверки параметра, денежную стоимость требующейся при этом аппаратуры и т.д.

ФЛМ и возможную совокупность тестов обычно представляют в табличном виде (рис.2). Для этого обозначают символом “1” элементы, работоспособность которых контролируется некоторым тестом t_i , и символом “0” элементы, остающиеся непроверенными после применения t_i . Тогда, если элементы ОК некоторым образом пронумерованы, то каждому тесту можно поставить в соответствие его двоичный код и перечень этих кодов составит таблицу (матрицу) допустимых значений тестов. Если обозначить эту матрицу $T = \|t_{ij}\|$, $i = \overline{1, m}$, $j = \overline{1, n}$, то ее элементы определяются следующим образом:

$$t_{ij} = \begin{cases} 1, & \text{если } j \in \Omega_i, \\ 0, & \text{если } j \notin \Omega_i, \quad i = \overline{1, m}, \quad j = \overline{1, n}. \end{cases}$$

Матрице T поставлен в соответствие вектор столбец $t = \{t_1, \dots, t_m\}$, компоненты которого определяют затраты на применение каждого теста. Заданы априорные независимые вероятности q_j , $j = \overline{1, n}$ выхода из строя каждого элемента.

Целью решения задачи поиска является определение состояния всех элементов множества Ω , или определение подмножества элементов, обладающих определенным свойством. Поиск производится последовательным применением тестов матрицы T , причем выбор теста для применения на каждом следующем шаге зависит от исходов предшествующих тестов. Задача состоит в выборе некоторой совокупности тестов, достаточной для решения поставленной задачи поиска, и в определении условного порядка применения тестов этой совокупности так, чтобы обеспечить минимальное значение некоторого критерия.

Решению этой задачи посвящено значительное число работ. Достаточно подробно задачи подобного типа рассмотрены в [1,2]. В [1] предложен единый подход к решению задач контроля, так называемый рекурсивный метод, являющийся комбинацией метода динамического программирования и метода ветвей и границ.

В данной работе рассматривается метод решения наиболее распространенных задач поиска неисправностей, основанный на решении задачи о наименьшем покрытии (ЗНП) или ее частного случая - задачи о наименьшем разбиении (ЗНР) [3].

2 Описание метода решения задачи. Пусть задана ФЛМ некоторого объекта, содержащего n элементов, составляющих множество Ω , априорные независимые вероятности q_j , $j = \overline{1, n}$, матрица тестов $T = \|t_{ij}\|$, $i = \overline{1, m}$, $j = \overline{1, n}$, достаточная для определения состояния любого элемента, и вектор-столбец $t = \{t_1, \dots, t_m\}$ затрат на проведение тестов. Необходимо найти некоторую совокупность тестов, достаточную для определения состояния всех элементов, и определить условный порядок применения тестов этой совокупности так, чтобы обеспечить минимальное значение некоторого критерия.

Стратегию поиска удобно определить следующим образом:

$$S(\Omega) = [t_i^0, \dots, t_v^0], \quad (1)$$

т.е. как условный порядок применения (в порядке их написания) тестов $t_i^0, \dots, t_v^0$, где каждый следующий тест применяется только при положительном исходе предыдущего. Верхний индекс в обозначении теста означает номер стратегии, в которую входит данный тест (нулевой номер имеет стратегия для исходного множества Ω). При отрицательном исходе любого теста (например t_i^0) процесс переходит на частичную стратегию $(S(\Omega_i^0))$ определения состояния элементов в подмножестве (Ω_i^0) . Принцип записи частичных стратегий аналогичен (1).

Если определена некоторая стратегия $S(\Omega)$, то определены и все частичные стратегии $S(\Omega_i^j)$. Фактически частичные стратегии входят в состав $S(\Omega)$, но не записываются из-за громоздкости такой записи. Таким образом, стратегия $S(\Omega)$ определяет аналитическую форму записи программы поиска. Графически программу поиска можно представить дихотомическим деревом, вершины которого обозначают применяемые тесты, горизонтальные дуги – положительные исходы тестов, вертикальные – отрицательные исходы тестов (рис. 3).

Пусть определена некоторая стратегия $S(\Omega)$ со всеми частичными стратегиями $S(\Omega_i^j)$. Тогда соответствующие ей затраты, необходимые для определения состояния элементов множе-

Тесты T	Элементы ОК Ω				Затраты t
	1	2	...	N	
t_1	t_{11}	t_{12}		t_{1n}	t_1
t_2	t_{21}	t_{22}		t_{2n}	t_2
...					...
t_m	t_{m1}	t_{m2}		t_{mn}	t_m
Q	q_1	q_2	...	q_n	

Р и с. 2. Табличное представление тестов

ства Ω , являются случайной величиной, значение которой зависит от состояния элементов.

Р и с. 3. Графический способ представления стратегии поиска

определить соответственно состояние элементов из множества Ω_{j_k} . Тогда для выполнения задачи поиска необходимо

$$\left(\bigcup_{k=1}^n \Omega_{j_k} \right) \mathbf{I} \Omega_i^0 = \Omega_i^0. \quad (2)$$

Сформулируем правила, по которым какой либо тест из матрицы T может быть включен в стратегию $S(\Omega_i^0)$. Первый тест $t_{j_1}^i$ должен разбивать исходное множество Ω_i^0 на два подмножества, т.е.

$$\Omega_{j_1} \mathbf{I} \Omega_i^0 \neq \emptyset \text{ и } \Omega_{j_1} \mathbf{I} \Omega_i^0 \neq \Omega_i^0.$$

Второй тест должен удовлетворять условиям

$$\Omega_{j_2} \mathbf{I} (\Omega_i^0 \mathbf{I} \overline{\Omega_{j_1}}) \neq \emptyset \text{ и } \Omega_{j_2} \mathbf{I} (\Omega_i^0 \mathbf{I} \overline{\Omega_{j_1}}) \neq \Omega_i^0.$$

Соответственно k -тый ($k < n$) тест должен удовлетворять условиям:

$$\Omega_{j_k} \mathbf{I} \left(\Omega_i^0 \mathbf{I} \left(\bigcup_{p=1}^{k-1} \overline{\Omega_{j_p}^0} \right) \right) \neq \emptyset \text{ и } \Omega_{j_k} \mathbf{I} \left(\Omega_i^0 \mathbf{I} \left(\bigcup_{p=1}^{k-1} \overline{\Omega_{j_p}^0} \right) \right) \neq \Omega_i^0.$$

Для последнего n -го теста достаточно соблюдения одного условия

$$\Omega_{j_n} \mathbf{I} \left(\Omega_i^0 \mathbf{I} \left(\bigcup_{p=1}^{n-1} \overline{\Omega_{j_p}^0} \right) \right) \neq \emptyset.$$

Тесты, которые могут быть включены на данном этапе в стратегию, будем называть существенными. Из предыдущих рассуждений следует, что множество существенных тестов меняется по мере построения стратегии.

Из (2) следует, что тесты, вошедшие в некоторую стратегию $S(\Omega_i^0)$ производят **покрытие** исходного множества Ω_i^0 . Таким образом, если известны затраты на реализацию всех тестов, вошедших в стратегию $S(\Omega_i^0)$, а также всех соответствующих частичных стратегий, оптимальное значение $C[S^*(\Omega)]$ может быть найдено решением известной задачи о наименьшем покрытии (ЗНП). Причем, при решении ЗНП, затраты на реализацию любого теста t_j , вошедшего в стратегию $S(\Omega_i^k)$, должны определяться следующим образом:

$$t_j^i = t_j + q_j^i C[S(\Omega_j^i)], \quad (3)$$

где: t_j^i - средние суммарные затраты на проведение теста t_j при реализации $S(\Omega_i^k)$; q_j^i - вероятность применения частичной стратегии $S(\Omega_j^i)$ (т.е. вероятность отрицательного исхода теста t_j). Значения t_j^i и q_j^i зависят от того, для какого множества строится стратегия $S(\Omega_i^k)$.

Отметим, что множество Ω_j^i из (3), для которого строится стратегия $s(\Omega_j^i)$, в общем случае не совпадает с множеством Ω_j , т.е. с множеством, соответствующего тесту t_j , причем

$$\Omega_j^i \subseteq \Omega_j, |\Omega_j^i| \leq |\Omega_j|. \quad (4)$$

Множество Ω_j^i в дальнейшем будем называть существенным.

Из (4) следует возможность построения оптимальных программ поиска на основе решения ЗНП, при этом для построения стратегии $s(\Omega_j^i)$, вообще говоря, необходимо предварительно построить частичные стратегии для всех подмножеств Ω_j^i , которые могут быть использованы для построения стратегии $s(\Omega_j^i)$, т.е. необходимо построить частичные стратегии для всех подмножеств Ω_s , для которых

$$\Omega_s \cap \Omega_i^k \neq \emptyset \text{ и } |\Omega_s \cap \Omega_i^k| < |\Omega_i^k|. \quad (5)$$

Однако построение частичных стратегий для всех подмножеств, удовлетворяющих условию (5), является чрезвычайно трудоемкой задачей. Из чего следует, что для построения оптимальных стратегий целесообразно применять рекурсию с тем, чтобы строить частичные стратегии только для тех подмножеств, которые могут быть получены исходя из данной совокупности тестов.

Вместе с тем при решении ряда задач диагностики тесты должны удовлетворять условию неперекрываемости. При этом для любых двух тестов $t_{j_u}^i$ и $t_{j_v}^i$ должно выполняться:

$$\Omega_{j_u} \cap \Omega_{j_v} = \emptyset, \Omega_{j_u} \cap \Omega_i^0 = \Omega_{j_u}.$$

В этом случае построение оптимальной стратегии возможно решением ЗНР для подмножеств, контролируемых соответствующими тестами. При этом тесты, очевидно, предварительно необходимо расположить в порядке возрастания их мощности.

Применение ЗНР также возможно при решении многих задач диагностики при соблюдении следующих условий:

1) таблица тестов содержит все возможные тесты для исходного множества Ω (размер таблицы тестов $[2^n \times n]$);

2) затраты на тесты удовлетворяют условию:

$$t_i \geq t_j, \text{ если } \Omega_j \subseteq \Omega_i. \quad (6)$$

Рассмотрим алгоритм решения ЗНР [3] в приложении к задаче построения частичных оптимальных стратегий.

Сначала, используя исходную таблицу тестов, строятся “блоки” строк по одному на каждый элемент r_j из Ω , $j = \overline{1, n}$, причем k -тый блок состоит из таких тестов, в которых на k -том месте присутствует единица, а на любом $j < k$ месте стоит нуль (см. табл.1).

В процессе работы алгоритма блоки отыскиваются последовательно и формирование k -того блока начинается после того, как каждый элемент r_j , $1 \leq j \leq k-1$, будет покрыт частным решением. Таким образом, если для какого-либо теста t_j соответствующее множество Ω_j содержит элементы с индексами, меньшими k , оно должно быть отброшено (на этом этапе) в соответствии с требованием неперекрываемости. Множества в пределах каждого блока размещаются в порядке возрастания их стоимостей и перенумеровываются так, что тест t_j соответствует j -той строке таблицы тестов.

Построение блоков тестов при решении ЗНР

Тесты		Ω				
		r_1	r_2	r_3	$\dots$	r_n
Блок 1	t_1 $\dots$ t_k	1 $\dots$ 1	0 или 1			
Блок 2	t_{k+1} $\dots$ t_v	0	1 $\dots$ 1	0 или 1		
Блок 3	t_{v+1} $\dots$ t_u	0	0	1 $\dots$ 1	0 или 1	

Если B и $\hat{B}$ - соответственно текущее и наилучшее разбиения исходного множества, z и $\hat{z}$ - стоимость текущего и наилучшего разбиений, E - множество покрытых элементов тестами из B , то алгоритм решения ЗНР, использующий дерево поиска, может быть описан следующим образом.

Алгоритм 1

Присвоение начальных значений.

Шаг 1. Построить исходную таблицу и начать с частного решения: $B = \emptyset$, $E = \emptyset$, $z = 0$, $\hat{z} = \infty$.

Расширение.

Шаг 2. Найти $p = \min\{i | r_i \notin E\}$, т.е. первый непокрытый элемент. Над блоком p поставить метку (над его первым тестом, которое, как следует из построения таблицы, имеет наименьшую стоимость).

Шаг 3. Начиная с отмеченной позиции в блоке p , перебирать его тесты t_j^p в порядке возрастания индекса j , выбирая из них только те, которые удовлетворяют неравенству $z = \Gamma(B \cup t_j^p) < \hat{z}$, где $\Gamma(B \cup t_j^p)$ - нижняя оценка затрат на стратегию, если на данном шаге в текущее решение B включается тест t_j^p .

Если найден тест t_j^p , такой, что $t_j^p \notin E$ и $z = \Gamma(B \cup t_j^p) < \hat{z}$, то перейти к шагу 5. В противном случае, т.е. если блок p исчерпан, перейти к шагу 4.

Шаг возвращения.

Шаг 4. Множество B не может привести к лучшему решению. Если $B = \emptyset$ (т.е. блок 1 исчерпан), то алгоритм заканчивает работу и оптимальным решением является $\hat{B}$. В противном случае удалить последний тест из решения, скажем t_k^l , положить $p = l$, поставить метку над тестом t_{k+1}^l , удалить предшествующую метку в блоке l и перейти к шагу 3.

Проверка нового решения.

Шаг 5. Обновить данные: $B = B \cup t_j^p$, $E = E \cup \Omega_j^p$, $z = C[B]$, где $C[B]$ - стоимость текущего решения B . Если найдено лучшее решение $E = R$, то положить $\hat{B} = B$, $\hat{z} = z$ и перейти к шагу 4. Иначе перейти к шагу 3.

Недостатком описанного алгоритма является невозможность получения оптимального решения в том случае, когда значение критерия зависит от последовательности тестов в стратегии.

Эти недостатки исключены в следующем алгоритме, особенностью которого является то, что метки устанавливаются не над построенными блоками, а над тестами, входящими в текущее решение. Для удобства в этом алгоритме B и $\hat{B}$ представляют собой массивы соответственно для хранения текущего и наилучшего разбиений.

Алгоритм 2

Присвоение начальных значений.

Шаг 1. Начать с частного решения: $B = \emptyset$, $E = \emptyset$, $z = 0$, $\hat{z} = \infty$. Установить метку над первым элементом ($L=1$) решения B равную единице: $P[L]=1$. Назначим порядковый номер, равный единице тесту, который первым войдет в текущее решение: $L=1$. Приравнять единице метку для первого теста, которое войдет в решение: $P[L]=1$.

Расширение.

Шаг 2. Начиная с позиции $j = P[L]$, перебирать тесты t_j в порядке возрастания индекса j , выбирая из них только те, которые удовлетворяют неравенству $z = \Gamma(B \cup t_j) < \hat{z}$.

Если найден тест t_j , такой, что $t_j \notin E$ и $z = \Gamma(B \cup t_j) < \hat{z}$, то пометить t_j как t_j^L и перейти к шагу 4.

В противном случае, т.е. если таблица тестов исчерпана ($j > m$) или выбран такой тест t_j , что $z = \Gamma(B \cup t_j) \geq \hat{z}$, перейти к шагу 3.

Шаг возвращения.

Шаг 3. Множество B не может привести к лучшему решению. Если $B = \emptyset$ (таблица тестов исчерпана), то алгоритм заканчивает работу и оптимальным решением является $\hat{B}$. В противном случае метку над последним тестом t_j^L , вошедшим в решение B , приравнять единице: $P[L]=1$; снять верхнюю метку (L) с этого теста; обновить текущие затраты $z = C[B \setminus t_j^L]$; удалить этот тест из B ($B[L]=0$); обновить множество $E = E \cup \Omega_j^L$; уменьшить на единицу L и перейти к шагу 2.

Проверка нового решения.

Шаг 4. Обновить данные: $B[L]=t_j^L$, $E = E \cup \Omega_j^L$, $z = C[B]$. Если найдено лучшее решение $E = \Omega$, то положить $\hat{B} = B$, $\hat{z} = z$ и перейти к шагу 3. Иначе перейти к шагу 2.

Как показали вычислительные эксперименты, трудоемкость второго алгоритма несколько выше, что объясняется его большими возможностями.

Таким образом, при решении конкретных задач диагностики или контроля на основе решения ЗНП или ЗНР необходимо задать:

- 1) таблицу тестов с указанием затрат на их проведение и данными о вероятностях неисправности отдельных элементов;
- 2) вид критерия оптимизации, т.е. математическое выражение для вычисления стоимости текущего решения $z = C[B]$;
- 3) способ вычисления нижней оценки $\Gamma(B \cup t_j)$ при включении в текущее решение следующего теста;
- 4) определить тип алгоритма (последовательный или рекурсивный, на основе решения ЗНР или ЗНП), исходя из вида критерия оптимизации, состава таблицы тестов, соотношения между затратами на отдельные тесты.

3 Построение программ диагностирования. При построении оптимальных программ диагностирования предполагается достоверно известным факт существования в ОК одного отказавшего элемента. Вероятность отказа j -того элемента (при условии годности остальных) определяется следующим образом:

$$\tilde{Q}_j = q_j \prod_{\substack{k=1 \\ k \neq j}}^n (1 - q_k) \left[\sum_{i=1}^n q_i \prod_{\substack{l=1 \\ l \neq i}}^n (1 - q_l) \right]^{-1} = q_j \left[\sum_{i=1}^n q_i \right]^{-1}, \quad (7)$$

где $q_i = q_i / (1 - q_i)$, а вероятность отрицательного исхода теста, контролирующего состояние элементов подмножества Ω_j , как

$$Q_i = \sum_{j \in \Omega_i} \tilde{Q}_j, \quad (8)$$

Если тестами, выполненными на предыдущем этапе установлено, что отказавший элемент принадлежит множеству Ω_j ($\Omega_i \subset \Omega_j$), апостериорные вероятности (7), (8) необходимо нормировать делением их на априорную вероятность нахождения отказавшего элемента в подмножестве Ω_j :

$$\tilde{Q}_{i(j)} = \tilde{Q}_i / Q_j.$$

Пусть для диагностики подмножества Ω_i используется частичная стратегия $s(\Omega_i) = [t_1^i, \dots, t_k^i]$. Средние затраты на реализацию этой стратегии равны:

$$C[s(\Omega_i)] = \frac{1}{\tilde{Q}_i} \left\{ \sum_{l=1}^{k-1} \left[t_l \sum_{u=l}^k \tilde{Q}_u^i \right] + \sum_{l=1}^k C[s(\Omega_i)] \tilde{Q}_l^i \right\}. \quad (9)$$

Если известны все частичные стратегии, то используя (9) рекурсивным образом, можно вычислить затраты на реализацию основной стратегии.

Из (9) следует, что значение средних затрат зависит от порядка выполнения тестов. Различаются при этом также значения и даже набор существенных множеств $\tilde{Q}_u^i$, что в общем случае предполагает применение рекурсивного алгоритма на основе ЗНП.

Рассмотрим возможность применения ЗНР для формирования частичных стратегий диагностики (безусловно, при этом должно выполняться условие (6)). При решении ЗНР по алгоритму 1 не принимается внимание последовательность вхождения тестов в стратегию. Поэтому необходимо производить дополнительно операцию упорядочения в пределах стратегии, обеспечивающая минимум критерия (9). Покажем также, что такое упорядочение в пределах одной стратегии не влияет на оптимальность всей программы диагностики.

Оптимальную последовательность определим используя перестановочный прием. Пусть $C_{j,j+1}$ - затраты на стратегию, в которой после j -того теста следует тест с номером $j+1$, $C_{j+1,j}$ - затраты на стратегию, в которой после $j+1$ -того теста следует тест с номером j .

Учитывая (9), получим:

$$C_{j,j+1}[s(\Omega_i)]\tilde{Q}_i = B_1 + t_j\tilde{Q}_j^i + t_j\tilde{Q}_{j+1}^i + B_j + t_{j+1}\tilde{Q}_{j+1}^i + B_{j+1} + B_2, \quad (10)$$

$$C_{j+1,j}[s(\Omega_i)]\tilde{Q}_i = B_1 + t_{j+1}\tilde{Q}_{j+1}^i + t_{j+1}\tilde{Q}_j^i + B_{j+1} + t_j\tilde{Q}_j^i + B_j + B_2, \quad (11)$$

где $B_1 = \sum_{l=1}^{j-1} \left[t_l \sum_{u=l}^k \tilde{Q}_u^i \right] + \sum_{l=1}^{j-1} c[s(\Omega_l^i)]\tilde{Q}_l^i$, $B_2 = \sum_{l=j+2}^{k-1} \left[t_l \sum_{u=l}^k \tilde{Q}_u^i \right] + \sum_{l=j+2}^k c[s(\Omega_l^i)]\tilde{Q}_l^i$,

$$B_j = t_j \sum_{u=j+2}^k \tilde{Q}_u^i + c[s(\Omega_j^i)] + c[s(\Omega_{j+1}^i)], \quad B_{j+1} = t_{j+1} \sum_{u=j+2}^k \tilde{Q}_u^i + c[s(\Omega_j^i)] + c[s(\Omega_{j+1}^i)].$$

Вычитая (11) из (10), получим

$$\{C_{j,j+1}[s(\Omega_i)] - C_{j+1,j}[s(\Omega_i)]\}\tilde{Q}_i = t_j\tilde{Q}_{j+1}^i - t_{j+1}\tilde{Q}_j^i. \quad (12)$$

Последнее выражение (12) отрицательно при

$$t_j/\tilde{Q}_j^i < t_{j+1}/\tilde{Q}_{j+1}^i. \quad (13)$$

Это означает, что в построенных стратегиях тесты должны быть упорядочены в соответствии с (13). Кроме того можно сделать вывод о возможности упорядочения тестов в пределах одной стратегии, поскольку в (13) не входят затраты на частичные стратегии $C[s(\Omega_j^i)]$, $j = \overline{1, k}$.

Вычисление нижних оценок промежуточных решений позволяет значительно сократить объем вычислений за счет отбрасывания промежуточных решений. В [1] предложены различные достаточно эффективные способы вычисления нижних оценок. Недостатком их является значительная вычислительная сложность. Предлагается строить оценки, исходя из минимальных затрат на диагностирование каждого элемента в отдельности. Преимуществом такого подхода является то, что значительная часть вычислений производится до применения основного алгоритма построения стратегий. Возможно применение этого способа и в ходе построения стратегий, при этом точность вычисления нижних оценок возрастает за счет некоторого ухудшения вычислительных характеристик алгоритма.

Сущность этого способа следующая. Пусть в ходе построения стратегии, или, что то же самое, решения ЗНР, произведено частичное покрытие исходного множества Ω_i тестами t_j , $j = \overline{1, k}$. Пусть непокрытые элементы образуют множество Ω_v^i . Тогда в качестве оценки текущего решения можно взять следующее значение:

$$\Gamma[s(\Omega_i)] = \frac{1}{\tilde{Q}_i} \left\{ \sum_{l=1}^k \left[t_l \left(1 - \sum_{u=1}^{l-1} \tilde{Q}_u^i \right) \right] + \sum_{l=1}^k c[s(\Omega_l^i)]\tilde{Q}_l^i + \sum_{r_l \in \Omega_v^i} Q_l^* t_l^* \right\},$$

где t_l^* - затраты на проведение теста, который может быть использован для диагностирования элемента r_l , входящего в Ω_v^i и имеющего минимальное значение; Q_l^* - нижняя оценка вероятности применения этого теста. Для задачи построения программ диагностики в качестве t_l^* может быть взято значение затрат на выполнение такого теста t_j , для которого выполняется $r_l \in \Omega_j$, и затраты на выполнение минимальны. В качестве Q_l^* можно использовать $\tilde{Q}_l$, т.е. апостериорную вероятность отказа элемента r_l . Заметим, что при предварительной оценке t_l^* перед применением алгоритма используется вся таблица тестов. При оценке t_l^* в ходе выполнения алгоритма точность оценки очевидно можно повысить за счет того, что рассматривается только та совокупность тестов, которая может быть применена на данном этапе.

4 Построение программ идентификации. В этом случае предполагается наличие произвольного числа неисправностей. Возможно, в том числе, и работоспособное состояние всех элементов, при этом необходимо дополнить матрицу T дополнительным $(N + 1)$ столбцом, соответствующим фиктивному элементу, затраты на проверку которого равны нулю. Вероятность $\tilde{q}_i$ отказа в множестве Ω_i равна:

$$\tilde{q}_i = 1 - \prod_{j \in \Omega_i} (1 - q_j). \quad (14)$$

Как показано в [1], тесты должны удовлетворять условию непересекаемости. Пусть для идентификации подмножества Ω_i используется частичная стратегия $h(\Omega_i) = [t_1^i, \dots, t_k^i]$. Средние затраты на реализацию этой стратегии равны:

$$C[h(\Omega_i)] = \sum_{j=1}^{k-1} t_j + \frac{1}{\tilde{q}_i} \left\{ \sum_{j=1}^k \tilde{q}_j C[h(\Omega_j)] + t_k \left[1 - \prod_{j=1}^k (1 - \tilde{q}_j) - \tilde{q}_{N+1} \right] \right\}, \quad (15)$$

где $\tilde{q}_{N+1}$ - вероятность отказа фиктивного элемента, которая определяется как вероятность работоспособного состояния всех элементов Ω_i , если Ω_i включает фиктивный элемент, в противном случае $\tilde{q}_{N+1}$ полагается равной нулю; $\tilde{q}_j$ вычисляется по (14), если Ω_i не содержит фиктивного элемента, и $\tilde{q}_j = 1$ в противном случае. Последний член в (15) учитывает так называемый конечный эффект, когда последний тест частичной стратегии не применяется при положительном исходе всех предыдущих.

Учитывая вид критерия (15) (затраты не зависят от порядка, в котором применяются тесты, за исключением затрат на выполнение последнего теста) и условие непересекаемости тестов, в данном случае целесообразно стратегии идентификации строить на основе решения ЗНР (алгоритм 1) с последующим перемещением теста, обеспечивающего минимальный конечный эффект в конец стратегии.

Нижняя оценка текущего решения, включающего на данном этапе k тестов (при этом не покрытым остается множество Ω_v^i), вычисляется по формуле:

$$\Gamma[h(\Omega_i)] = \sum_{j=1}^k t_j + \frac{1}{\tilde{q}_i} \left\{ \sum_{j=1}^k \tilde{q}_j C[h(\Omega_j)] \right\} + \sum_{r_l \in \Omega_v^i} q_l^* t_l^*.$$

Значения могут t_l^* и q_l^* могут быть получены исходя из следующих соображений. Особенностью задачи идентификации является обязательное присутствие в построенной программе одноэлементных тестов, причем их количество равно числу элементов в исходном множестве Ω . Следовательно, в качестве t_l^* могут быть взяты затраты на проведение соответствующего одноэлементного теста, контролирующего элемент r_l . Уменьшить средние затраты на идентификацию элемента можно включением одноэлементного теста в какую либо частичную стратегию, которая предназначена для идентификации некоторого подмножества Ω_i , которому соответствует тест t_i . Учитывая то, что затраты на частичные стратегии (в данном случае затраты на проведение одноэлементного теста) входят в (15) с коэффициентом $\tilde{q}_i/\tilde{q}_i$ (где $\tilde{q}_i$ - апостериорная вероятность отказа элемента r_i), для получения нижней оценки q_l^* необходимо использовать максимальное значение $\tilde{q}_i$:

$$\tilde{q}_i^* = \tilde{q}_i / \tilde{q}_{i \max},$$

т.е. для вычисления q_l^* необходимо выбрать тест t_i , имеющий максимальное значение вероятности отрицательного исхода, и, разумеется, тест t_i должен удовлетворять условию $r_l \subset \Omega_i$.

5 Построение минимаксных планов. Если отсутствуют данные о надежности элементов, как правило, критерием оптимальности служит минимум максимально-возможных затрат. Для программ диагностирования (в объекте возможен один отказ) суммарные затраты на реализацию некоторой стратегии $s(\Omega)$ являются случайной величиной. Пусть $R[s(\Omega)]$ - максимальное значение этой случайной величины. Тогда задача сводится к построению оптимальной стратегии $s^*(\Omega)$, для которой

$$R[s^*(\Omega)] = \min_{s(\Omega)} R[s(\Omega)].$$

Если известны максимальные затраты на реализацию всех частичных стратегий $s(\Omega_i)$, $i = \overline{1, k}$, то $R[s(\Omega)]$ можно определить из выражения

$$R[s(\Omega)] = \max \left\{ \max_{1 \leq j \leq k-1} \left[\sum_{i=1}^j t_i + R[s(\Omega_i)] \right], \left[\sum_{i=1}^{k-1} t_i + R[s(\Omega_k)] \right] \right\}. \quad (16)$$

Поскольку в данном случае тесты, вошедшие в стратегию, могут пересекаться и значение критерия (16) зависит от последовательности тестов, то необходимо использовать рекурсивный алгоритм на основе решения ЗНП (алгоритм 2). Вычисление нижних оценок при этом целесообразно производить в соответствии с методикой, изложенной в [1].

6 Реализация алгоритмов. Рассмотренные алгоритмы реализованы в среде программирования Delphi. Они включают в себя средства для ввода исходных данных о ФЛМ объекта, автоматического формирования таблиц тестов исходя из априорных сведений о вероятностных характеристиках возможных отказов и затрат на проведение тестов, ведения базы данных для различных, ранее введенных ФЛМ, и результатов решения задач. Возможно построение оптимальных программ диагностики и идентификации для критерия средних затрат и минимаксного критерия. Предложенный метод также используется для построения оптимальных программ контроля с восстановлением и построения программ контроля с принятием байесовских решений в условиях неопределенности. Применение разработанных программ для решения тестовых и реальных задач показало достаточно высокую эффективность использованных алгоритмов решения задач поиска неисправностей.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Пашковский Г.С. Задачи оптимального обнаружения и поиска отказов в РЭА. М.: Радио и связь. 1981. 280 с.
2. Альсведе Р., Вегнер И. Задачи поиска. М.: Мир. 1982. 368 с.
3. Кристофидес Н. Теория графов. Алгоритмический подход. М.: Мир. 1978. 432 с.