**Практическое занятие № 2.**

**Исследование моделей надежности программного обеспечения с использованием моделей Джелинского – Моранды*,* эвристической** **и Нельсона**

**Время выполнения практического занятия (аудиторные часы)** –   
4 часа.

**Время самостоятельной работы студента (дополнительные часы)** – 4 часа.

**Цель работы:** изучение методик определения надежности программного обеспечения.

**Ключевые понятия, которые необходимо знать:** надежность, отказ, основные показатели надежности.

**Оборудование и программное обеспечение:** работа выполняется на ПЭВМ типа IBM PC с использованием пакета прикладных программ MathCad.

**ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

Модель надежности программного обеспечения - это математическая модель, построенная для оценки зависимости надежности программного обеспечения от некоторых определенных параметров.Основные модели надежности программных средств представлены на рис. 1

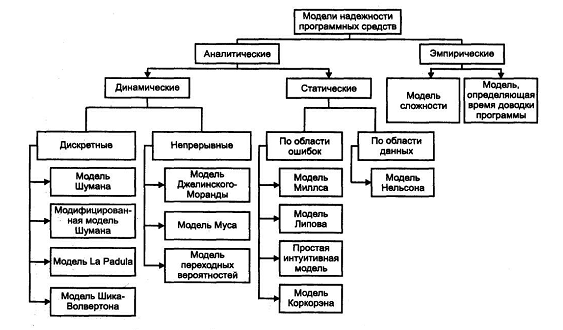


Рис. 1 Основные модели надежности программных средств

Аналитические модели позволяют рассчитать количественные показатели надежности, основываясь на данных о поведении программы в процессе тестирования. Делятся на динамические и статические. В динамических моделях поведение ПО (появление отказов) рассматривается во времени (дискретные – делятся на интервалы, статические – определённый момент времени на всей числовой оси).

В статических моделях появление отказов не связывают со временем, а учитывают зависимость количества ошибок либо от числа тестовых прогонов, либо от характеристики входных данных.

**Модель Джелинского – Моранды**

***Модель Джелинского - Моранды*** (Z. Jelinski, P. Moranda) предложена в 1996 г. Иначе данная модель называется моделью роста надежности. Она основана на предположении об экспоненциальной зависимости плотности вероятности интервалов времени между проявлением ошибок от интенсивности ошибок. Кроме того, в модели полагается, что интенсивность ошибок на каждом случайном интервале времени линейно зависит от количества оставшихся в программе ошибок.

Если допустить, что ошибка после ее каждого проявления устраняется и при этом в программный модуль не вносятся новые, то интенсивность ошибок *λ(ti)*на интервале *ti* - определяется следующим соотношением:



(1)

где ***N*** - количество ошибок до начала тестирования; i = 1**...** *т (т* ***-***число ошибок, обнаруженных во время тестирования**);** *к*- коэффициент пропорциональности.

Для плотности вероятности ошибки *p(ti)*на случайном интервале *ti*справедливо выражение:

(2)



Применяя для двух неизвестных этого уравнения (ими в выражении (2) являются величины *к* и*N)*метод максимального правдоподобия, авторы этого метода оценки надежности программных средств получили следующую систему из двух уравнений:



(3)

где n-количество обнаруженных в процессе тестирования ошибок.

Уравнения, образующие систему (3), представляют модель Джелински - Моранды, позволяющей оценить количество ошибок в программе до начала тестирования ***N*** по количеству обнаруженных ошибок n.

Для упрощения расчетов считают, что каждый тест может обнаружить только одну ошибку. Продолжительность интервала тестирования *ti*, измеряется не в единицах времени, а количеством тестов, которое потребовалось для обнаружения очередной ошибки. Таким образом, если очередная ошибка обнаруживается одним тестом, то интервал времени равен единице. Если ошибка обнаруживается *т*тестами (т. е. тест с номером *т*- 1 не обнаружил ошибки, в тесте *т*ошибка была обнаружена), то интервал времени равен *т*(рисунок 1).

7

6

5

4

3

2

1

t

m

1

2

3

Рисунок 1 - Учет интервалов времени при выявлении ошибок

Важным условием применимости на практике модели Джелински -Моранды является соответствие результатов тестирования допущению об уменьшении интенсивности ошибок после устранения очередной ошибки. Подтверждением этого соответствия должно быть увеличение интервалов времени (количества тестов) для обнаружения каждой последующей ошибки.

Таким образом, модель Джелински - Моранды основывается на соблюдении следующих условий:

* **экспоненциальная зависимость плотности вероятности интервалов времени между появлением ошибок;**
* **интенсивность ошибок линейно зависит от количества оставшихся ошибок на любом случайном интервале;**
* **каждый тест находит только одну ошибку;**
* **после каждого появления ошибка устраняется и не вносится новая ошибка.**

В силу того, что с течением времени интенсивность ошибок уменьшается и растет интервал между проявлением ошибок, модель **Джелински - Моранды** в некоторых источниках называют **моделью роста надежности**.

**Постановка задачи 1: Определение количества ошибок до начала тестирования**

В результате тестирования программы серией из *M* случайно выбранных из набора тестов обнаружено ***n*** ошибки. Ошибки обнаружены ***m*** тестами.

**Задание на выполнение практического занятия:**

1. Для заданных исходных данных, определить количество ошибок до начала тестирования (задание по вариантам приведено в таблице 1, номер варианта устанавливается преподавателем).

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| № варианта | Кол-во обнаруженных ошибок,  n | **Номера тестов, в которых обнаружены ошибки**,  m | **Количество тестов,**  **M** |
| 1 | 2 | 1,3 | 4 |
| 2 | 2 | 2,6 | 6 |
| 3 | 2 | 1,8 | 10 |
| 4 | 2 | 3,8 | 8 |
| 5 | 2 | 4,11 | 11 |
| 6 | 2 | 3,10 | 11 |

2. Определить количество ошибок ***N*** в программе до начала тестирования.

**Постановка задачи 2:** Определение количества ошибок в программе, не устраненных после проведения тестирования

В результате тестирования программы серией из *M* случайно выбранных из набора тестов обнаружено *n* ошибок. Ошибки обнаружены *m* тестами. Все ошибки исправлены сразу после обнаружения.

***Замечание: исправление ошибок не повлекло появление новых ошибок.***

Результаты расчетов округлять в большую или меньшую сторону по стандартным правилам (например, если округлить число 2,3, то получим 2, а если округлить 2,5 или 2,6, то после округления получим 3).

**Задание на выполнение практического занятия:**

1. Для заданных исходных данных, определить количество ошибок не устраненных после проведения тестирования(задание по вариантам приведено в таблице 2, номер варианта устанавливается преподавателем).

Таблица 2

|  |  |  |  |
| --- | --- | --- | --- |
| № варианта | Кол-во обнаруженных ошибок,  n | **Номера тестов, в которых обнаружены ошибки**,  m | **Количество тестов,**  **M** |
| 1 | 2 | 1,4 | 4 |
| 2 | 2 | 3,10 | 11 |
| 3 | 2 | 1,13 | 14 |
| 4 | 2 | 1,10 | 11 |
| 5 | 2 | 1,11 | 15 |
| 6 | 2 | 1,5 | 10 |

***Эвристическая модель***

***Эвристическая модель*** оценки надежности программных средств позволяет оценить количество ошибок ***N*** до начала тестирования по результатам тестирования программы двумя независимыми группами. Для этого применяется следующее выражение:





(4)

где: ***N1*** - количество ошибок, обнаруженных первой группой тестирующих;

N**2** - количество ошибок, обнаруженных второй группой тестирующих;

***N1,*2** - количество ошибок, которые обнаружила и первая, и вторая группа (общие обнаруженные ошибки).

Эвристическая модель хорошо работает при «перекрестном» тестировании программ несколькими группами тестировщиков, поскольку обеспечивает достаточно легкую обработку получаемых результатов.

**Постановка задачи 1:**

Программа тестируется двумя независимыми группами тестировщиков, которые силами групп выявили в программе *N1* и *N2* ошибок соответственно. При этом оказалось, что *N1,2* ошибок - общие, их нашли обе группы.

**Задание на выполнение практического занятия:**

1. Для заданных исходных данных, оценить общее количество ошибок в программе до начала тестирования и сделать вывод о необходимости продолжения тестирования или возможности его завершении (задание по вариантам приведено в таблице 3, номер варианта устанавливается преподавателем).

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
| № варианта | Количество ошибок, обнаруженных первой независимой группой тестировщиков,  ***N1*** | Количество ошибок, обнаруженных второй независимой группой тестировщиков,  ***N2*** | Количество ошибок, обнаруженных как первой, так и второй группой тестировщиков,  N1,2 |
| 1 | 40 | 20 | 10 |
| 2 | 15 | 9 | 5 |
| 3 | 33 | 26 | 16 |
| 4 | 29 | 31 | 4 |
| 5 | 7 | 13 | 6 |
| 6 | 15 | 19 | 8 |

**Постановка задачи 2:**

Две независимые группы тестировщиков проводили тестирование программного средства. Первая группа обнаружила *N1*  ошибок, а вторая - *N2*. На основании результатов тестирования было определено, что до начала тестирования в программе содержалось *N* ошибки.

**Задание на выполнение практического занятия:**

1. Для заданных исходных данных, определить, сколько ошибок было обнаружено как первой, так и второй группой тестировщиков (задание по вариантам приведено в таблице 4, номер варианта устанавливается преподавателем).

Таблица 4

|  |  |  |  |
| --- | --- | --- | --- |
| № варианта | Количество ошибок, обнаруженных первой независимой группой тестировщиков,  ***N1*** | Количество ошибок, обнаруженных второй независимой группой тестировщиков,  ***N2*** | Количество ошибок до начала тестирования,  N |
| 1 | 40 | 20 | 42 |
| 2 | 15 | 9 | 57 |
| 3 | 33 | 26 | 38 |
| 4 | 29 | 31 | 19 |
| 5 | 7 | 13 | 21 |
| 6 | 15 | 19 | 31 |

## **Модель Нельсона**

Классической измерительной моделью надежности является известная модель Нельсона, предложенная в 1978 г. Модель основана на выделении областей исходных данных *Ei*покрывающих все множество вариантов их использования в программе Е:

Множество результатов выполнения программы F(E) определяется композицией результатов по всем вариантам данных F(Ei):



(5)

Выражение (5) позволяет определить программу как описание некоторой вычисляемой функции F на множестве всех значений наборов входных данных Е.

Если обозначить Fф(Ei) множество фактических результатов выполнения программы на наборе *Ei*, то множество правильных значений будет определяться условием:

(6)

где εi - допустимое расхождение между правильным (эталонным) и фактическим результатом выполнения программы на наборе Еi.

Рабочим отказом считается ситуация, при которой выполняется условие:

(7)

или программа «зацикливается» (выполняется бесконечное продолжение работы программы, при котором программа не может закончиться). При этом предполагается, что выбор любого набора исходных данных Ei⊂E равновероятен.

Мощность всего множества наборов исходных данных Е обозначим символом N. Множество, состоящее из всех наборов Ei, для которых получены неудовлетворительные результаты, обозначим *Eo.* Мощность множества *Eo* обозначим No. Совокупность действий, включающих ввод Ei и выполнение программы, которое заканчивается получением результата Fф(Ei) или рабочим отказом, называется прогоном программы. Следует заметить, что входные данные, образующие набор Ei не обязательно должны подаваться на вход одновременно.

При этих допущениях справедливо утверждать следующее: вероятность Р того, что прогон программы приведет к рабочему отказу, равна вероятности того, что набор данных *Ei,* который использовался в прогоне, принадлежит множеству Еo. Тогда вероятность появления ошибки при прогоне программы на входном наборе, случайно выбранном из числа равновероятных, определяется следующим выражением:

(8)

Вероятность R того, что прогон программы на наборе входных данных Ei, случайно выбранном из Е среди равновероятных наборов, приведет к приемлемому результату, определяется выражением:

(9)

Если выбор набора данных из множества Е не равновероятен, а возможны какие-либо приоритеты выбора набора данных, то для оценки надежности программы следует использовать другое выражение:

(10)

где: pi - вероятность (частота) использования i-го набора исходных данных,

уi - динамическая переменная, которая принимает нулевое значение, если прогон заканчивается приемлемым результатом, и значение 1, если прогон заканчивается рабочим отказом.

Вероятность R(n) успешного выполнения п прогонов программы при независимом для каждого прогона выборе исходных данных определяется выражением:

(11)

Эту вероятность R(ni) можно представить в следующем виде:

(12)

где *pj* - вероятность отказа для *j*-го прогона.

Дальнейшие преобразования формулы (12) показывают значительное сходство с технологией определения безотказности, принятой в теории надежности технических устройств. Модель Нельсона в наибольшей степени отражает традиционный подход, принятый для определения надежности технических устройств, для измерения надежности программ.

**Постановка задачи 1:**

Для испытания программы использовалось ***N*** наборов исходных данных, которые равновероятно выбирались для прогона ***N*** тестов. При этом ***N0*** тестов обнаружили дефекты программного обеспечения.

**Задание на выполнение практического занятия 1:**

1. Для заданных исходных данных, рассчитать надежность программного обеспечения по результатам испытаний (задание по вариантам приведено в таблице 5, номер варианта устанавливается преподавателем).

**Таблица 5**

|  |  |  |
| --- | --- | --- |
| № варианта | Общее кол-во тестов,  ***N*** | Количество тестов с обнаружением дефектов программы,  ***N0*** |
| 1 | 20 | 10 |
| 2 | 11 | 5 |
| 3 | 17 | 12 |
| 4 | 18 | 3 |
| 5 | 28 | 17 |
| 6 | 21 | 15 |

**Постановка задачи 2:**

**Вариант 1:**

Для испытания программы использовалось 30 наборов исходных данных, которые выбирались в соответствии с функцией распределения частот, значения которой представлены ниже.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № теста | Частота выбора  теста | Исход прогона  теста | № теста | Частота выбора  теста | Исход прогона  теста |
| 1 | 0,04 | 1 | 16 | 0,01 | 0 |
| 2 | 0,01 | 0 | 17 | 0,02 | 1 |
| 3 | 0,03 | 0 | 18 | 0,01 | 0 |
| 4 | 0,05 | 0 | 19 | 0,03 | 1 |
| 5 | 0,02 | 1 | 20 | 0,19 | 0 |
| 6 | 0,03 | 0 | 21 | 0,03 | 1 |
| 7 | 0,05 | 1 | 22 | 0,02 | 0 |
| 8 | 0,01 | 0 | 23 | 0,04 | 1 |
| 9 | 0,04 | 0 | 24 | 0,01 | 1 |
| 10 | 0,01 | 0 | 25 | 0,02 | 1 |
| 11 | 0,02 | 1 | 26 | 0,01 | 1 |
| 12 | 0,07 | 0 | 27 | 0,03 | 1 |
| 13 | 0,01 | 0 | 28 | 0,06 | 1 |
| 14 | 0,02 | 1 | 29 | 0,02 | 1 |
| 15 | 0,05 | 1 | 30 | 0,04 | 1 |

В 17 тестах были обнаружены ошибки. Все исходы прогонов, закончившиеся отказом, в таблице обозначены единицами.

**Вариант 2:**

Для испытания программы использовалось 16 наборов исходных данных, которые выбирались в соответствии с функцией распределения частот, значения которой представлены ниже.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № теста | Частота выбора  теста | Исход прогона  теста | № теста | Частота выбора  теста | Исход прогона  теста |
| 1 | 0,05 | 0 | 9 | 0,07 | 0 |
| 2 | 0,03 | 1 | 10 | 0,04 | 0 |
| 3 | 0,07 | 1 | 11 | 0,08 | 1 |
| 4 | 0,06 | 0 | 12 | 0,09 | 0 |
| 5 | 0,08 | 1 | 13 | 0,04 | 1 |
| 6 | 0,1 | 1 | 14 | 0,07 | 0 |
| 7 | 0,06 | 1 | 15 | 0,06 | 0 |
| 8 | 0,06 | 0 | 16 | 0,05 | 0 |

В 7 тестах были обнаружены ошибки. Все исходы прогонов, закончившиеся отказом, в таблице обозначены единицами.

**Вариант 3:**

Для испытания программы использовалось 18 наборов исходных данных, которые выбирались в соответствии с функцией распределения частот, значения которой представлены ниже.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № теста | Частота выбора  теста | Исход прогона  теста | № теста | Частота выбора  теста | Исход прогона  теста |
| 1 | 0,06 | 0 | 10 | 0,05 | 0 |
| 2 | 0,03 | 0 | 11 | 0,06 | 0 |
| 3 | 0,07 | 1 | 12 | 0,05 | 0 |
| 4 | 0,06 | 0 | 13 | 0,04 | 0 |
| 5 | 0,07 | 1 | 14 | 0,07 | 0 |
| 6 | 0,05 | 1 | 15 | 0,06 | 1 |
| 7 | 0,06 | 1 | 16 | 0,03 | 0 |
| 8 | 0,07 | 1 | 17 | 0,07 | 0 |
| 9 | 0,04 | 1 | 18 | 0,06 | 1 |

В 8 тестах были обнаружены ошибки. Все исходы прогонов, закончившиеся отказом, в таблице обозначены единицами.

**Вариант 4:**

Для испытания программы использовалось 18 наборов исходных данных, которые выбирались в соответствии с функцией распределения частот, значения которой представлены ниже.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № теста | Частота выбора  теста | Исход прогона  теста | № теста | Частота выбора  теста | Исход прогона  теста |
| 1 | 0,09 | 1 | 10 | 0,08 | 0 |
| 2 | 0,02 | 0 | 11 | 0,09 | 0 |
| 3 | 0,06 | 1 | 12 | 0,05 | 1 |
| 4 | 0,05 | 0 | 13 | 0,04 | 1 |
| 5 | 0,07 | 1 | 14 | 0,07 | 0 |
| 6 | 0,05 | 1 | 15 | 0,06 | 1 |
| 7 | 0,03 | 1 | 16 | 0,05 | 0 |
| 8 | 0,02 | 1 | 17 | 0,1 | 0 |
| 9 | 0,04 | 1 | 18 | 0,03 | 1 |

В 11 тестах были обнаружены ошибки. Все исходы прогонов, закончившиеся отказом, в таблице обозначены единицами.

**Вариант 5:**

Для испытания программы использовалось 22 набора исходных данных, которые выбирались в соответствии с функцией распределения частот, значения которой представлены ниже.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № теста | Частота выбора  теста | Исход прогона  теста | № теста | Частота выбора  теста | Исход прогона  теста |
| 1 | 0,05 | 0 | 12 | 0,05 | 0 |
| 2 | 0,03 | 0 | 13 | 0,07 | 1 |
| 3 | 0,03 | 1 | 14 | 0,07 | 0 |
| 4 | 0,05 | 0 | 15 | 0,06 | 0 |
| 5 | 0,06 | 1 | 16 | 0,06 | 0 |
| 6 | 0,05 | 1 | 17 | 0,09 | 1 |
| 7 | 0,03 | 1 | 18 | 0,03 | 1 |
| 8 | 0,02 | 0 | 19 | 0,01 | 0 |
| 9 | 0,04 | 0 | 20 | 0,03 | 1 |
| 10 | 0,06 | 1 | 21 | 0,02 | 0 |
| 11 | 0,07 | 0 | 22 | 0,02 | 0 |

В 9 тестах были обнаружены ошибки. Все исходы прогонов, закончившиеся отказом, в таблице обозначены единицами.

**Вариант 6:**

Для испытания программы использовалось 24 набора исходных данных, которые выбирались в соответствии с функцией распределения частот, значения которой представлены ниже.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № теста | Частота выбора  теста | Исход прогона  теста | № теста | Частота выбора  теста | Исход прогона  теста |
| 1 | 0,07 | 1 | 13 | 0,04 | 1 |
| 2 | 0,04 | 0 | 14 | 0,03 | 1 |
| 3 | 0,06 | 1 | 15 | 0,05 | 0 |
| 4 | 0,05 | 0 | 16 | 0,04 | 1 |
| 5 | 0,05 | 0 | 17 | 0,05 | 1 |
| 6 | 0,05 | 1 | 18 | 0,03 | 0 |
| 7 | 0,03 | 0 | 19 | 0,01 | 1 |
| 8 | 0,05 | 1 | 20 | 0,03 | 1 |
| 9 | 0,04 | 1 | 21 | 0,03 | 0 |
| 10 | 0,05 | 0 | 22 | 0,02 | 1 |
| 11 | 0,05 | 1 | 23 | 0,04 | 1 |
| 12 | 0,05 | 0 | 24 | 0,04 | 1 |

В 15 тестах были обнаружены ошибки. Все исходы прогонов, закончившиеся отказом, в таблице обозначены единицами.

**Задание на выполнение практического занятия 2:**

1. Для заданных исходных данных, определить надежность программы по результатам испытаний.

**3. Контрольные вопросы**

1. Как классифицируются основные модели надежности программных средств.
2. На каких допущениях основана модель Джелинского – Моранды?
3. Почему модель Джелинского – Моранды называют «модель роста надежности»?
4. Как оценить надежность программ при помощи эвристической модели? Где данная модель может применятся практически?
5. Что в модели Нельсона признается «отказом»?
6. Что называется «прогоном программы»?

**4. Требования к отчету о проделанной работе**

Отчёт выполняется каждым студентом индивидуально. Работа должна быть оформлена в электронном виде в формате .doc и распечатана на листах формата А4. На титульном листе указываются: наименование учебного учреждения, наименование дисциплины, название и номер работы, вариант, выполнил: фамилия, имя, отчество, группа, проверил: преподаватель ФИО.

Отчет должен содержать:

* распечатку исходных данных;
* используемые формульные соотношения;
* выводы по каждому пункту работы.