ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №4

«ПОСТРОЕНИЕ И АНАЛИЗ АЛГОРИТМОВ РАСПОЗНАВАНИЯ ТЕХНИЧЕСКИХ СОСТОЯНИЙ СИСТЕМЫ»

ПО КУРСУ: «РАСПОЗНАВАНИЕ ОБРАЗОВ»

**ЛАБОРАТОРНАЯ РАБОТА №4**

**Исходные данные на лабораторную работу**

Согласно варианту №12 (№2), задана функциональная схема бортовой системы (рис. 1), а также вероятности безотказной работы блоков:

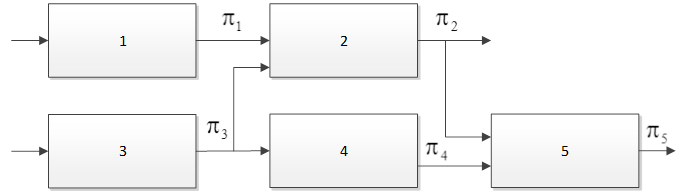


Рисунок 1 – Функциональная схема бортовой системы №2

Таблица 1 – Вероятности безотказной работы блоков

|  |  |
| --- | --- |
|  | 0.06 |
|  | 0.02 |
|  | 0.03 |
|  | 0.07 |
|  | 0.04 |

**Задание**.

1. Вычислить исходную неопределенность объекта до выполнения проверок.
2. Исследовать модель и определить результаты проверок в каждом информационном состоянии.
3. Выбрать минимально необходимый состав проверок по критерию максимума информативности.
4. Сделать выводы по полученным результатам.
5. Оформить отчет о работе.

**Использованные формулы**

1. Исходная неопределенность объекта до выполнения проверок:

- вероятность -го класса состояний объекта.

1. Вероятность исхода проверки:

– вероятность -го исхода проверки .

1. Количество информации, получаемой при выполнении конкретной проверки:

– число возможных проверок.

**Ход работы**

Определим значения для диагностических признаков по принципу: текущий и зависимые от него блоки считаются не функционирующими (0), все остальные блоки будем считать работоспособными (1) – и внесем в таблицу 2. Далее составим таблицу вероятностей отказов блоков (таб. 2) и рассчитаем вероятности нахождения БС в определенном состоянии.

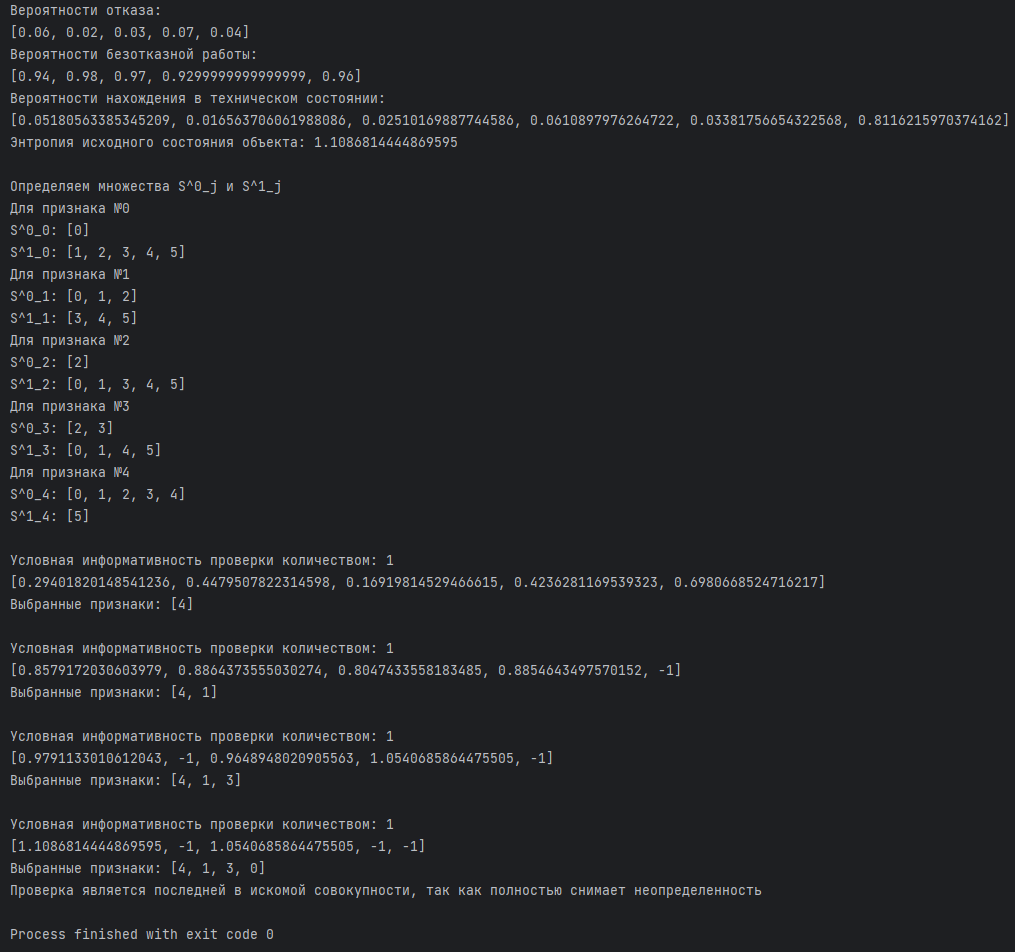
Таблица 2 – Вероятности отказа

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  | 0.06 | 0.02 | 0.03 | 0.07 | 0.04 |
|  | 0.94 | 0.98 | 0.97 | 0.93 | 0.96 |
|  | 0.0638 | 0.0204 | 0.0309 | 0.0753 | 0.0417 |

Таблица 3 – Диагностические признаки

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ТС | Диагностические признаки | | | | |  |
|  |  |  |  |  |
|  | 0 | 0 | 1 | 1 | 0 | 0.05181 |
|  | 1 | 0 | 1 | 1 | 0 | 0.01656 |
|  | 1 | 0 | 0 | 0 | 0 | 0.02510 |
|  | 1 | 1 | 1 | 0 | 0 | 0.06109 |
|  | 1 | 1 | 1 | 1 | 0 | 0.03382 |
|  | 1 | 1 | 1 | 1 | 1 |  |

Для поиска искомого множества проверок был написан скрипт Python, листинг которого представлен в приложении А. Результат выполнения представлен ниже:

Таким образом, получаем упорядоченное множество проверок, имеющих наибольшую эффективность:

**Выводы**

В ходе выполнения лабораторной работы были получены навыки в применении выбора множества проверок с максимальной информативностью. Для варианта №12 это множество оказалось:

**Приложение А. Листинг программы**

**Скрипт Python:**

import math  
import itertools  
  
  
def load\_data(file\_name):  
 with open(file\_name) as file:  
 return [float(x) for x in file.readline().rsplit()]  
  
  
def load\_table(file\_name):  
 with open(file\_name) as file:  
 numbers = []  
 for line in file:  
 numbers.append([int(x) for x in line.rsplit()])  
 return numbers  
  
  
def define\_s(table):  
 numbers = []  
 for i in range(len(table[0])):  
 temp\_line = [[], []]  
 for j in range(len(table)):  
 temp\_line[table[j][i]].append(j)  
 numbers.append(temp\_line)  
 return numbers  
  
  
def calculate\_informative(states, probs):  
 p = []  
 for i in range(len(states)):  
 p.append([sum(probs[j] for j in states[i][0]), sum(probs[j] for j in states[i][1])])  
 return [p[i][0] \* -math.log2(p[i][0]) + p[i][1] \* -math.log2(p[i][1]) for i in range(len(p))]  
  
  
def calculate\_informative\_with\_picked(states, probs, picked\_p):  
 p = []  
 for i in range(len(states)):  
 patterns = list(itertools.product([0, 1], repeat=len(picked\_p) + 1))  
 temp\_line = []  
 for pattern in patterns:  
 p\_states = list(range(len(probs)))  
 next\_p\_states = []  
 for p\_state in p\_states:  
 if p\_state in states[i][pattern[0]]:  
 next\_p\_states.append(p\_state)  
 p\_states = next\_p\_states.copy()  
 for pattern\_iter in range(len(picked\_p)):  
 next\_p\_states = []  
 for p\_state in p\_states:  
 if p\_state in states[picked\_p[pattern\_iter]][pattern[pattern\_iter + 1]]:  
 next\_p\_states.append(p\_state)  
 p\_states = next\_p\_states.copy()  
 temp\_line.append(sum([probs[x] for x in p\_states]))  
 p.append(temp\_line)  
 i\_s = []  
 for i in range(len(p)):  
 if i in picked\_p:  
 i\_s.append(-1)  
 else:  
 i\_temp = []  
 for x in p[i]:  
 if x != 0:  
 i\_temp.append(x \* -1 \* math.log2(x))  
 i\_s.append(sum(i\_temp))  
 return i\_s  
  
def informative\_step(s\_s, p\_s, picked\_s, high):  
 print("\nУсловная информативность проверки количеством: 1")  
 i = calculate\_informative\_with\_picked(s\_s, p\_s, picked\_s)  
 picked\_s.append(i.index(max(i)))  
 print(i)  
 print("Выбранные признаки:", picked\_s)  
 if max(i) == high:  
 print("Проверка является последней в искомой совокупности, так как полностью снимает неопределенность")  
 else:  
 informative\_step(s\_s, p\_s, picked\_s, high)  
  
  
  
qp = load\_data("LAB-4\_Data.txt")  
pp = [1 - x for x in qp]  
qppp = [qp[i] / pp[i] for i in range(len(qp))]  
ps = [qppp[i] / (sum(qppp) + 1) for i in range(len(qp))]  
ps.append(1 / (sum(qppp) + 1))  
print("Вероятности отказа:")  
print(qp)  
print("Вероятности безотказной работы:")  
print(pp)  
print("Вероятности нахождения в техническом состоянии:")  
print(ps)  
  
h = sum([x \* -math.log2(x) for x in ps])  
print("Энтропия исходного состояния объекта:", h)  
  
print("\nОпределяем множества S^0\_j и S^1\_j")  
t = load\_table("LAB-4\_Table.txt")  
s = define\_s(t)  
for i in range(len(s)):  
 print(f"Для признака №{i}")  
 print(f"S^0\_{i}: {s[i][0]}")  
 print(f"S^1\_{i}: {s[i][1]}")  
  
p\_test = [0.2, 0.05, 0.24, 0.15, 0.06, 0.3]  
t\_test = load\_table("test.txt")  
s\_test = define\_s(t\_test)  
  
informative\_step(s, ps, [], h)

**Файл «LAB-4\_Data.txt»:**

0.06 0.02 0.03 0.07 0.04

**Файл «LAB-4\_Table.txt»:**

0 0 1 1 0  
1 0 1 1 0  
1 0 0 0 0  
1 1 1 0 0  
1 1 1 1 0  
1 1 1 1 1