ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМ. Р.Е. АЛЕКСЕЕВА»
(НГТУ)

Институт радиоэлектроники и информационных технологий
Кафедра: Информатика и системы управления
Направление подготовки: «Информатика и вычислительная техника»

Анализ надежности с помощью деревьев отказов. Автоматизация построения дерева отказов.

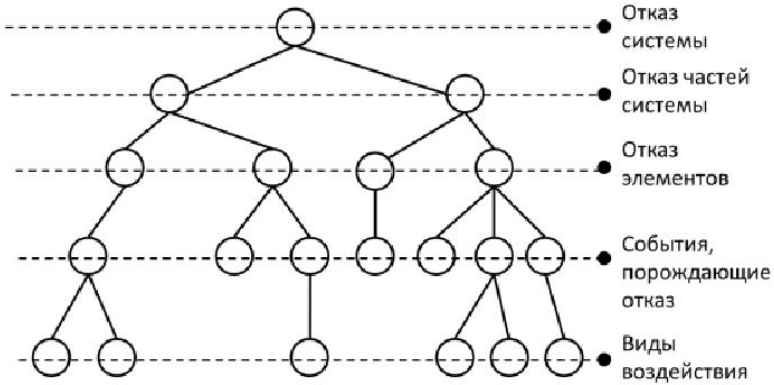
Выполнил: студент группы М22-ИВТ-4
Напылов Е.И.

Руководитель: д.т.н., профессор
Соколова Э.С.

Нижний Новгород
2022

Дерево отказов

Дерево отказов лежит в основе логико-вероятностной модели причинно-следственных связей отказов систем и состоит из последовательностей и комбинаций неисправностей.



Построение дерева отказов

Алгоритм построения “Сверху вниз”:

1. Определение конечного события, например отказ всей системы - корень дерева.
2. Определение причин или событий для определения того, что может вызвать событие-корень.
3. Этапы 1 и 2 повторяются до тех пор, пока не получаются события-листья, которые являются нерасщепляемыми.

Преимущества и недостатки

1. Универсальность - разнообразие причин различного рода.
2. Подход “сверху вниз” позволяет рассматривать только то, что влияет на конечный результат.
3. Подходит для сложных систем.
4. Показывает узкие места системы.
5. **Наглядное представление.**
6. Выявляет проблемные места.
7. Позволяет анализировать сложные логические взаимосвязи.

1. **Большие временные и денежные затраты.**
2. Элементы имеют только 2 состояния - исправное и неисправное.
3. Сложно учитывать резервирование элементов.
4. Требуются высококвалифицированные специалисты.
5. Описывает систему только в определенный момент времени.
6. Нет возможности работать с обратной связью.

Автоматизация построения - входные данные


```
G = {
    '1': ['2', '3'],
    '2': ['4'],
    '3': ['5'],
    '4': ['6', '7'],
    '5': ['7'],
    '6': ['8'],
    '7': ['8'],
    '8': []
}
```

Список смежности

```
names = {
    'system': '"Система очистки воздуха"',
    '1': '"Фильтр грубой очистки"',
    '2': '"Центробежный вентилятор №1"',
    '3': '"Центробежный вентилятор №2"',
    '4': '"Фильтр мелкой очистки №1"',
    '5': '"Фильтр мелкой очистки №2"',
    '6': '"Обеззараживатель №1"',
    '7': '"Обеззараживатель №2"',
    '8': '"Распределитель чистого воздуха"',
}
```

Названия блоков

```
reasons = {
    'r_1': ['Заблокирован', 'Пробоина'],
    'r_2': ['Отсутствие питания'],
    'r_3': ['Отсутствие питания'],
    'r_4': ['Сильно загрязнен'],
    'r_5': ['Сильно загрязнен'],
    'r_6': ['Неисправность УФ ламп', 'Отсутствие питания'],
    'r_7': ['Неисправность УФ ламп', 'Отсутствие питания'],
    'r_8': ['Пробоина'],
}
```

Причины выхода из строя

Автоматизация построения - пути распространения неисправностей


```
# рекурсивный алгоритм поиска путей
def find_paths_algorithm(v1, v2, G, paths=[], q=[], visited=set()) -> list

# обертка
def find_all_paths(v1, v2, G) -> list
```

→ [['8', '6', '4', '2', '1'], ['8', '7', '4', '2', '1'], ['8', '7', '5', '3', '1']]

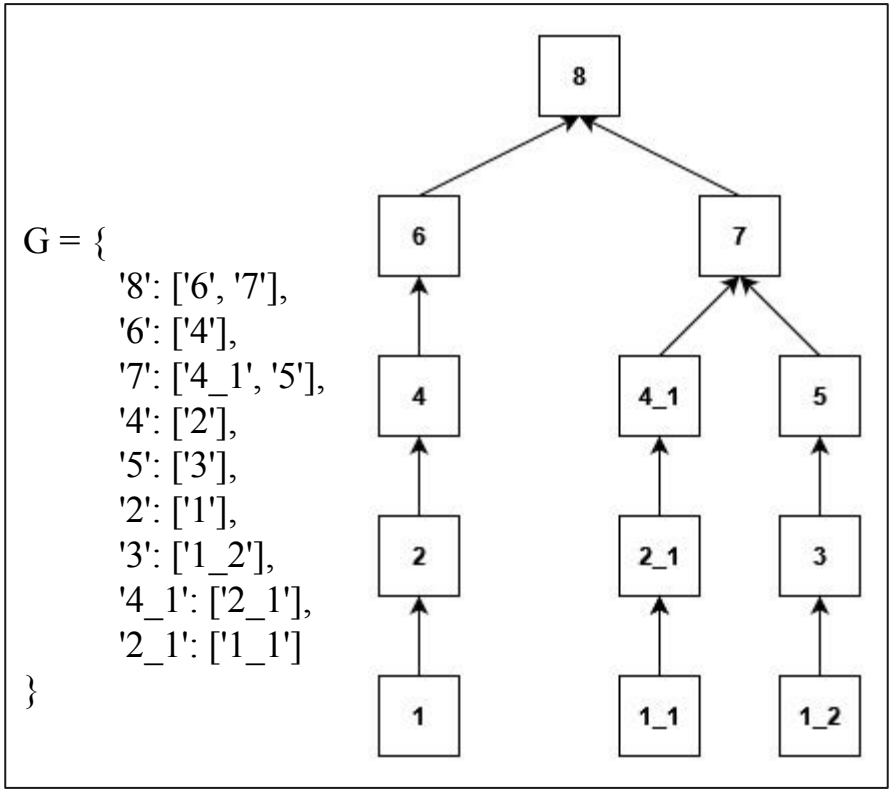
Автоматизация построения - дерево неисправностей

Список путей распространения неисправностей:
[['8', '6', '4', '2', '1'],
['8', '7', '4', '2', '1'],
['8', '7', '5', '3', '1']]

Задача - построить дерево путей.

```
# построение дерева всех путей
def merge_paths_to_tree(paths) -> dict
```

Для устранения конфликтов номеров вершин используется дополнительный индекс.



Автоматизация построения - дерево отказов и его графическое представление

Осталось соединить названия блоков и причины выхода из строя с полученным деревом. Затем визуализация.

```
# +названия, +причины, визуализация
def plot_graph(G, paths, names, reasons) -> None
```

```
// Fault Tree
digraph {
    r_1_0 [label="Заблокирован"]
    r_1_1 [label="Пробоина"]
    r_2_0 [label="Отсутствие питания"]
    r_3_0 [label="Отсутствие питания"]
    r_4_0 [label="Сильно загрязнен"]
    r_5_0 [label="Сильно загрязнен"]
    r_6_0 [label="Неисправность УФ ламп"]
    r_6_1 [label="Отсутствие питания"]
    r_7_0 [label="Неисправность УФ ламп"]
    r_7_1 [label="Отсутствие питания"]
    r_8_0 [label="Пробоина"]
    8 [label="Отказ блока \"Распределитель чистого
воздуха\" <=> Отказ системы" shape=box]
    6 [label="Отказ блока \"Обеззараживатель №1\""
shape=box]
    6 -> 8
    7 [label="Отказ блока \"Обеззараживатель №2\""
shape=box]
    7 -> 8
// ...
}
```

Визуализация выполняется с помощью Graphviz.

DOT Language

Автоматизация построения - дерево отказов и его графическое представление