**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана**
**(национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

# Отчёт по лабораторной работе №2 по дисциплине "Проектирование экспертных систем"

**Тема** Алгоритм поиска в графе И/ИЛИ в ширину от данных

**Студент** Варламова Е. А.

**Группа** ИУ7-33М

**Оценка (баллы)** _____

**Преподаватели** Русакова З.Н.

Москва — 2024 г.

# СОДЕРЖАНИЕ

# ВВЕДЕНИЕ

Цель работы – реализовать алгоритм поиска по графу в ширину от данных. Для достижения поставленной цели потребуется:

— описать используемые структуры данных;

— описать алгоритм поиска по графу в ширину от данных;

— привести реализацию алгоритма;

— привести примеры работы алгоритма.

# 1 | Используемые структуры данных

Разработаем класс вершины графа. Поля класса:

— номер вершины;

— флаг вершины – открыта/закрыта.

Разработаем класс правила. Поля класса:

— список входных вершин;

— целевая вершина;

— номер правила;

— флаг правила – открыто/закрыто.

Разработаем класс поиска. Поля класса:

— список правил;

— 2 флага: есть решение, нет решения, поставим их в 1;

— целевая вершина.

# 2 | Алгоритм поиска по графу в ширину от данных

Вход:

— доказанные вершины;

— целевая вершина.

Выход:

— информация о решении.

## 2.1 Основной метода поиска

Отметить все доказанные вершины как закрытые. Поа флаги решения истинны, выполняем:

— вызвать метод потоми, который возвращает количество закрытых правил;

— если флаг решение найдено, то выход с сообщением об успешном поиске;

— если количество закрытых правил равно 0, то выход с сообщением о неудачном поиске;

## 2.2 Метод потомки

В цикле по списку правил выполнить.

— Если правило не открыто, то пропустить его.

— Если все выходные вершины правила покрываются закрытыми вершинами, то пометить правило закрытым, пометить выходную вершину правила как закрытую. Если выходная вершина равна целевой, то изменить флаг решение найдено. Увеличить количество закрытых правил.

# 3 | Реализация

Листинг 3.1: Структуры данных

```
class Label(Enum):
    OPEN = 0
    CLOSE = 1


class Node:
    def __init__(self, number: int, flag: int = Label.OPEN):
        self.number = number
        self.flag = flag

    def __str__(self):
        res = '' + f'{self.number}'
        return res

    def __repr__(self):
        res = '' + f'{self.number}'
        return res


class Rule:
    def __init__(self, number: int, out_node: Node, node_arr: List[Node],
        label=Label.OPEN):
        self.number = number
        self.out_node = out_node
        self.node_arr = node_arr
        self.label = label
```

Листинг 3.2: Класс поиска

```

class Search:
    def __init__(self, rule_arr: [Rule]):
        self.rule_arr = rule_arr

        self.goal_node = None
        self.solution_flg = 1
        self.no_solution_flg = 1
        self.closed_nodes = []
```

```
10
11      def run(self, goal_node: Node, in_node_arr: [Node]):
12          self.goal_node = goal_node
13          self.set_nodes_closed(in_node_arr)
14          print("Start state: ")
15          self.print_closed_nodes()
16          while self.solution_flg and self.no_solution_flg:
17              rule_cnt = self.parent_search()
18
19              if self.solution_flg == 0:
20                  return
21
22              if rule_cnt == 0:
23                  self.no_solution_flg = 0
24                  print("Solution was not found")
25
26
27      def parent_search(self):
28          cnt_rules = 0
29
30          for rule in self.rule_arr:
31              if self.solution_flg:
32                  if rule.label != Label.OPEN:
33                      continue
34
35                  if self.is_close_nodes_cover(rule.node_arr):
36                      print(f'Rule {rule.number}: {rule.node_arr} -> {rule.
                            out_node}: all in-nodes are closed, added {rule.
                            out_node} to closed')
37                      rule.label = Label.CLOSE
38                      rule.out_node.flag = Label.CLOSE
39                      self.closed_nodes.append(rule.out_node)
40                      self.print_closed_nodes()
41
42                      if rule.out_node == self.goal_node:
43                          self.solution_flg = 0
44                          print(f'Rule {rule.number} has output node equal to
                                goal')
45
46                      cnt_rules += 1
47              else:
48                  break
49
50
51          print(f'Rule {rule.number} list of closed rules: ', end='')
52          self.print_closed_rules()
53          return cnt_rules
54
55      def is_close_nodes_cover(self, in_node_arr: [Node]):
56          for node in in_node_arr:
```

```
57              if node.flag != Label.CLOSE:
58                  return False
59          return True
60
61      def set_nodes_closed(self, node_arr):
62          for node in node_arr:
63              node.flag = Label.CLOSE
64              self.closed_nodes.append(node)
```

# 4 | Пример работы

Входной граф:

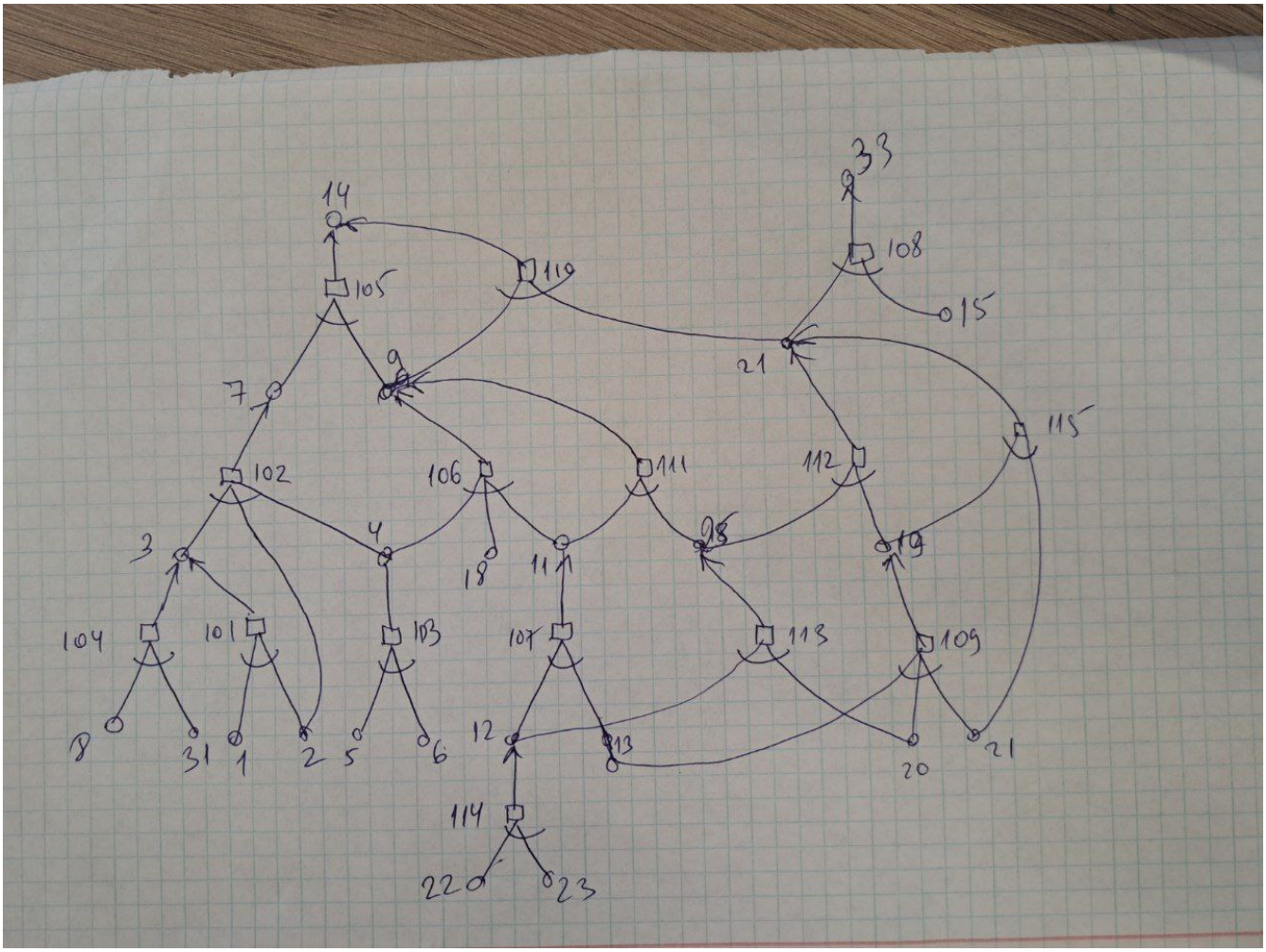


Рис. 4.1: Граф

Результат работы программы

```
========== RESTART: /Users/kate/Desktop/rep_exp_systems/lab_02/main.py =========
Start state:
Closed nodes: 5 6 2 1 18 22 23 7 13
Rule 101: [1, 2] -> 3: all in-nodes are closed, added 3 to closed
Closed nodes: 5 6 2 1 18 22 23 7 13 3
Rule 103: [5, 6] -> 4: all in-nodes are closed, added 4 to closed
Closed nodes: 5 6 2 1 18 22 23 7 13 3 4
Rule 114: [22, 23] -> 12: all in-nodes are closed, added 12 to closed
Closed nodes: 5 6 2 1 18 22 23 7 13 3 4 12
Rule 116 list of closed rules: 101 103 114
Rule 102: [3, 2, 4] -> 7: all in-nodes are closed, added 7 to closed
Closed nodes: 5 6 2 1 18 22 23 7 13 3 4 12 7
Rule 107: [12, 13] -> 11: all in-nodes are closed, added 11 to closed
Closed nodes: 5 6 2 1 18 22 23 7 13 3 4 12 7 11
Rule 116 list of closed rules: 101 102 103 107 114
Rule 106: [4, 18, 11] -> 9: all in-nodes are closed, added 9 to closed
Closed nodes: 5 6 2 1 18 22 23 7 13 3 4 12 7 11 9
Rule 116 list of closed rules: 101 102 103 106 107 114
Rule 105: [7, 9] -> 14: all in-nodes are closed, added 14 to closed
Closed nodes: 5 6 2 1 18 22 23 7 13 3 4 12 7 11 9 14
Rule 105 has output node equal to goal
Rule 106 list of closed rules: 101 102 103 105 106 107 114
Solution was found
>>>
```

Рис. 4.2: Результат работы программы

# ЗАКЛЮЧЕНИЕ

В результаты работы цель была достигнута.
Для достижения поставленной цели потребовалось:

— описать используемые структуры данных;

— описать алгоритм поиска по графу в ширину от данных;

— привести реализацию алгоритма;

— привести примеры работы алгоритма.