Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Кафедра информационных технологий и систем

**РЕШЕНИЕ ЛОГИЧЕСКИХ ЗАДАЧ МЕТОДОМ ПОИСКА НА ПРОСТРАНСТВЕ СОСТОЯНИЙ**

Лабораторная работа по дисциплине:

“Функциональное и логическое программирование”

Выполнил студент группы 8091:

\_\_\_\_\_\_\_\_\_\_ / Васильев И.В.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Принял преподаватель:

\_\_\_\_\_\_ / Михайлов Д.В.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Великий Новгород

2021

# Цель и задачи

Целью работы является овладение методологией решения логических задач с применением известных на сегодняшний день стратегий поиска в пространстве состояний.   
  
Задача №1  
Изучить на приведенном примере задачи о волке, козе и капусте работу базовой программы для решения задач методом поиска в глубину.  
  
Задача №2  
Задача о шахматном коне (задача Эйлера):

Требуется обойти все клетки шахматной доски ходом коня.

# Решение

Для решения задач была выбрана среда Visual Prolog 5.2

Задача №1  
Изучить на приведенном ниже примере задачи о волке, козе и капусте работу базовой программы для решения задач методом поиска в глубину.

Программный код представлен в файле, приложенный к архиву лабораторной работы под именем “Lab81.pro”

Результат работ выполнения программы:

Входными данными является обозначения цели поиска в разделе goal. Поиск начинается с начального состояния: initial\_state(wgc("Лодка на левом берегу.",["Волк","Коза","Капуста"],[])).

Выходные данные:



Рисунок 1 Результат выполнения программы задачи №1

Обоснование:

Работа программы начинается с вызова предиката **start()**. Последовательность переходов из начального состояния в целевое строится с помощью правила **solve\_dfs()**, а поиск достижимого состояния через правило **update()**. Определение возможности перехода делается с помощью предиката **move()**. Начальное состояние задается через факт **initial\_state()**, а конечное через **final\_state()**. С помощью правила **legal()** проверяется допустимость состояния.

Задача №2

Задача о шахматном коне (задача Эйлера):

Требуется обойти все клетки шахматной доски ходом коня.

Решение задачи представлено на языке SWI-Prolog.

Программный код представлен в файле, приложенный к архиву лабораторной работы под именем “Lab82.pl”

Обоснование:

Запуск программы осуществляется вводом ?-solve([point(3,3)], Solution).

Функция принимает список **Visted** из посещенных клеток — при старте программы этот список должен содержать одну начальную клетку. Программа работает до тех пор, пока в списке не окажется 64 клетки (если такое случилось — решение найдено и остается лишь вернуть список **Visited**

в качестве результата (**Solution**).  
  
Иначе — берется последняя посещенная клетка (**Last**), для этого выделяется первый элемент списка **Visited**([**Last**|**Other**]) и с помощью описанной выше функции move

выполняется поиск следующей возможной клетки. C помощью member(**To**, **Other**), выполняется поиск клетки, а конструкция \+ задает отрицание цели, то есть \+ member(**To**, **Other**) завершится успешно если клетка не будет найдена (не была посещена ранее). Если возможный переход найден — новая клетка добавляется в начало списка ([**To**, **Last**|**Other**]) и выполняется рекурсивный поиск начиная с новой позиции (solve([**To**, **Last**|**Other**], **Solution**)).

Результат работы программы:

В результате работы программы будет выведено множество решений в виде строки чисел. Каждая пара чисел в этой строке – это координаты.

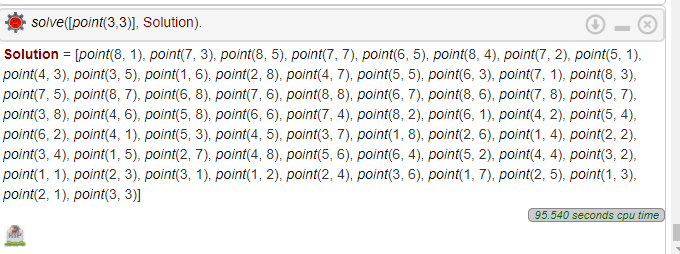


Рисунок Результат выполнения программы задачи №2

Пример строки:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 11 | 2 | 9 | 28 | 57 | 32 | 49 | **64** |
| 8 | 21 | 12 | 31 | 14 | 29 | 58 | 33 |
| 3 | 10 | **1** | 56 | 27 | 50 | 63 | 48 |
| 22 | 7 | 20 | 13 | 30 | 15 | 34 | 59 |
| 19 | 4 | 55 | 26 | 51 | 60 | 47 | 62 |
| 54 | 23 | 6 | 37 | 16 | 35 | 44 | 41 |
| 5 | 18 | 25 | 52 | 39 | 42 | 61 | 46 |
| 24 | 53 | 38 | 17 | 36 | 45 | 40 | 43 |

Таблица 1 Визуальный пример обхода шахматной доски

# Выводы по проведенным экспериментам

В ходе выполнения задач лабораторной работы, я изучил методологию решения логических задач с применением стратегий поиска в пространстве состояний.