Министерство науки и Высшего образования Российской Федерации
Севастопольский государственный университет
Кафедра ИС

Отчет
по лабораторной работе №5
«Методы поиска решений задач в пространстве состояний»
по дисциплине
«МЕТОДЫ И СРЕДСТВА ИСКУССТВЕННОГО ИНТЕЛЛЕКТА»

Выполнил студент группы ИС/б-17-2-о
Горбенко К. Н.
Проверил
Забаштанский А.К.

Севастополь
2021

1 ЦЕЛЬ РАБОТЫ

Исследование методов поиска решений задач в пространстве состояний и овладение методологией решения логических задач с применением этих методов.

2 ЗАДАНИЕ НА РАБОТУ

Задача: Путь коня. На шахматной доске NxN, из которой вырезано несколько клеток, заданы две клетки. Построить минимальный путь коня из одной клетки в другую.

Метод поиска решений: A*-алгоритм.

3 ХОД РАБОТЫ

Текст программы:

```
1     (defun a-search(start goal)
2     (setq open (cons start nil))
3     (setq closed nil)
4     (loop
5         (cond
6             ((null open)(return 'неудача))
7             (t
8                 (setq n (first open))
9                 (setq open (cdr open))
10                (setq closed (cons n closed))
11                (if (equal (first n) goal) (return 'удача))
12                (put-in-list (list-children n))
13    ;           (terpri)
14                ; (princ "closed=")
15    ;           (prin1 closed)
16    ;           (terpri)
17                ; (princ "open=")
18    ;           (prin1 open)
19            )
20        )
21    )
22 )
23
24 (defun list-children (n)
25    (setq L nil)
26    (setq a (list 2 -2))
27
28    (dolist (da a L)
```

```
29          (setq b (list 1 -1))
30          (dolist (db b)
31              (setq L (child-element (child-element L n db da) n da db))
32          )
33      )
34 )
35
36 (defun child-element (L n da db)
37      (setq a (+ (car (first n)) da))
38      (setq b (+ (cdr (first n)) db))
39
40      (if (and (>= a 1) (>= b 1) (<= a 8) (<= b 8))
41          (setq L
42              (cons
43                  (list (cons a b) (first n) (+ (third n) 1) (+ (fourth n) 1))
44                  L
45              )
46          )
47          (setq L L)
48      )
49 )
50
51 (defun exists (n da db)
52      (setq a (+ (car n) da))
53      (setq b (+ (cdr n) db))
54      (and (>= a 1) (>= b 1) (<= a 8) (<= b 8))
55 )
56
57 (defun rev(temp)
58      (setq a (first temp))
59      (setq temp (rest temp))
60      (setq b (first temp))
61      (setq temp (rest temp))
62      (setq temp (cons b (cons a temp)))
63 )
64
65 (defun put-third (str x)
66      (setq temp1 (butlast str 2))
67      (setq temp2 (last str))
68      (append temp1 (cons x temp2))
69 )
70
71 (defun put-fourth (str x)
72      (setq temp1 (butlast str 1))
73      (append temp1 (cons x nil))
74 )
75
76 (defun put-in-list (dv)
```

```
77      (cond
78          ((null dv ) nil)
79          ((and (not (member1 (caar dv) open)) (not (member1 (caar dv) closed)))
80              (setf open (add (first dv) open))
81              (put-in-list (rest dv))
82          )
83          (t
84              (setf open (del (first dv) open))
85              (setf closed (del (first dv) closed))
86              (setf open (add (first dv) open))
87              (put-in-list (rest dv))
88          )
89      )
90  )
91
92  (defun del(v l)
93      (cond
94          ((null l) nil)
95          ((and (equal (first v)(first(first l))) (<= (fourth v)(fourth (first l)
                ) ))
96              (setf l (cdr l))
97          )
98          (t
99              (append (list(car l)) (del v (cdr l)))
100         )
101     )
102 )
103 (defun add(v l)
104     (cond
105         ((null l) (cons v l))
106         ((<= (fourth v)(fourth (first l)))
107             (setf l (cons v l))
108         )
109         (t
110             (append (list(car l)) (add v (cdr l)))
111         )
112     )
113 )
114
115 (defun member1 (v l)
116     (cond
117         ((null l) nil)
118         ((equal v (caar l)) t)
119         (t (member1 v (rest l)))
120     )
121 )
122
123 (defun back-way (goal start)
```

```
124     (setq g goal)
125     (setq L nil)
126     (dolist (temp closed L)
127         (if (equal (first temp) g)
128             (prog1
129                 (setq L (cons (list (first temp) (third temp)) L))
130                 (setq g (second temp))
131             )
132         )
133     )
134 )
135
136 (print (a-search '((1 . 1) 0 0 0) '(8 . 8)))
137 (print (back-way '(8 . 8) '(1 . 1)))
```

Результат работы программы изображен на рисунке 1

```
УДАЧА
(((1 . 1) 0) ((2 . 3) 1) ((3 . 5) 2) ((5 . 6) 3) ((6 . 8) 4) ((7 . 6) 5)
 ((8 . 8) 6))
```

Рисунок 1 – Результат работы программы

ВЫВОДЫ

В ходе работы были исследованы методы поиска решений задач в пространстве состояний, а также решения логических задач с применением этих методов.