МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное автономное бюджетное образовательное учреждение высшего образования

«**Севастопольский государственный университет**»

кафедра Информационных систем

Институт информационных технологий и управления в технических системах

Лисянский Александр Игоревич

курс 4 группа ИС/б-42-о

09.03.02 Информационные системы (уровень бакалавриата)

**ОТЧЁТ**

о лабораторном практикуме №5

по дисциплине «Методы создания искусственного интеллекта»

Отметка о зачёте \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

Руководитель практикума

\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность) (подпись) (инициалы, фамилия)

Севастополь

2015

1. Цель работы

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

1. **Задание**

Требуется представить задачу, заданную вариантом задания, в пространстве состояний. Написать на языке Лисп программу, которая находит решение задачи в соответствии с заданным методом. Оценить качество работы метода перебора по критериям временной и пространственной сложности, целенаправленности поиска.

Варианты заданий приведены в таблице 5.1.

|  |  |
| --- | --- |
| Задача | Метод |
| Задача “Ханойская башня”. Имеется три стержня, на первом из которых помещены N дисков разного диаметра, причем меньший диск обязательно лежит на большем. Требуется переместить все диски на третий стержень, двигая их по очереди. Один ход состоит в снятии верхнего диска с одного из стержней и перемещении его поверх дисков (если они есть) другого стержня. Ограничение: больший диск нельзя класть на меньший | А\*-алгоритм |

1. **Код Lisp**

;;;;<1. Описание графа состояний

;;; граф представляется в виде а- списка с подсписками (p d c h) ,где

;;; p – родительская вершина

;;; d – соответствующая дочерняя вершина

;;; c – стоимость участка графа между вершинами p и d

;;; h– оценка стоимости кратчайшего пути из вершины d целевую вершину.

;;; явное описание графа состояний

(setq graph '((s d 1 10)(s c 3 7)(s b 5 4)

(s a 7 1)(a g 10 0) (b a 1 1)

(c b 1 4)(c a 2 1)(d c 1 7)))

;;; элементы списков OPEN и CLOSED будут представляться в виде подсписков

;;; ( d p g f), где d – некоторая вершина на графе состояний

;;; p – ссылка на родительскую вершину

;;; g – оценка стоимости пути из начальной вершины в вершину d

;;; f – значение оценочной функции

;;;;<2 Основная Функция поиска в соответствии с А-алгоритмом

(defun a-search(start goal)

;; поместить в open стартовую вершину, описываемую

;; подсписком со структурой ( s s 0 h)

(setq open (cons start nil))

;; инициализация closed

(setq closed nil)

(loop ; цикл

(cond ((null open)(return 'неудача)) ; на графе нет пути

(t

;; запомнить первый подсписок списка open

(setq n (first open))

;; удалить n из open

(setq open (cdr open))

;; внести n в closed

(setq closed (cons n closed))

;; проверить, является ли (first n) целевой вершиной

(if (eq (first n) goal) (return 'удача))

;; внести в open дочерние вершины и удалить

;; их при необходимости из closed

(put-in-list (list-children n))

;; печать промежуточных результатов

(terpri)

(princ "closed=")(prin1 closed)

(terpri)

(princ "open=") (prin1 open)))))

;;;;<3 Функция, возвращающая список дочерних вершин для вершины (first n)

(defun list-children (n)

(setq L nil)

(dolist (temp graph L) ; выполнить действия для каждого подсписка из graph

;; переставить первый и второй элементы temp

(setq rtemp (rev temp))

;; выяснить, содержит ли temp дочернюю вершину для (first n)

(when (eq (first n) (first temp))

; вычислить значения оценки g(n) для дочерней вершины

(setq rtemp (put-third rtemp (+ (third n)(third rtemp))))

; вычислить значения оценки f(n) для дочерней вершины

(setq rtemp (put-fourth rtemp (+ (third rtemp) (fourth rtemp))))

;добавляем описание дочерней вершины в виде подсписка rtemp

;в результирующий список L

(setq L (cons rtemp L)))))

;;;;<4 Вспомогательные функции

;;; функция, переставляющая местами первый и второй элементы в temp

(defun rev(temp)

(setq a (first temp)) (setq temp (rest temp))

(setq b (first temp)) (setq temp (rest temp))

(setq temp (cons b (cons a temp))))

;;; функция, меняющая значение третьего элемента в temp (оценка g(n)) на x

(defun put-third (str x)

(setq temp1 (butlast str 2))

(setq temp2 (last str))

(append temp1 (cons x temp2)))

;;; функция, меняющая значение четвертого элемента в temp (оценка f(n)) на x

(defun put-fourth (str x)

(setq temp1 (butlast str 1))

(append temp1 (cons x nil)))

;;;Список дочерних вершин, возвращаемый функцией LIST-CHILDREN,

;;;используется в функции PUT-IN-LIST для соответствующей модификации списков OPEN

;;;и CLOSED

(defun put-in-list (dv) ; dv – а-список дочерних вершин

(cond

;; условие завершения рекурсии

((null dv ) nil)

;; если очередная дочерняя вершина не входит в

;; списки OPEN и CLOSED, то добавить её в список OPEN

((and (not (member1 (caar dv) open))

(not (member1 (caar dv) closed)))

(setf open (add (first dv) open))

(put-in-list (rest dv)))

;; если очередная дочерняя вершина входит в

;; один из списков OPEN или CLOSED, то удалить её из

;; этих списков и заново внести в список OPEN с учетом значения

;; оценочной функции

(t (setf open (del (first dv) open))

(setf closed (del (first dv) closed))

(setf open (add (first dv) open))

(put-in-list (rest dv)))))

;;; Функция DEL(V L) выполняет удаление подсписка V, представляющего вершину графа

;;;состояний, из списка L. Удаление выполняется, если оценочная функция для

;;;удаляемой вершины V меньше, либо равна значению оценочной функции вершины из

;;; списка L

(defun del(v l)

(cond ((null l) nil) ; завершение рекурсии 67

;; проверка вхождения v в l

((and (eq (first v)(first(first l)))

;; сравнение оценочных функций

(<= (fourth v)(fourth (first l))))

(setf l (cdr l)))

(t (append (list(car l)) (del v (cdr l))))))

;;;Вставка дочерних вершин в упорядоченный список OPEN

(defun add(v l)

(cond ((null l) (cons v l)) ; выход из рекурсии

;;сравнить значения оценочных функций

((<= (fourth v)(fourth (first l)))

;;если меньше, то вставить вершину v в l

(setf l (cons v l)))

(t (append (list(car l)) (add v (cdr l))))))

;;; Функция MEMBER1(V L)

(defun member1 (v l)

(cond ((null l) nil)

((eq v (first(first l))) t)

(t (member1 v (rest l)))))

;;;функция, восстанавливающая по списку closed искомый путь

(defun back-way (goal start)

(setq g goal)

(setq L nil)

(dolist (temp closed L)

(if (eq (first temp)g)

(prog1 (setq L (cons (rev temp) L))

(setq g (second temp))))))

1. **Результат выполнения**
2. **Вывод**

В ходе работы были изучены способы организации простых баз данных с помощью А-списков и списков свойств, получены практических навыки использования и разработки функций высшего порядка, изучение средств файлового ввода-вывода в языке Лисп.