Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Институт/Факультет | Институт информационных технологий и анализа | | | |
|  |  | наименование | |  |
| данных | |  | |  |
| Кафедра/Структурное подразделение | | | Автоматизированные системы | |
|  | | наименование (при наличии) | | |
| обработки информации и управления | | | | |

**ОТЧЕТ**

по лабораторной работе № \_2\_\_\_

по дисциплине \_\_Системы искусственного интеллекта\_\_\_\_

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил | АСУб-20-2 |  |  |  | Арбакова А.В. |
|  | шифр группы |  | подпись |  | Фамилия И.О. |
| Проверил |  |  |  |  | Столбов А.Б. |
|  | должность |  | подпись |  | Фамилия И.О. |

Иркутск – 20 \_23\_

**Лабораторная работа №4: Реализация поиска в пространстве состояний**

**Цель работы:** Формирование умения реализации в среде CLIPS задачи поиска в пространстве состояний и освоение способов анализа ее решения.

**Задание:**

Разработать систему для решения головоломки о фермере.

**Решение**:

CLIPS-программа:

(defmodule MAIN

(export deftemplate ?ALL))

(deftemplate MAIN::node

(slot farmer-loc (type SYMBOL) (allowed-symbols shore-1 shore-2))

(slot fox-loc (type SYMBOL) (allowed-symbols shore-1 shore-2))

(slot goat-loc (type SYMBOL) (allowed-symbols shore-1 shore-2))

(slot cab-loc (type SYMBOL) (allowed-symbols shore-1 shore-2))

(slot s-depth (type INTEGER) (range 1 ?VARIABLE))

(slot parent (type FACT-ADDRESS SYMBOL) (allowed-symbols no-parent))

(slot last-move (type SYMBOL) (allowed-symbols alone goat fox cab no-move)))

(deffacts MAIN::init

(node

(s-depth 1)

(parent no-parent)

(farmer-loc shore-1)

(fox-loc shore-1)

(goat-loc shore-1)

(last-move no-move)))

(deffacts opposites

(opposite-of shore-1 shore-2)

(opposite-of shore-2 shore-1))

;---------------------------------------------------;

(defrule MAIN::move-alone

?nd<-(node

(s-depth ?num)

(farmer-loc ?ps))

(opposite-of ?ps ?ns)

=>

(duplicate ?nd

(s-depth(+ 1 ?num))

(farmer-loc ?ns)

(parent ?nd)

(last-move alone)))

;---------------------------------------------------;

(defrule MAIN::move-fox

?nd<-(node

(s-depth ?num)

(fox-loc ?ps)

(farmer-loc ?ps))

(opposite-of ?ps ?ns)

=>

(duplicate ?nd

(s-depth(+ 1 ?num))

(farmer-loc ?ns)

(fox-loc ?ns)

(parent ?nd)

(last-move fox)))

;---------------------------------------------------;

(defrule MAIN::move-goat

?nd<-(node

(s-depth ?num)

(goat-loc ?ps)

(farmer-loc ?ps))

(opposite-of ?ps ?ns)

=>

(duplicate ?nd

(s-depth(+ 1 ?num))

(farmer-loc ?ns)

(goat-loc ?ns)

(parent ?nd)

(last-move goat)))

;---------------------------------------------------;

(defrule MAIN::move-cab

?nd<-(node

(s-depth ?num)

(cab-loc ?ps)

(farmer-loc ?ps))

(opposite-of ?ps ?ns)

=>

(duplicate ?nd

(s-depth(+ 1 ?num))

(farmer-loc ?ns)

(cab-loc ?ns)

(parent ?nd)

(last-move cab)))

;---------------------------------------------------;

;---------------------------------------------------;

;---------------------------------------------------;

(defmodule CONSTRAINS

(import MAIN deftemplate ?ALL))

;---------------------------------------------------;

(defrule CONSTRAINS::goat-eaten

(declare (auto-focus TRUE))

?nd<-(node

(farmer-loc ?s1)

(fox-loc ?s2&~?s1)

(goat-loc ?s2))

=>

(retract ?nd))

;---------------------------------------------------;

(defrule CONSTRAINS::cab-eaten

(declare (auto-focus TRUE))

?nd<-(node

(farmer-loc ?s1)

(cab-loc ?s2&~?s1)

(goat-loc ?s2))

=>

(retract ?nd))

;---------------------------------------------------;

(defrule CONSTRAINS::loop

(declare (auto-focus TRUE))

(node

(farmer-loc ?fl)

(fox-loc ?fol)

(goat-loc ?gl)

(cab-loc ?cl)

(s-depth ?d))

?nd<-(node

(farmer-loc ?fl)

(fox-loc ?fol)

(goat-loc ?gl)

(cab-loc ?cl)

(s-depth ?d2&:(< ?d ?d2)))

=>

(retract ?nd))

;---------------------------------------------------;

;---------------------------------------------------;

;---------------------------------------------------;

(defmodule SOLUTION

(import MAIN deftemplate node))

(deftemplate SOLUTION::moves

(slot id (type FACT-ADDRESS SYMBOL) (allowed-symbols no-parent))

(multislot moves-list (type SYMBOL) (allowed-symbols no-move alone fox goat cabbage)))

;---------------------------------------------------;

(defrule SOLUTION::goal-test

(declare (auto-focus TRUE))

?nd <- (node (parent ?parent)

(farmer-loc shore-2)

(fox-loc shore-2)

(goat-loc shore-2)

(cab-loc shore-2)

(last-move ?move))

=>

(printout t "Solution found (see facts window)"))

;(assert (moves (id ?parent) (moves-list ?move))))

;---------------------------------------------------;