МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ИНСТИТУТ НЕПРЕРЫВНОГО И ДИСТАНЦИОННОГО ОБРАЗОВАНИЯ

|  |
| --- |
| КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ |

ОЦЕНКА

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| д-р техн. наук, профессор |  |  |  | С.И. Колесникова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1 |
| Моделирование принятия решения в многокритериальной задаче выбора |
| по дисциплине: Компьютерное моделирование |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | Z1431 |  |  |  | М.Д. Быстров |
|  | номер группы |  | подпись, дата |  | инициалы, фамилия |
| Студенческий билет № | 2021/3572 | |  |  |  |

Санкт-Петербург 2024

**ЦЕЛЬ РАБОТЫ**

Цель настоящей работы – знакомство с математическим аппаратом СППР для моделирования слабоструктурированных задач.

**ХОД РАБОТЫ**

Нужно произвести выбор секретаря референта из подавших резюме. Отбор претендентов происходит по трем критериям:

С1. Филологическое образование и знание предметной области.

С2. Знание английского языка.

С3. Знание компьютера.

Собеседование прошли три претендента: П1, П2, П3.

После собеседования получились следующие описания претендентов.

П1: отличное знание английского языка; нет навыков работы на компьютере, посредственное знание предметной области.

П2: незнание английского языка, нет навыков работы на компьютере, предметную область знает посредственно.

П3: очень хорошее знание предметной области и филологическое образование, хорошие навыки работы на компьютере, посредственное знание английского языка.

1) На основе метода AHP выбрать претендента, в зависимости от разных наборов «весов» критериев:

а) С1=0,4; С2=0,2; С3=0,3

б) С1=0,3; С2=0,3; С3=0,4

в) С1=0,2; С2=0,5; С3=0,3

2) На основе метода AHP+ выбрать претендента, в зависимости от разных наборов «весов» критериев, в зависимости от нового добавленного в группу претендента П4={знает делопроизводство, навыки работы на компьютере, слабое знание английского языка}.

Разработать программу, моделирующую принятие решение о выборе претендента в зависимости от «стоимости» критериев по двум методам.

1. На первом этапе реализованы алгоритмы МАИ и ММАИ в программной среде Matlab. Исходные коды реализаций представлены в Приложении 1.
2. На втором этапе составлены МПС (матрицы парных сравнений) для сравнения всех альтернатив (претендентов) для каждого критерия.

С1. Филологическое образование и знание предметной области

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | П1 | П2 | П3 | П4 |
| П1 | 1 | 1 | 1/9 | 1/3 |
| П2 | 1 | 1 | 1/9 | 1/3 |
| П3 | 9 | 9 | 1 | 7 |
| П4 | 3 | 3 | 1/7 | 1 |

С2. Знание английского языка

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | П1 | П2 | П3 | П4 |
| П1 | 1 | 9 | 5 | 7 |
| П2 | 1/9 | 1 | 1/5 | 1/3 |
| П3 | 1/5 | 5 | 1 | 3 |
| П4 | 1/7 | 3 | 1/3 | 1 |

С3. Знание компьютера

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | П1 | П2 | П3 | П4 |
| П1 | 1 | 1 | 1/7 | 1/5 |
| П2 | 1 | 1 | 1/7 | 1/5 |
| П3 | 7 | 7 | 1 | 3 |
| П4 | 5 | 5 | 1/3 | 1 |

1. Полученные значения использованы для задания входных данных алгоритмам МАИ и ММАИ. В результате комбинирования различных вариантов «весов» и расширения перечня альтернатив количество вариантов входных данных составило 9: 3 - для выбора претендента алгоритмом МАИ, 6 – для выбора алгоритмом ММАИ.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №№ | Алгоритм | «Веса» критериев | Альтернативы | Результат |
| 1 | МАИ | С1=0,4; С2=0,2; С3=0,3 | П1, П2, П3 | наилучшая альтернатива:  3  результат:  0.6708  оценки всех альтернатив:  0.2381 0.0910 0.6708 |
| 2 | МАИ | С1=0,3; С2=0,3; С3=0,4 | П1, П2, П3 | наилучшая альтернатива:  3  результат:  0.6213  оценки всех альтернатив:  0.2886 0.0901 0.6213 |
| 3 | МАИ | С1=0,2; С2=0,5; С3=0,3 | П1, П2, П3 | наилучшая альтернатива:  3  результат:  0.5048  оценки всех альтернатив:  0.4130 0.0821 0.5048 |
| 4 | ММАИ | С1=0,4; С2=0,2; С3=0,3 | П1, П2, П3 | наилучшая альтернатива:  3  результат:  0.4683  оценки всех альтернатив:  0.3002 0.2314 0.4683 |
| 5 | ММАИ | С1=0,3; С2=0,3; С3=0,4 | П1, П2, П3 | наилучшая альтернатива:  3  результат:  0.4539  оценки всех альтернатив:  0.3195 0.2266 0.4539 |
| 6 | ММАИ | С1=0,2; С2=0,5; С3=0,3 | П1, П2, П3 | наилучшая альтернатива:  3  результат:  0.4199  оценки всех альтернатив:  0.3675 0.2127 0.4199 |
| 7 | ММАИ | С1=0,4; С2=0,2; С3=0,3 | П1, П2, П3, П4 | наилучшая альтернатива:  3  результат:  0.3596  оценки всех альтернатив:  0.2152 0.1605 0.3596 0.2647 |
| 8 | ММАИ | С1=0,3; С2=0,3; С3=0,4 | П1, П2, П3, П4 | наилучшая альтернатива:  3  результат:  0.3491  оценки всех альтернатив:  0.2309 0.1570 0.3491 0.2629 |
| 9 | ММАИ | С1=0,2; С2=0,5; С3=0,3 | П1, П2, П3, П4 | наилучшая альтернатива:  3  результат:  0.3287  оценки всех альтернатив:  0.2738 0.1506 0.3287 0.2470 |

Вывод программы расчета:

>> lab1

алгоритм

{'МАИ'}

веса критериев

0.4000 0.2000 0.3000

МПС критериев

1

1.0000 1.0000 0.1111

1.0000 1.0000 0.1111

9.0000 9.0000 1.0000

2

1.0000 9.0000 5.0000

0.1111 1.0000 0.2000

0.2000 5.0000 1.0000

3

1.0000 1.0000 0.1429

1.0000 1.0000 0.1429

7.0000 7.0000 1.0000

наилучшая альтернатива:

3

результат:

0.6708

оценки всех альтернатив:

0.2381 0.0910 0.6708

алгоритм

{'МАИ'}

веса критериев

0.3000 0.3000 0.4000

МПС критериев

1

1.0000 1.0000 0.1111

1.0000 1.0000 0.1111

9.0000 9.0000 1.0000

2

1.0000 9.0000 5.0000

0.1111 1.0000 0.2000

0.2000 5.0000 1.0000

3

1.0000 1.0000 0.1429

1.0000 1.0000 0.1429

7.0000 7.0000 1.0000

наилучшая альтернатива:

3

результат:

0.6213

оценки всех альтернатив:

0.2886 0.0901 0.6213

алгоритм

{'МАИ'}

веса критериев

0.2000 0.5000 0.3000

МПС критериев

1

1.0000 1.0000 0.1111

1.0000 1.0000 0.1111

9.0000 9.0000 1.0000

2

1.0000 9.0000 5.0000

0.1111 1.0000 0.2000

0.2000 5.0000 1.0000

3

1.0000 1.0000 0.1429

1.0000 1.0000 0.1429

7.0000 7.0000 1.0000

наилучшая альтернатива:

3

результат:

0.5048

оценки всех альтернатив:

0.4130 0.0821 0.5048

алгоритм

{'ММАИ'}

веса критериев

0.4000 0.2000 0.3000

МПС критериев

1

1.0000 1.0000 0.1111

1.0000 1.0000 0.1111

9.0000 9.0000 1.0000

2

1.0000 9.0000 5.0000

0.1111 1.0000 0.2000

0.2000 5.0000 1.0000

3

1.0000 1.0000 0.1429

1.0000 1.0000 0.1429

7.0000 7.0000 1.0000

наилучшая альтернатива:

3

результат:

0.4683

оценки всех альтернатив:

0.3002 0.2314 0.4683

алгоритм

{'ММАИ'}

веса критериев

0.3000 0.3000 0.4000

МПС критериев

1

1.0000 1.0000 0.1111

1.0000 1.0000 0.1111

9.0000 9.0000 1.0000

2

1.0000 9.0000 5.0000

0.1111 1.0000 0.2000

0.2000 5.0000 1.0000

3

1.0000 1.0000 0.1429

1.0000 1.0000 0.1429

7.0000 7.0000 1.0000

наилучшая альтернатива:

3

результат:

0.4539

оценки всех альтернатив:

0.3195 0.2266 0.4539

алгоритм

{'ММАИ'}

веса критериев

0.2000 0.5000 0.3000

МПС критериев

1

1.0000 1.0000 0.1111

1.0000 1.0000 0.1111

9.0000 9.0000 1.0000

2

1.0000 9.0000 5.0000

0.1111 1.0000 0.2000

0.2000 5.0000 1.0000

3

1.0000 1.0000 0.1429

1.0000 1.0000 0.1429

7.0000 7.0000 1.0000

наилучшая альтернатива:

3

результат:

0.4199

оценки всех альтернатив:

0.3675 0.2127 0.4199

алгоритм

{'ММАИ'}

веса критериев

0.4000 0.2000 0.3000

МПС критериев

1

1.0000 1.0000 0.1111 0.3333

1.0000 1.0000 0.1111 0.3333

9.0000 9.0000 1.0000 7.0000

3.0000 3.0000 0.1429 1.0000

2

1.0000 9.0000 5.0000 7.0000

0.1111 1.0000 0.2000 0.3333

0.2000 5.0000 1.0000 3.0000

0.1429 3.0000 0.3333 1.0000

3

1.0000 1.0000 0.1429 0.2000

1.0000 1.0000 0.1429 0.2000

7.0000 7.0000 1.0000 3.0000

5.0000 5.0000 0.3333 1.0000

наилучшая альтернатива:

3

результат:

0.3596

оценки всех альтернатив:

0.2152 0.1605 0.3596 0.2647

алгоритм

{'ММАИ'}

веса критериев

0.3000 0.3000 0.4000

МПС критериев

1

1.0000 1.0000 0.1111 0.3333

1.0000 1.0000 0.1111 0.3333

9.0000 9.0000 1.0000 7.0000

3.0000 3.0000 0.1429 1.0000

2

1.0000 9.0000 5.0000 7.0000

0.1111 1.0000 0.2000 0.3333

0.2000 5.0000 1.0000 3.0000

0.1429 3.0000 0.3333 1.0000

3

1.0000 1.0000 0.1429 0.2000

1.0000 1.0000 0.1429 0.2000

7.0000 7.0000 1.0000 3.0000

5.0000 5.0000 0.3333 1.0000

наилучшая альтернатива:

3

результат:

0.3491

оценки всех альтернатив:

0.2309 0.1570 0.3491 0.2629

алгоритм

{'ММАИ'}

веса критериев

0.2000 0.5000 0.3000

МПС критериев

1

1.0000 1.0000 0.1111 0.3333

1.0000 1.0000 0.1111 0.3333

9.0000 9.0000 1.0000 7.0000

3.0000 3.0000 0.1429 1.0000

2

1.0000 9.0000 5.0000 7.0000

0.1111 1.0000 0.2000 0.3333

0.2000 5.0000 1.0000 3.0000

0.1429 3.0000 0.3333 1.0000

3

1.0000 1.0000 0.1429 0.2000

1.0000 1.0000 0.1429 0.2000

7.0000 7.0000 1.0000 3.0000

5.0000 5.0000 0.3333 1.0000

наилучшая альтернатива:

3

результат:

0.3287

оценки всех альтернатив:

0.2738 0.1506 0.3287 0.2470

>>

**ХОД РАБОТЫ**

В ходе выполнения первой лабораторной работы №1 была написана программа, позволяющая решать слабоструктурированные задачи выбора с помощью метода анализа иерархий (МАИ), а также с помощью его модифицированной версии (ММАИ). Программа написана в среде MATLAB (R2023b).

Решен вариант слабоструктурированной задачи с использованием разных методов, «весов» критериев, набора альтернатив. В итоговых оценках альтернатив видна разница при использовании одинаковых входных данных для разных вариантов алгоритмов. При выполнении работы оба варианта алгоритма всегда выбирали одну и ту же альтернативу, однако разница в оценивании видна при рассмотрении наборов итоговых оценок альтернатив.

Приложение 1 Исходный код программы

1. lab1.m

% "веса" критериев

criteriasWeights = [

0.4, 0.2, 0.3;

0.3, 0.3, 0.4;

0.2, 0.5, 0.3];

% МПС альтернатив по критериям

criteriaMps = containers.Map('KeyType','int32','ValueType','any');

criteriaMps(1) = [

1, 1, 1/9, 1/3;

1, 1, 1/9, 1/3;

9, 9, 1, 7;

3, 3, 1/7, 1

];

criteriaMps(2) = [

1, 9, 5, 7;

1/9, 1, 1/5, 1/3;

1/5, 5, 1, 3;

1/7, 3, 1/3, 1

];

criteriaMps(3) = [

1, 1, 1/7, 1/5;

1, 1, 1/7, 1/5;

7, 7, 1, 3;

5, 5, 1/3, 1

];

% расчет решений

compare(1, criteriaMps, criteriasWeights(1, :), 3);

compare(1, criteriaMps, criteriasWeights(2, :), 3);

compare(1, criteriaMps, criteriasWeights(3, :), 3);

compare(2, criteriaMps, criteriasWeights(1, :), 3);

compare(2, criteriaMps, criteriasWeights(2, :), 3);

compare(2, criteriaMps, criteriasWeights(3, :), 3);

compare(2, criteriaMps, criteriasWeights(1, :), 4);

compare(2, criteriaMps, criteriasWeights(2, :), 4);

compare(2, criteriaMps, criteriasWeights(3, :), 4);

**function** [result] = **compare**(alg, criteriaMps, criteriaWeights, alternativeNum)

% выполнить поиск наилучшей альтернативы

% alg - используемый алгоритм - 1 - МАИ, 2 - ММАИ

% criteriaMps - containers.Map (№ критерия -> МПС[№ альтернативы, № альтернативы])

% criteriaWeights - веса критериев

% alternateNum - кол-во альтернатив

criteriaNum = size(criteriaWeights, 2);

pairComparisonMatrix = zeros(criteriaNum, alternativeNum, alternativeNum);

% заполнение трехмерной матрицы МПС [критерий - альтернатива - альтернатива]

% -> оценка

**for** criteria = 1:criteriaNum

mps = criteriaMps(criteria);

mps = mps(1:alternativeNum, 1:alternativeNum);

pairComparisonMatrix(criteria, :, :) = mps;

**end**

% запуск работы алгоритма

**switch** (alg)

**case** 1

[solution, score, scores] = ahp(criteriaWeights, pairComparisonMatrix);

**case** 2

[solution, score, scores] = ahpPlus(criteriaWeights, pairComparisonMatrix);

**otherwise**

error('wrong algorithm number');

**end**

algs = {'МАИ', 'ММАИ'};

% вывод результатов

disp('алгоритм');

disp(algs(alg));

disp('веса критериев');

disp(criteriaWeights);

disp('МПС критериев');

**for** criteria = 1:criteriaNum

disp(criteria);

disp(permute(pairComparisonMatrix(criteria, :, :), [2,3,1]));

**end**

disp('наилучшая альтернатива:');

disp(solution);

disp('результат:');

disp(score);

disp('оценки всех альтернатив:');

disp(scores);

result = solution;

**end**

1. ahp.m

**function** [solution, score, alternativeScores] = **ahp**(criteriaWeights, pairComparisonMatrix)

% analytic hierarchy process

% criteriaWeights: array of the criterias' weights -> array[criteriaNum] =

% criteria weight

% pairComparisonMatrix: matrix[criteria, alternative, alternative] = rate

% returns - number of selected alternative, score, result score vector

% get numbers of criterias and alternatives

criteriaNum = size(criteriaWeights, 2);

alternativeNum = size(pairComparisonMatrix, 2);

% array[criteria] = array[alternativeNum] = counted weight (rows - w^i)

criteriaAlternativeWeightVectorArray = zeros(criteriaNum, alternativeNum);

% normalize weights vector

criteriaWeights = criteriaWeights / sum(criteriaWeights);

% iterate on criterias

**for** criteria = 1:criteriaNum

% matrix normalization

**for** alternative = 1:alternativeNum

s = 0;

**for** row = 1:alternativeNum

s = s + pairComparisonMatrix(criteria, row, alternative);

**end**

**for** row = 1:alternativeNum

pairComparisonMatrix(criteria, row, alternative) = pairComparisonMatrix(criteria, row, alternative) / s;

**end**

**end**

criteriaWeight = criteriaWeights(criteria);

%find score by criteria for alternatives

**for** alternative = 1:alternativeNum

alternativeVector = pairComparisonMatrix(criteria, alternative, :);

score = mean(alternativeVector);

weightedCriteria = score \* criteriaWeight;

criteriaAlternativeWeightVectorArray(criteria, alternative) = weightedCriteria;

**end**

**end**

alternativeScores = zeros(1, alternativeNum);

**for** alternative = 1:alternativeNum

alternativeScores(alternative) = sum(criteriaAlternativeWeightVectorArray( ...

1:criteriaNum, ...

alternative));

**end**

%disp(alternativeScores);

[score, solution] = max(alternativeScores);

**end**

1. ahpPlus.m

**function** [solution, score, alternativeScores] = **ahpPlus**(criteriaWeights, pairComparisonMatrix)

% analytic hierarchy process - modified

% criteriaWeights: array of the criterias' weights -> array[criteriaNum] =

% criteria weight

% pairComparisonMatrix: matrix[criteria, alternative, alternative] = rate

% returns - number of selected alternative, score, result score vector

% get numbers of criterias and alternatives

criteriaNum = size(criteriaWeights, 2);

alternativeNum = size(pairComparisonMatrix, 2);

% array[criteria] = array[alternativeNum] = counted weight (rows - w^i)

criteriaAlternativeScoreMatrix = zeros(criteriaNum, alternativeNum);

% normalize weights vector

criteriaWeights = criteriaWeights / sum(criteriaWeights);

% STAGE 1 - iterate on criterias and calculate alternative rates

**for** criteria = 1:criteriaNum

% matrix normalization

**for** alternative = 1:alternativeNum

s = 0;

**for** row = 1:alternativeNum

s = s + pairComparisonMatrix(criteria, row, alternative);

**end**

**for** row = 1:alternativeNum

pairComparisonMatrix(criteria, row, alternative) = pairComparisonMatrix(criteria, row, alternative) / s;

**end**

**end**

%find score by criteria for alternatives

**for** alternative = 1:alternativeNum

alternativeVector = pairComparisonMatrix(criteria, alternative, :);

score = mean(alternativeVector);

criteriaAlternativeScoreMatrix(criteria, alternative) = score;

**end**

**end**

% STAGE 2 - create b-matrixes for criterias

criteriaBMatrixes = containers.Map('KeyType','int32','ValueType','any');

**for** criteria = 1:criteriaNum

bMatrix = zeros(alternativeNum, alternativeNum, 2);

**for** alternative1 = 1:alternativeNum

**for** alternative2 = 1:alternativeNum

score1 = criteriaAlternativeScoreMatrix(criteria, alternative1);

score2 = criteriaAlternativeScoreMatrix(criteria, alternative2);

s = score1 + score2;

normScore1 = score1 / s;

normScore2 = score2 / s;

bMatrix(alternative1, alternative2, 1) = normScore1;

bMatrix(alternative1, alternative2, 2) = normScore2;

**end**

**end**

criteriaBMatrixes(criteria) = bMatrix;

**end**

% STAGE 3 - create common W-matrix

wMatrix = zeros(alternativeNum, alternativeNum, 2);

**for** alternative1 = 1:alternativeNum

**for** alternative2 = 1:alternativeNum

sum1 = 0;

sum2 = 0;

**for** criteria = 1:criteriaNum

bMatrix = criteriaBMatrixes(criteria);

alternateScore1 = bMatrix(alternative1, alternative2, 1);

alternateScore2 = bMatrix(alternative1, alternative2, 2);

criteriaWeight = criteriaWeights(criteria);

sum1 = sum1 + criteriaWeight \* alternateScore1;

sum2 = sum2 + criteriaWeight \* alternateScore2;

**end**

wMatrix(alternative1, alternative2, 1) = sum1;

wMatrix(alternative1, alternative2, 2) = sum2;

**end**

**end**

% STAGE 4 - count global alternative scores

alternativeScores = zeros(1, alternativeNum);

scoreSum = 0;

**for** alternative1 = 1:alternativeNum

s = 0;

**for** alternative2 = 1:alternativeNum

s = s + wMatrix(alternative1, alternative2, 1);

**end**

alternativeScores(alternative1) = s;

scoreSum = scoreSum + s;

**end**

alternativeScores = alternativeScores / scoreSum;

[score, solution] = max(alternativeScores);

**end**