Федеральное государственное образовательное бюджетное   
учреждение высшего образования  
“ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ  
РОССИЙСКОЙ ФЕДЕРАЦИИ”  
 (Финансовый университет)

**Факультет**  
**информационных технологий и анализа больших данных**  
**Кафедра “Бизнес-информатики”**

Расчётно-аналитическая работа

по курсу «Математические методы принятия решений»

**Выполнили:**   
Студенты группы БИ20-4  
Матвеев Артем Александрович

**Научный руководитель/Проверил**   
Старший преподаватель

Департамента анализа данных и машинного обучения   
Аксенов Дмитрий Андреевич

Москва 2022

**Содержание**

1. Условие первой задачи (физическая модель)4

1.1. Алгоритм решения задач линейного программирования5

1.2. Идея и описание алгоритма6

1.3. Математическое решение алгоритма8

1.3.1. Реализация алгоритма на Python8

1.3.2. Реализация алгоритма в Excel10

1.4. Вывод 12

1.5. Перспективы и варианты использования12

2. Условие второй задачи (физическая модель)13

2.1. Алгоритм Дейкстра13

2.1.1. Идея и описание алгоритма13

2.1.2. Математическое решение алгоритма Дейкстра14

2.1.3. Реализация алгоритма на Python15

2.2. Алгоритм Флойда18

2.2.1. Идея и описание алгоритма18

2.2.2. Математическое решение алгоритма Флойда18

2.2.3. Реализация алгоритма на Python19

2.3. Идея и описание алгоритма поиска решения в Excel 20

2.3.1. Описание алгоритма поиска решения21

2.4. Варианты использования системы24

2.4.1. Первый вариант использования24

2.4.2. Второй вариант использования25

2.5. Тестирование26

2.6. Заключение27

3. Условие третьей задачи (физическая модель)28

3.1. Математическая модель30

3.2. Игры с «Природой»30

3.3. Формулировка задачи33

3.4. Ввод данных пользователем34

3.5. Критерий Байеса36

3.5.1. Реализация на питоне36

3.6. Критерий Лапласа37

3.6.1. Реализация на питоне38

3.7. Критерий Гермейера39

3.7.1. Реализация на питоне40

3.8. Критерий оптимизма41

3.8.1. Реализация на питоне41

3.9. Критерий пессимизма42

3.9.1. Реализация на питоне42

3.10. Критерий Вальда43

3.10.1. Реализация на питоне44

3.11. Критерий Сэвиджа45

3.11.1 Реализация на питоне45

3.12. Критерий Гурвица46

3.12.1. Реализация на питоне47

3.13. Антагонистическая игра48

3.13.1. Решение антагонистической игры на питоне51

3.13.2. Решение антагонистической задачи в Excel53

3.14. Биматричный алгоритм56

3.14.1. Решение биматричной задачи в Excel56

3.15. Тестирование58

3.16. Вариант использования59

3.17. Заключение60

4. Список используемых источников61

**1. Условие первой задачи (физическая модель)**

В нашей задаче имеются 2 пункта отправления грузов. Обозначим их как а1 и а2 соответственно. Также в условии указаны объемы отправления по каждому пункту (запасы) в количестве 60, 80. Нам известна потребность в грузах в количестве 35, 50, 55 по каждому из 3 пунктов назначения. Пункты назначения мы также обозначим b1, b2 и b3 соответственно. Задана матрица стоимостей доставки по каждому варианту, она представлена ниже:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поставщик/ потребитель | b1 | b2 | b3 | Запасы |
| a1 | 1 | 2 | 3 | 60 |
| a2 | 4 | 3 | 2 | 80 |
| Потребность | 35 | 50 | 55 | **140** |

*Матрица 1. Стоимость доставки по каждому варианту*

Необходимо рассчитать оптимальный план перевозок, то есть определить сколько груза должно быть отправлено из каждого i-го пункта отправления в каждый j-ый пункт назначения с минимальными транспортными издержками.

Эту задачу нам удалось решить двумя способами. Для решения этой задачи мы будем реализовывать алгоритм в Excel, а также в Python, так как это позволит эффективно найти оптимальное решение.

**1.1. Алгоритм решения задач линейного программирования**

Практически все транспортные задачи имеют единую математическую модель. Классический вариант решения иллюстрирует самый экономный план перевозок одинаковых или схожих продуктов от производственного объекта в пункт потребления. Планирование перевозок с помощью математических и вычислительных методов дает хороший экономический эффект.

Условия и ограничения транспортной задачи достаточно обширны и разнообразны. Поэтому для ее решения разработаны специальные методы. С помощью любого из них можно найти опорное решение. А впоследствии улучшить его и получить оптимальный вариант. Условия транспортной задачи можно представить двумя способами: в виде схемы и в виде матрицы.

В процессе решения могут быть ограничения (либо задача решается без них). По характеру условий различают следующие типы транспортных задач: открытые транспортные задачи (запас товара у поставщика не совпадает с потребностью в товаре у потребителя); закрытые (суммарные запасы продукции у поставщиков и потребителей совпадают). Закрытая транспортная задача может решаться методом потенциалов. Она всегда разрешима. Открытый тип сводят к закрытому с помощью прибавления к суммарному запасу или потребности в товаре недостающих единиц, чтобы добиться равенства.

**1.2. Идея и описание алгоритма**

Задача линейного программирования (ЗЛП) состоит в определении значений упорядоченной совокупности переменных xj, j = 1(1)n при которых линейная целевая функция достигает экстремального значения и при этом выполняются (удовлетворяются) все ограничения в форме равенств или неравенств. Требуется найти план Х <n> = <x1, x2, ..., xn>, который обеспечивает получение целевой функцией экстремального значения. Но имеются и свои ограничения: а) если система ограничений ЗЛП обладает хотя бы одним решением, она называется совместной в противном случае несовместной; б) допустимое множество решений ЗЛП не пусто, если система ограничений совместна; в) множество допустимых решений ЗЛП (если оно не пусто) в общем случае является многогранным множеством. Линейная функция Q(X<n>) называется функцией цели, целевой функцией (ЦФ), множество планов {X<n>} удовлетворяющих системе ограничений - множеством допустимых решений (альтернатив) и обозначается символом R, X<n>є Ω, допустимый план X<n>є Ω, доставляющий целевой функции экстремальное значение, называется оптимальным.

Удобство нашей формы ЗЛП состоит в том, что она позволяет предельно просто получить первое допустимое решение. Для этой формы должны быть выполнены условия:

* правые части в ограничениях – неотрицательны bi ≥ 0, i = 1(1)m;
* каждое уравнение содержит переменную xj ≥0 с коэффициентом при ней равным “1” в этом уравнении и с коэффициентом “0” во всех других уравнениях; эти переменные называют дополнительными или искусственными;
* в ЦФ эти переменные входят с коэффициентом “0”;
* определение опорного плана (начальной вершины) выполняют методом искусственного базиса, что ещё до решения задачи позволяет выяснить факт существования решения.

При таком типе возможны два варианта развития событий: суммарный объем производства превышает суммарную потребность в товаре; суммарная потребность больше суммы запасов.

Открытую транспортную задачу приводят к закрытому типу. В первом случае вводят фиктивного потребителя. Его потребности равны разнице всего объема производства и суммы существующих потребностей. Во втором случае вводят фиктивного поставщика. Объем его производства равен разнице суммарной потребности и суммарных запасов. Единица перевозки груза для фиктивного участника равняется 0. Когда все преобразования выполнены, транспортная задача становится закрытой и решается обычным способом.

**1.3. Математическое решение алгоритма.**

**1.3.1. Реализация алгоритма на Python**

Имеются m пунктов отправления груза и объёмы отправления по каждому по каждому пункту a1, a2. Известна потребность в грузах b1, b2, b3 по каждому из n пунктов назначения. Задана матрица стоимостей доставки по каждому варианту Cij, i = вектор (l,m), j = вектор (l,n). Необходимо рассчитать оптимальный план перевозок, т.е. определить, сколько груза должно быть отправлено из каждого i-го пункта отправления (от поставщика) в каждый j-й пункт назначения (до потребителя) с минимальными транспортными издержками. Исходя из составленной матрицы были составлены следующие ограничения:

(x[0] + x[1] + x[2] == 60)

(x[3] + x[4] + x[5] == 80)

(x[0] + x[3] == 35)

(x[1] + x[4] == 50)

(x[2] + x[5] == 55),

а также все представленные значения должны быть не отрицательными.

Так как произведение 𝐶𝑖𝑗∗𝑋𝑖𝑗 определяет затраты на перевозку груза от i-го поставщика j-му потребителю, то суммарные затраты на перевозку всех грузов равны:

c[0]\*x[0] + c[1]\*x[1] +c[2]\* x[2] +c[3]\*x[3] + c[4]\*x[4] +c[5]\* x[5] => max

Условие задачи требует минимальных затрат при перевозках. Следовательно, функция задачи имеет вид:

c[0]\*x[0] + c[1]\*x[1] +c[2]\* x[2] +c[3]\*x[3] + c[4]\*x[4] +c[5]\* x[5] => min

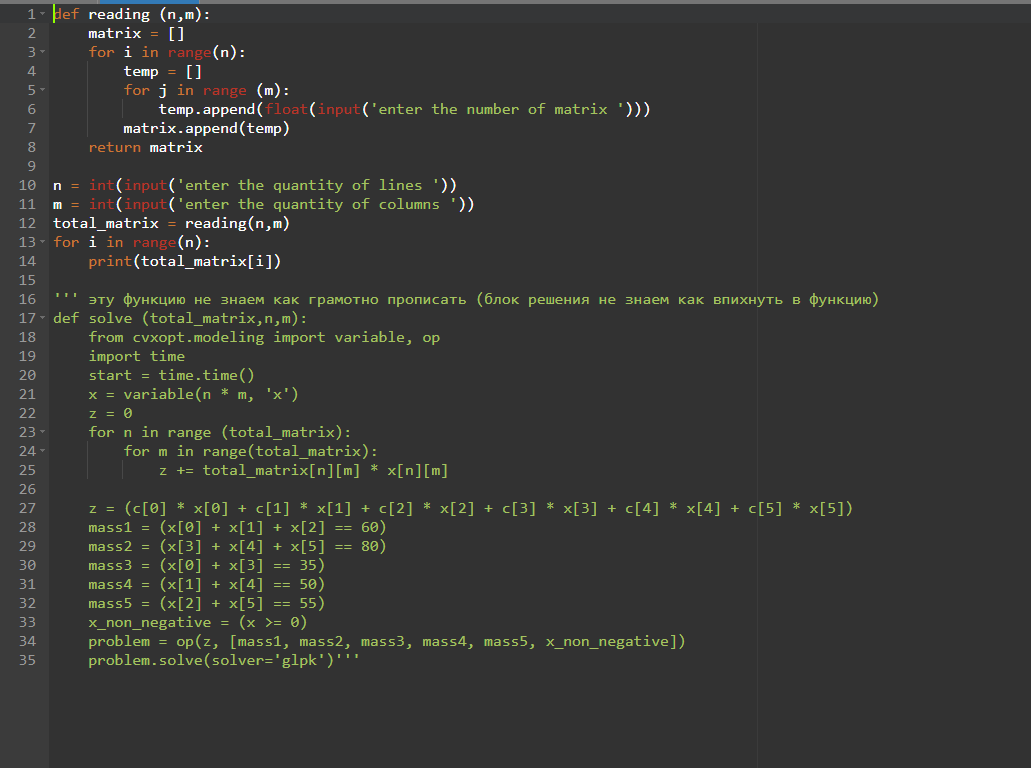
Учитывая условие не отрицательности объемов перевозок, математическая модель выглядит так (полное решение представлено на Рисунке 1):

1) c[0]\*x[0] + c[1]\*x[1] +c[2]\* x[2] +c[3]\*x[3] + c[4]\*x[4] +c[5]\* x[5] => min

2) 𝑎𝑖,𝑖 = 𝑛𝑗 = 1,2,3….,𝑚.

3) 𝑏𝑖,𝑗 = 𝑚𝑖 = 1,2,3….,𝑛.

4) 𝑥𝑖𝑗≥0, 𝑖=1,2,3…,𝑚;𝑗=1,2,…,𝑛

*****Рисунок 1. Решение задачи в Python*

**1.3.2. Реализация алгоритма в Excel**

Для решения данной задачи в табличном процессоре необходимо составить две таблицы. Для решения транспортной задачи потребуются функции: СУММПРОИЗВ, СУММ и надстройка «Поиск решения». Для отображения формул необходимо на вкладке «Формулы» в группе «Зависимости формул» выбрать «Показать формулы» либо горячее сочетание клавиш «Ctrl+` (тильда)». Дальше выбираем команду «Поиск решения» на вкладке «Данные». Решение поставленной задачи представлено ниже.

Проверим, является ли модель транспортной задачи сбалансированной. Для этого все количество производимого товара сравним с суммарным объемом потребности в продукции. Сформулируем ограничения: объем перевозимой продукции не может быть отрицательным и весь товар должен быть доставлен к пунктам назначения. Затем введем стоимость перевозки единицы продукции в рабочие ячейки Excel. Далее введем формулы для расчета суммарной потребности в товаре. Это будет первое ограничение. И введем формулы для расчета суммарного объема производства. Это будет второе ограничение. Вносим известные значения потребности в товаре и объема производства. Вводим формулу целевой функции СУММПРОИЗВ, где первый массив – стоимость единицы перевозки товаров, а второй – искомые значения транспортных расходов.

После данных шагов, вызываем команду «Поиск решения» на закладке «Данные» (если там нет данного инструмента, то его нужно подключить в настройках Excel, а как это сделать описано в статье: расширенные возможности финансового анализа). Заполняем диалоговое окно. В графе «Установить целевую ячейку» - ссылка на целевую функцию. Ставим галочку «Равной минимальному значению». В поле «Изменяя ячейки» - массив искомых критериев. В поле «Ограничения»: искомый массив >=0, целые числа; «ограничение 1» = объему потребностей; «ограничение 2» = объему производства. Нажимаем «Выполнить». Команда подберет оптимальные переменные при заданных ограничениях. Полученный результат представлен на рисунке 2.



*Рисунок 2. Решение задачи в Excel*

**1.4. Вывод**

Решив данную задачу через Python и Excel, мы можем прийти к общему выводу и представить оптимальный план перевозок:

|  |  |  |  |
| --- | --- | --- | --- |
| **Производитель/Потребитель** | **b1** | **b2** | **b3** |
| **a1** | 35 | 25 | 0 |
| **a2** | 0 | 25 | 55 |

*Матрица 2. Итоговый результат*

Отвечая на главный вопрос данной задачи, наша стоимость доставки будет составлять 270 д. е.

**1.5. Перспективы и варианты использования**

Наш код можно достаточно легко использовать для решения различных производственных задач, которые в наше время очень нужны большому количеству производств в разных сферах деятельности. Правда с небольшими изменениями под конкретного заказчика. Благодаря нашему коду заказчик сможет находить максимально выгодные пути производства в задачах, где это требуется. В дальнейшем мы планируем совершенствовать и улучшать нашу разработку для более точных расчетов и удобства пользования.

**2. Условие второй задачи (физическая модель)**

Каждой дуге (x,y) исходного графа G ставится в соответствие число a(x,y). Если в графе отсутствует некоторая дуга, то полагается, что a(x,y)=∞. Число a(x,y) называется длиной дуги, или весовым коэффициентом. Длина пути – сумма длин отдельных дуг, составляющих этот путь. Для любых двух вершин s и t графа G могут существовать несколько путей, соединяющих вершину s с вершиной t. Путь, имеющий минимально возможную длину, называется кратчайшим путем.

**2.1. Алгоритм Дейкстра. Идея и описание алгоритма**

Идея. Предполагается, что известны m вершин, ближайших к вершине s (близость любой вершины x к вершине s определяется длиной кратчайшего пути, ведущего из s в х), и кратчайшие пути, соединяющие вершину s с выделенными m вершинами.

Далее определяется (m+1)-я ближайшая вершина к s. Окрашивается вершина s и m ближайших к ней вершин.

Для каждой неокрашенной вершины y строятся пути, непосредственно соединяющие с помощью дуг (x,y) каждую окрашенную вершину х с у. Из этих путей выбирается кратчайший, который считается условно кратчайшим путем из вершины s в вершину у. Вершина, для которой условно кратчайший путь имеет наименьшую длину, и будет являться (m+1)-й ближайшей вершиной к вершине s. Начиная с m = 0, описанная процедура повторяется до тех пор, пока не будет получен кратчайший путь из s в t.

**2.1.2. Математическое решение алгоритма Дейкстра**

Шаг 1. Перед началом выполнения алгоритма все вершины и дуги не окрашены. Каждой вершине присваивается число d(x), равное длине кратчайшего пути из s в x, включающего только окрашенные вершины.

Пусть d(s) = 0 , d(x) =∞ для всех вершин x, отличных от s. Окрасить вершину s и положить y = s.

Шаг 2. Для каждой неокрашенной вершины х пересчитать величину d(x) по формуле:

d(x) = min{d(x),d(y) + a(y,x) )}

Если d(x) =∞ для всех неокрашенных вершин х, закончить процедуру алгоритма: в исходном графе отсутствуют пути из s в неокрашенные вершины. В противном случае окрасить ту из вершин х, для которой величина d(x) является наименьшей, а также окрасить дугу, ведущую в выбранную на данном шаге вершину x.

Положить у = х.

Шаг 3. Если у = t, закончить процедуру алгоритма: кратчайший путь из вершины s в вершину t найден (это единственный путь из s в t, составленный из окрашенных дуг). В противном случае перейти к шагу 2.

Окрашенные дуги образуют в исходном графе ориентированное дерево с корнем в вершине s. Это дерево называется ориентированным деревом кратчайших путей. Замечание. Алгоритм Дейкстры применяется в предположении неотрицательности длин дуг исходного графа.

**2.1.3. Реализация алгоритма на Python**

Для начала зададим необходимые данные для создания матрицы

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 3.1. Данные для создания матрицы*

Начало алгоритма Дейкстры

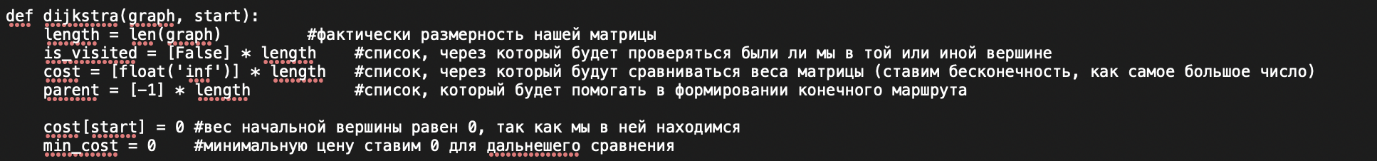
Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 3.2. Начало алгоритма*

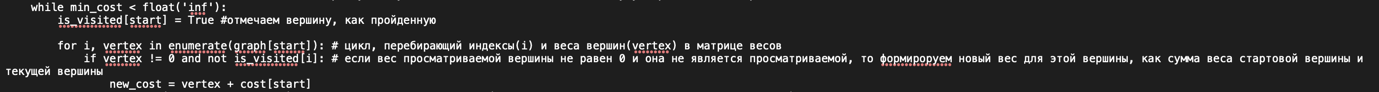
Заводим функцию поиска наикратчайшего расстояния. Так как матрицу можно представить в виде списка списков, будем рассматривать нашу матрицу построчно.

Наша функция Дейкстра имеет два аргумента:(graph, start), обозначающие саму матрицу (где i-тый элемент этой матрицы есть строка матрицы) и вершину, с которой мы хотим начать.



*Рисунок 3.3. Функция поиска наикратчайшего расстояния*

Заводим цикл, в котором находится основной алгоритм Дейкстры, он выполняется, пока все минимальные веса по алгоритму станут меньше бесконечности, то есть будут пройдены все вершины



*Рисунок 3.4. Создание цикла*

Если вес новой вершины (найдена по формуле) меньше, предыдущего значения (изначально, мы составили матрицу бесконечностей), то мы выбираем то значение, которое меньше.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 3.5. Выбор значения*

Дальше нам необходимо выбрать следующую вершину, с которой нам начать движение. Это должна быть вершина с наименьшим весовым значением из всей строки. Цикл, реализующий этот алгоритм ниже. В конце мы присвоим переменной start тот индекс, вес вершины которой оказался наименьшим. Это и будет наша следующая вершина.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 3.6. Выбор следующей вершины*

Реализация цикла для формирования маршрута путей

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 3.7. Реализация цикла*

**2.2. Алгоритм Флойда.**

**2.2.1. Идея и описание алгоритма**

Постановка задачи. Поиск на графе кратчайшего пути между каждой парой вершин. Вершины исходного графа перенумеровываются целыми числами от 1 до N. Через dijm обозначается длина кратчайшего пути из вершины i в вершину j, который в качестве промежуточных может содержать только первые m вершин графа.

Промежуточной вершиной пути является любая принадлежащая ему вершина, не совпадающая с его начальной или конечной вершинами.

Если между вершинами i и j не существует ни одного пути указанного типа, то условно считается, что dijm =∞. Величина dij0 – длина кратчайшего пути из i в j, не имеющего промежуточных вершин, т. е. длина кратчайшей дуги (i, j). Полагается, что dij = 0. Пусть Dm – матрица размера N × N, элемент (i, j)-й которой совпадает с dijm.

**2.2.2. Математическое решение алгоритма Флойда**

Шаг 1. Перенумеровать вершины исходного графа целыми числами от 1 до N. Определить матрицу D0, задав величину каждого ее элемента (i, j) равной длине кратчайшей дуги, соединяющей вершину i c вершиной j. Если в исходном графе указанные вершины не соединяются дугами, положить dij0=∞. Кроме того, для всех вершин i положить dij0 = 0 .

Шаг 2. Для целого m, последовательно принимающего значения 1, 2, 3, .., N, определить по величинам элементов матрицы Dm-1 величины элементов матрицы Dm , используя соотношение (1.2). При определении величины каждого элемента матрицы Dm фиксировать соответствующий кратчайший путь. По окончании данной процедуры величина элемента (i, j) матрицы DN определяет длину кратчайшего пути, ведущего из вершины i в вершину j.

**2.2.3. Реализация алгоритма на Python**

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 4.1. Первая часть реализованного кода*

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 4.2. Вторая часть реализованного кода*

**2.3. Идея и описание алгоритма поиска решения в Excel**

Задача выбора кратчайшего пути задана сетью, изображенной на рис. 3.1. Найти кратчайший путь от узла с номером 1 до узла с номером 5, если *c12*=7 , *c13*=9, *c16*=14, *c24*=15, *c23*=10, *c34*=11, *c36*=2, *c56*=9, *c45*=6.

Поскольку граф неориентированный, расстояние между соединенными точками одинаково.

**Изображение выглядит как текст, часы

Автоматически созданное описание**

*Рисунок 5. Неориентированный граф*

**2.3.1. Описание алгоритма поиска решения**

**Изображение выглядит как стол

Автоматически созданное описание**

*Рисунок 6. Таблица кратчайших расстояний*

Здесь, в Таблице кратчайших расстояний мы видим, что если между отдельными точками отсутствует возможность перевозки товара, то в соответствующие ячейки таблицы (выделенные темным фоном) заносятся нули.

Данная задача решается аналогично решению транспортной задачи, в целевую ячейку мы заносим формулу: =СУММПРОИЗВ.

Таблица – симметричная матрица, поскольку мы имеем взвешенный, неориентированный граф. Мы создаем дополнительную таблицу, в которой и будет осуществлено решение путем использования инструмента «Поиск решения». Используя меню Сервис ⇒ Поиск решения открываем диалоговое окно Поиск решения (Рисунок 7.1), в котором устанавливаем целевую ячейку равной минимальному значению, определяем диапазон изменяемых ячеек и ограничения и запускаем процедуру вычисления, щелкнув по кнопке Выполнить.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 7.1. Параметры поиска решения*

Результат решения данной задачи представлен на Рисунке 7.2

**Изображение выглядит как стол

Автоматически созданное описание**

*Рисунок 7.2. Результат решения задачи*

После того, как Excel найдет решение, можно определить кратчайший путь следующим образом. В нижней строке нетрудно видеть, что из первого пункта автомобиль следует в пункт 3, затем в пункт 6, откуда приходит в конечную точку 5. Наикратчайший маршрут равен 20.

Таким образом, мы видим, что кратчайший путь перевозки товара, следующий: 2→3→6→5. Расстояние перевозки при этом составит 20. Аналогично данную задачу можно решить и на максимум, т.е. найти самый длинный путь доставки товара. При расчетах результат Excel совпал с алгоритмом Дейкстра.

**2.4. Варианты использования системы**

Данная система очень нужна и важна по следующим причинам:

Первая причина заключается в быстроте достижения заданной цели, которая напрямую зависит от правильности выбора маршрута. При этом самый короткий путь в данной задаче является самым быстрым. Следует принимать во внимание количество проходимых узлов и отрезков. Поэтому, верный расчет строится на анализе различных путей и полученных результатах;

Вторая причина в том, что значение имеет и уровень расхода. Чтобы правильно рассчитать маршрут и расход, необходимо учитывать множество факторов. Поэтому, проложить маршрут по карте и рассчитать расстояние следует с учетом расхода. Снижение данных расходов представляет собой фактическую прибыль.

**2.4.1. Первый вариант использования**

Графы можно использовать для моделирования транспортной сети, где точки (вершины) являются объектами, которые отправляют или получают посылки/продукты (места, что водитель должен посетить), а рёбра (линии) представляют собой соединяющие их дороги.

Графы могут быть двух типов: направленный граф: если для каждой пары узлов можно сделать переход от одного узла к другому только в определённом направлении; для обозначения такого графа используются стрелки вместо простых линий (пример из жизни: когда на дороге есть только одностороннее движение); ненаправленный граф: если для каждой пары узлов можно сделать переход от одного к другому в обоих направлениях (пример из жизни: когда на дороге есть двустороннее движение). Как работает алгоритм?

Алгоритм произведёт кратчайший путь от узла 0 ко всем остальным узлам графа. Для этого графа мы будем предполагать, что вес рёбер (вес или стоимость всегда являются неотрицательными, т.е. ≥ 0; это цифра возле каждой линии) представляет собой расстояние между двумя узлами.

**2.4.2. Второй вариант использования**

Как и любой базовый алгоритм, данные алгоритмы используются очень широко и много где. Одна из таких областей — это генетика. С помощью данных алгоритмов можно вывести признаки потомства у определенного семейства, где важную роль на выбор пути будут играть доминантные и рецессивные гены. И чем больше узлов, грубо говоря: чем дальше растёт потомство, тем больше способов достижения определенного фенотипа у организма.

**2.5. Тестирование**

|  |  |  |  |
| --- | --- | --- | --- |
| Входные данные | Алгоритм Дейкстра | Алгоритм Флойда | MS Excel |
| Датасет 1 (путь из 1 в 5) | 20 | 20 | 20 |
| Датасет 2 (путь из 2 в 6) | 12 | 12 | 12 |
| Датасет 3 (путь из 2 в 1) | 7 | 7 | 7 |

*Таблица 1. Результаты тестирования программы*

После тестирования всех наших способов решения стало ясно, что все опробованные способы показывают точный результат без погрешностей. В ходе тестирования мы приняли решение, что самым оптимальным вариантом использования является MS Excel так как по быстроте вычислений и простоте работы с ним для рядового пользователя он подходит лучше всего.

**2.6. Заключение**

Созданное ПО позволяет решать задачи эффективной организации перевозок для любого конечного количества пунктов, что позволяет экономно расходовать денежные средства на перевозки и эффективно использовать рабочее время.

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | Алгоритм Дейкстра | Алгоритм Флойда | Алгоритм MS Excel |
| 1. Простота использования | Очень просто | Сложно | Относительно просто |
| 2. Наглядность решения | Достаточно наглядно | Достаточно наглядно | Наглядно |
| 3. Частота использования | Очень часто | Редко | Относительно часто |
| 4. Время выполнения | Меньший срок | Больший срок | Средний срок |
| 5. Обработка графов, в которых отрицательные рёбра | Невозможно | Возможно | Возможно |

*Таблица 2. Сравнение алгоритмов.*

Оптимальный способ для заказчика является алгоритм MS Excel, так как он более понятен обычному пользователю, не затрачивает большое количество времени на достижение результата и позволяет обрабатывать графы, в которых имеются отрицательные рёбра.

**3. Условие третьей задачи (Физическая модель)**

В задаче имеются две, конкурирующие в условиях рынка компании. Конкретно к нам обратился заказчик (компания А) с потребностью выйти на потребительский рынок товаров для детей, с такими товарами как: игрушки, детское питание, одежда для детей. Конкурентами нашей компании является компания (В), которая в свою очередь производит игрушки, детскую литературу, и подгузники. Нашему заказчику необходимо просчитать риски в представленных категориях. Целью компании А является желание занять лидирующее место в своей нише.

Для первой задачи мы составили матрицу следующего характера: поскольку у нас есть три сценария развития того, какие три товара мы можем продавать и какие три товара, может продавать наш конкурент, сравним стратегию в каждом отдельном случае.

Вероятность продажи первого наименования товара, а именно детских игрушек у компании А равен столько же, сколько и у компании Б.

Относительно вероятности продажи детских игрушек к вероятности продажи детской литературы, компания А продаст 75%, так как по данным исследования проекта Дети Mail.ru, имеющегося в распоряжении РИА Новости, отмечается, что самыми популярными категориями детских товаров, которые родители купили за последние полгода, стали игрушки (66%). Следом за ними идут книги (42%). По той же статистике, сравнивая продажи детских игрушек и подгузников, мы можем вычитать, что за последние полгода покупка игрушек (66%), немного превысила покупку средств гигиены, а именно подгузников (59%).

Продажа детского питания от компании А по отношению к продаже товаров компании Б соотносится как 70, 80 и 40. Также по статистике отметили, что чаще всего в магазинах покупают детское питание, чем вышеуказанные товары и поддержали эту точку зрения — 87% родителей.

Что касается детской одежды, статистика гласит о том, что самыми популярными категориями детских товаров, которые родители купили за последние полгода, стали одежда (93%), однако, ориентируясь на спросы на товар, подгузники значительно обходят покупку детской одежды.

**3.1. Математическая модель.**

**3.2. Игры с «природой»**

Неопределенность является характеристикой внешней среды (природы), в которой принимается управленческое решение о развитии (или функционировании) экономического объекта. Здесь будем рассматривать неопределенность «природы», вызванную отсутствием, недостатком информации о действительных условиях (факторах), при которых развивается объект управления. Внешняя среда («природа») может находиться в одном из множества возможных состояний. Это множество может быть конечным и бесконечным. Будем считать, что множество состояний конечно или по крайней мере количество состояний можно пронумеровать.

Пусть Si — состояние «природы», при этом i= 1,n, где n — число возможных состояний. Все возможные состояния известны, не известно только, какое состояние будет иметь место в условиях, когда планируется реализация принимаемого управленческого решения. Будем считать, что множество управленческих решений (планов) Rj также конечно и равно m.

Реализация Rj плана в условиях, когда «природа» находится в Si состоянии, приводит к определенному результату, который можно оценить, введя количественную меру. В качестве этой меры могут служить выигрыши от принимаемого решения (плана); потери от принимаемого решения, а также полезность, риск и другие количественные критерии. Данные, необходимые для принятия решения в условиях неопределенности, обычно задаются в форме матрицы, строки которой соответствуют возможным действиям (управленческим решения) Rj, а столбцы — возможным состояниям «природы» Si.

Допустим, каждому Rj-му действию и каждому возможному Si-му состоянию «природы» соответствует результат (исход), определяющий результат (выигрыш, полезность) при выборе j-го действия и реализации i-го состояния, — Vji.

Изображение выглядит как стол

Автоматически созданное описание

*Матрица 3. Матрица состояния «природы»*

Следовательно, математическая модель задачи принятия решений определяется множеством состояний {Si}, множеством планов (стратегий) {Rj} и матрицей возможных результатов \\Vji\\. В качестве результатов в отдельных задачах рассматривается матрица рисков \\rji\\. Риск — мера несоответствия между разными возможными результатами принятия определенных стратегий (действий). Элементы матрицы рисков \\rji\\ связаны с элементами матрицы полезностей (выигрышей) следующим соотношением:

Изображение выглядит как часы

Автоматически созданное описание

*Формула 1. Формула риска*

где Vi = max Vji — максимальный элемент в столбце i матрицы полезностей. Если матрица возможных результатов \\Vji\\ представляет собой матрицу потерь (затрат), то элементы матрицы рисков \\rj\\ следует определять по формуле:

Изображение выглядит как текст, часы

Автоматически созданное описание

*Формула 2. Формула риска*

где Vi = min Vji — минимальный элемент в столбце i матрицы потерь (результатов).

Таким образом, риск — это разность между результатом, который можно получить, если знать действительное состояние «природы», и результатом, который будет получен при j-й стратегии. Матрица рисков дает более наглядную картину неопределенной ситуации, чем матрица выигрышей (полезностей). Непосредственный анализ матриц выигрышей \\Vjr\\или рисков \\rji\\ не позволяет в общем случае принять решение по выбору оптимальной стратегии (плана), за исключением тривиального случая, когда выигрыши при одной стратегии выше, чем при любой другой для каждого состояния «природы» (элементы матрицы выигрышей в некоторой строке больше, чем в любой из других). Другими словами, имеется в наличии «доминирующая» стратегия. Для принятия решения в условиях неопределенности используется ряд критериев. Рассмотрим некоторые из них. Это критерий Лапласа, критерий Вальда, критерий Сэвиджа, критерий Гурвица.

**3.3. Формулировка задачи**

Составим условие для реализации нашей игры. Мы владельцы магазина игрушек. В начале каждого дня мы закупаем несколько игрушек по 50 руб. Цена реализации нашего продукта - 60 рублей за единицу. Из наблюдений известно, что спрос на этот продукт за день может быть равен 1,2,3 или 4 единицам. Пусть известно, что на практике спрос на 1 единицу наблюдался X раз, спрос на 2 единицы наблюдался Y раз, спрос на 3 единицы наблюдался Z раз, спрос на 4 единицы наблюдался Q раз. Если продукт в течение дня не был распродан, то в конце дня мы его продавали друзьям из ближнего зарубежья по цене 20 рублей за единицу.

**3.4. Ввод данных пользователем**

Для реализации данной задачи на Python пользователь должен ввести количество стратегий для нашей игры, а также указать какой спрос для различных сценариев покупки товара.

После приведённых выше действий, будут сформированы вероятности наступления этих событий.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 8.1. Результаты ввода данных пользователем*

Для вычисления необходимых критериев наша программа должна попросить от пользователя значения финансовых последствий, которые мы занесём в матрицу.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 8.2 Функция ввода данных пользователем*

Сформированная матрица выглядит следующим образом:

Изображение выглядит как текст

Автоматически созданное описание

*Матрица 4. Финансовые последствия*

**3.5. Критерий Байеса**

Критерий Байеса — это числовая характеристика стратегий в играх с природой. Значение критерия Байеса — это наибольшее значение математического ожидания выигрыша.

Изображение выглядит как текст

Автоматически созданное описание

*Формула 3. Нахождение критерия Байеса*

**3.5.1. Реализация на питоне**

#байес

a1=np.array(v1)

b1=np.array(matrix\_a)

bayes=np.sum(a1\*b1,axis=1)

print('Цена игры по критерию Байеса составляет: ',int(max(bayes)), "при выборе стратегии номер: ",

list(bayes).index(max(bayes))+1)

**3.6. Критерий Лапласа**

Этот критерий опирается на «принцип недостаточного основания» Лапласа, согласно которому все состояния «природы» Si, i = 1, n полагаются равновероятными. В соответствии с этим принципом каждому состоянию Si ставится вероятность qi, определяемая по формуле:

Изображение выглядит как текст

Автоматически созданное описание

*Формула 4. Формула вероятности*

При этом исходной может рассматриваться задача принятия решения в условиях риска, когда выбирается действие Rj, дающее наибольший ожидаемый выигрыш.

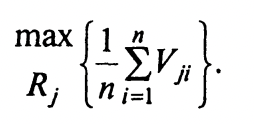
Для принятия решения для каждого действия Rj вычисляют среднее арифметическое значение выигрыша:

Изображение выглядит как текст

Автоматически созданное описание

*Формула 5. Среднее арифметическое значение выигрыша*

Среди Мj(R) выбирают максимальное значение, которое будет соответствовать оптимальной стратегии Rj. Другими словами, находится действие Rj, соответствующее:



Формула 6. Максимальное значение

Если в исходной задаче матрица возможных результатов представлена матрицей рисков \\rji\\, то критерий Лапласа принимает следующий вид:

**Изображение выглядит как текст

Автоматически созданное описание**

*Формула 7. Минимальное значение*

**3.6.1. Реализация на питоне**

#лаплас

v2=[0.25,0.25,0.25,0.25]

a2=np.array(v2)

laplas=np.sum(a2\*b1,axis=1)

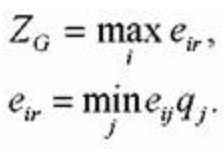
print('Цена игры по критерию Лапласа составляет: ',int(max(laplas)), "при выборе стратегии номер: ", list(laplas).index(max(laplas))+1)

**3.7. Критерий Гермейера**

Опираясь на подход к отысканию эффективных и пригодных к компромиссу решений, которые не считаются заведомо худшими, чем другие, - можно предложить критерий Гермейера (G), обладающий в некотором отношении определенной эластичностью.

Он с самого начала ориентирован на величины потерь, то есть на отрицательные значения всех еij.

В качестве оценочной функции выступает



*Формула 8. Критерии выступающие в качестве оценочной функции*

Изображение выглядит как текст

Автоматически созданное описание

*Формула 9. Оценочная функция*

Оптимальных по данному критерию решений записывается в виде eij < 0, обычно выполняется, в случае, когда среди eij – a1, при подходящем образом подобранном а > 0, следует, однако, иметь в виду, что оптимальный вариант решения зависит от а. Правило выбора, согласно G-критерию, формулируется следующим образом: {eij}, которое дополняется еще одним столбцом, содержащим в каждой строке наименьшее произведение имеющегося в ней результата на вероятность соответствующего состояния Fj. Затем выбираются те варианты Ei0, в строках которых находится наибольшее значение eir этого столбца. Тогда qj = 1/n, j=1, …, n ­- они становятся идентичными.

Условия применимости G-критерия таковы: вероятности появления состояний Fj известны; с появлением тех или иных состояний, отдельно или в комплексе, необходимо считаться; допускается некоторый риск; решение может реализоваться один или много раз. Если функция распределения известна не очень надежно, а числа реализаций малы, то, следуя G-критерию, получают неоправданно большой риск. Таким образом, здесь остается некоторая свобода для субъективных действий.

**3.7.1. Реализация на питоне**

#гермейер

G=[]

for i in range(n):

g=[]

for j in range(n):

q=matrix\_a[i][j]\*v1[j]

g.append(q)

G.append(g)

Gg=[]

for i in G:

Gg.append(min(i))

print('Цена игры по критерию Гермейра составляет: ',int(max(Gg)),"при выборе стратегии номер: ",

Gg.index(max(Gg))+1)

**3.8. Критерий оптимизма**

При использовании данного критерия, называемого также кри­терием максимакса, ЛПР ориентируется на то, что условия функционирования анализируемых систем будут для него наиболее бла­гоприятными. Вследствие этого оптимальным решением является стратегия, приво­дящая к получению наибольшего значения критерия оптимальности в платежной матрице. Этот критерий целесообразно применять в тех случаях, когда имеется принципиальная возможность повлиять на функции противоположной стороны.

Если анализируется матрица эффекта Е(Р, П) того или иного вида, то выбор управляемых факторов осуществляется таким образом, чтобы обеспечить максимум эффекта. И в этом случае критерий оптимизма записывается в виде

Изображение выглядит как текст

Автоматически созданное описание

*Формула 10. Критерий оптимизма*

**3.8.1.Реализация на питоне**

#оптимизм

O=[]

for i in matrix\_a:

O.append(max(i))

print('Цена игры по критерию оптимизма составляет: ',int(max(O)),"при выборе стратегии номер: ",

O.index(max(O))+1)

**3.9. Критерий пессимизма**

В отличие от критерия оптимизма, когда ЛПР ориентируется на наи­более благоприятную внешнюю среду, которая является неконтроли­руемой, и на оптимальное использование управляемых факторов, при использовании принципа пессимизма предполага­ется, что управляемые факторы могут быть использованы небла­гоприятным образом:

Изображение выглядит как текст

Автоматически созданное описание

*Формула 11. Критерий пессимизма*

где Е(Р, П) — функция эффективности принимаемых решений.

Применение этого принципа может вызывать некоторое со­мнение, если учесть, что факторы Р являются контролируемы­ми и их следует использовать оптимальным в том или ином смыс­ле образом.

Однако, в реальных ситуациях в ряде задач может оказаться невозможным кон­троль за неконтролируемыми фак­торами, принадлежащими множеству Р. Особенно это отно­сится к задачам, связанным с необходимостью учета фактора времени.

**3.9.1. Реализация на питоне**

#пессимизм

P=[]

for i in matrix\_a:

P.append(min(i))

print('Цена игры по критерию пессимизма составляет: ',int(min(P)),"при выборе стратегии номер: ",

P.index(min(P))+1)

**3.10. Критерий Вальда**

Критерий Вальда (минимаксный или максиминный критерий). Применение данного критерия не требует знания вероятностей состояний Si.

Этот критерий опирается на принцип наибольшей осторожности, поскольку он основывается на выборе наилучшей из наихудших стратегий Rj,

Если в исходной матрице (по условию задачи) результат Vji представляет потери лица, принимающего решение, то при выборе оптимальной стратегии используется минимаксный критерий.

Для определения оптимальной стратегии Rj необходимо в каждой строке матрицы результатов найти наибольший элемент max {Vji}, а затем выбирается действие Rj (строка j), которому будет соответствовать наименьший элемент из этих наибольших элементов, т. е. действие, определяющее результат, равный:

**Изображение выглядит как текст

Автоматически созданное описание**

*Формула 12*

Если в исходной матрице по условию задачи результат Vji представляет выигрыш (полезность) лица, принимающего решение, то при выборе оптимальной стратегии используется максиминный критерий.

Для определения оптимальной стратегии Rj в каждой строке матрицы результатов находят наименьший элемент min{Vji}, а затем выбирается действие Rj (строка j), которому будут соответствовать наибольшие элементы из этих наименьших элементов, т. е. действие, определяющее результат, равный

**Изображение выглядит как текст

Автоматически созданное описание**

*Формула 13*

**3.10.1. Реализация на питоне**

Можно использовать стратегию пессимизма (см.выше) и найти максимум, тогда:

print('Цена игры по критерию Вальда составляет: ',int(max(P)),"при выборе стратегии номер: ", P.index(max(P))+1)

**3.11. Критерий Сэвиджа**

Критерий Сэвиджа использует матрицу рисков \\rji\\. Элементы данной матрицы можно определить по формулам, которые перепишем в следующем виде:

**Изображение выглядит как текст

Автоматически созданное описание**

*Формула 14*

Это означает, что rji есть разность между наилучшим значением в столбце i и значениями Vji при том же i. Отметим, что независимо от того, является ли Vji доходом или потерями, rji в обоих случаях определяет величину потерь лица, принимающего решение. Критерий Сэвиджа рекомендует в условиях неопределенности выбирать ту стратегию Rj, при которой величина риска принимает наименьшее значение в самой неблагоприятной ситуации.

**3.11.1. Реализация на питоне**

#сэвидж

S1=[]

s=np.array(matrix\_a)

s=np.rot90(s,k=-1)

for i in s:

S1.append(max(i))

s=np.rot90(s)

for i in range(n):

for j in range(n):

s[i][j]=S1[j]-s[i][j]

S2=[]

for i in s:

S2.append(max(i))

print('Цена игры по критерию Сэвиджа составляет: ',int(min(S2)),

"при выборе стратегии номер: ", S2.index(min(S2))+1)

**3.12. Критерий Гурвица**

Основан на следующих двух предположениях: «природа» может находиться в самом невыгодном состоянии с вероятностью (1 — а) и в самом выгодном состоянии с вероятностью а, где а - коэффициент доверия. Если результат Vji — прибыль, полезность, доход и т п., то критерий Гурвица записывается так:

Изображение выглядит как текст

Автоматически созданное описание

*Формула 15. Критерий Гурвица при доходе*

Когда Vji представляет затраты (потери), то выбирают действие, дающее

Изображение выглядит как текст

Автоматически созданное описание

*Формула 16. Критерий Гурвица при затратах*

Если a = 0, получим пессимистический критерий Вальда.

Если а = 1, то приходим к решающему правилу вида maxmin Vji или к так называемой стратегии «здорового оптимиста», т. е. критерий слишком оптимистичный.

Критерий Гурвица устанавливает баланс между случаями крайнего пессимизма и крайнего оптимизма путем взвешивания обоих способов поведения соответствующими весами (1 — а) и а, где 0 ≤ а ≤ 1.

Значение а от 0 до 1 может определяться в зависимости от склонности лица, принимающего решение, к пессимизму или к оптимизму.

**3.12.1. Реализация на питоне**

#Гурвиц

print('Задайте ВАШ коэффициент готовности к риску от 0 до 10, где 0 - Вы шахматист,

а 10 - Вы пилот болида Формулы 1')

k=int(input())/10

O1=[]

for i in matrix\_a:

O1.append(max(i))

P1=[]

for i in matrix\_a:

P1.append(min(i))

Gur=[]

for i in range(n):

Gur.append(k\*O1[i]+(1-k)\*P1[i])

print('Цена игры по критерию Гурвица составляет: ',int(max(Gur)),

"при выборе стратегии номер: ", Gur.index(max(Gur))+1)

**3.13. Антагонистическая игра**

Антагонистическая игра — это игра двух участников с прямо противоположными интересами. Формально эта противоположность означает, что при переходе от одной игровой ситуации к другой увеличение выигрыша одного из игроков влечет численно равное уменьшение выигрыша другого, так что во всех ситуациях сумма выигрышей игроков постоянна (можно считать, что эта сумма равна нулю, т. е. что выигрыш одного игрока равен проигрышу другого). Поэтому А. и. наз. также играми двух лиц с нулевой суммой. Математическое понятие антагонистичности (равенство по величине и противоположность по знаку функции выигрыша) является формальным понятием, отличающимся от содержательного философского понятия. Если в А. и. в результате к.-л. переговоров и соглашений один из игроков смог бы увеличить свой выигрыш на некоторую сумму, то его противник потерял бы такую же сумму. Следовательно, любые соглашения оказываются невыгодными для одного из игроков и потому невозможными. Реальными конфликтными ситуациями, для которых А. и. служат достаточно адекватными моделями, являются нескорые (но не все) военные операции, спортивные и салонные игры, а также ситуации, связанные с принятием деловых решений в условиях конкуренции. Игры против природы и вообще принятие решений в условиях неопределенности (см. Статистическая игра) можно рассматривать как А. и. в предположении, что истинная закономерность природы, неизвестная игроку, приведет к действиям, наименее благоприятным для него.

В нормальной форме сводится к заданию множеств стратегий А и В соответственно игроков I и II и функции выигрыша Н игрока I, определенной на множестве всех ситуаций А × В (функция выигрыша игрока II равна, по определению А. и., - H). Формально А. и. Г записывается как тройка Г = 〈 A, В, H〉. Процесс разыгрывания игры Г состоит в выборе игроками нескорых своих стратегий а ∈ А, b ∈ В, после чего игрок I получает от игрока II сумму H(а, b). Это определение А. и. является достаточно общим, чтобы при должном описании множеств стратегий и функции выигрыша охватить все варианты А. и., включая динамические игры, дифференциальные игры и позиционные игры. Разумные действия игроков в А. и. осуществляются на основании принципа минимакса: если

Изображение выглядит как текст

Автоматически созданное описание

*Формула 17.1. Основания для принципа минимакса*

ИЛИ

Изображение выглядит как текст

Автоматически созданное описание

*Формула 17.2. Основания для принципа минимакса*

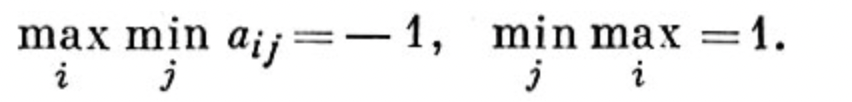
то в игре Г существуют оптимальные стратегии (ε - оптимальные стратегии) у обоих игроков. Общее значение обеих частей равенства (1') наз. значением игры Г. Однако равенство (1) или (1') уже в самых простых случаях может не иметь места. Напр., в матричной игре с матрицей выигрышей

Изображение выглядит как объект, часы

Автоматически созданное описание

*Матрица 5. Матрица выигрышей*

Имеют место равенства

****

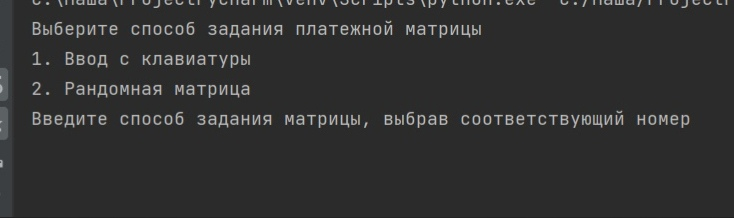
*Формула 18. Равенства принципа минимакс*

Поэтому множества стратегий игроков расширяются до множества смешанных стратегий, состоящих в случайном выборе игроками своих первоначальных стратегий, наз. чистыми, а функция выигрыша определяется как математических ожидание выигрыша в условиях применения смешанных стратегий.

В приведенном примере оптимальными смешанными стратегиями игроков являются выборы игроками обеих своих стратегий с вероятностями 1/2, а значение игры в смешанных стратегиях равно нулю. Если множества А и В конечны, то А. и. наз. матричной игрой; для нее всегда существуют значение игры и оптимальные смешанные стратегии у каждого из игроков. Если оба множества А и В бесконечны, то оптимальные (и даже ε-оптимальные) смешанные стратегии существуют не всегда (см. Бесконечная игра).

**3.13.1. Решение антагонистической игры на питоне**

В первую очередь, пользователю необходимо ввести платежную матрицу самому с клавиатуры, либо же выбрать случайное составление матрицы с помощью генератора на питоне.



*Рисунок 9.1. Ввод или генерация матрицы*

После пользователю предлагается ввести стратегии для каждого игрока, чтобы в дальнейшем программа могла вывести более понятное решение.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 9.2 Ввод стратегии*

Сам алгоритм основан на поиске максимина построчно и поиска минимакса по столбцам.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 9.3. Поиск максимина и минимакса*

Данный код позволяет найти чистые стратегии для игроков А и Б, а также вычисляет цены игры для этих стратегий.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 9.4. Цены игры для стратегий*

Для поиска смешанных стратегий мы использовали библиотеку nashpy, а конкретнее функцию nash.Game.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 9.5. Функция «nash.Game»*

Итоговый результат предоставлен на рисунке ниже.Первый массив отображает чистую стратегию игрока А, второй - чистую стратегию игрока Б, а третий смешанные стратегии.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 9.6. Итоговый результат*

**3.13.2. Решение Антагонистический задачи в Excel**

На Рисунке 10 представлена платежная матрица игрока А, а также прописаны стратегии игрока А и игрока Б

Изображение выглядит как стол

Автоматически созданное описание

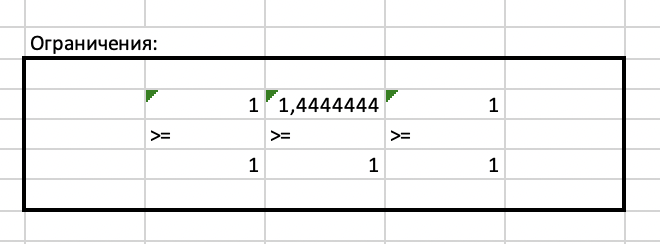
*Рисунок 10. Платежная матрица*

Платежная матрица была составлена на основе физической модели, представленной выше.

Первоначально мы искали минимальное значение по строкам и записывали их в столбец «MinMax» с помощью функции Excel: =МИН(ДИАПАЗОН), а потом находили максимальное значение среди них и выделяли его зеленым цветом.

Для формирования строки MaxMin производились немного другие вычисления: по каждому столбцу находилось максимальное значение с помощью функции Excel: =МАКС(ДИАПАЗОН), а потом в получившейся строке находилось минимальное значение и выделялось зеленым цветом.

Далее были прописаны ограничения:



*Рисунок 11. Ограничения*

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 12. Параметры поиска решения*

Используя меню Сервис ⇒ Поиск решения открываем диалоговое окно Поиск решения (Рисунок 12), в котором устанавливаем целевую ячейку равной минимальному значению, определяем диапазон изменяемых ячеек и ограничения и запускаем процедуру вычисления, щелкнув по кнопке Выполнить.

Находим цену игры разделив 1 на результат целевой функции.

Для того, чтобы определить смешанную стратегию умножаем полученные с помощью результата работы поиска решения переменные поочередно на цену игры.

Изображение выглядит как стол

Автоматически созданное описание

*Рисунок 13. Переменные и стратегия*

Получившиеся стратегии – наиболее удачные планы игры компании А.

**3.14. Биматричный алгоритм.**

**3.14.1. Решение Биматричной задачи в Excel**

На Рисунке 14 представлены платежные матрицы и стратегии игроков А и Б

**Изображение выглядит как стол

Автоматически созданное описание**

*Рисунок 14. Платёжные матрицы и стратегии игроков*

Для матрицы А находим максимальное значение в каждом столбце, выделяем его зеленым цветом и записываем строку ниже.

Для матрицы Б проводим аналогичные действия, только вычисляем максимальные значения по строкам.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 15. Параметры поиска решения*

Используя меню Сервис ⇒ Поиск решения открываем диалоговое окно Поиск решения (Рисунок 15), в котором устанавливаем целевую ячейку равной максимальному значению, определяем диапазон изменяемых ячеек и ограничения и запускаем процедуру вычисления, щелкнув по кнопке Выполнить.

В результате работы поиска решения находим целевую функцию, и с ее помощью вычисляем значение цен игры для игроков А и Б.

**3.15. Тестирование**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Входные данные | Байес | Лаплас | Гермейер | Оптимизм | Пессимизм | Вальд | Гурвиц | Севидж |
| Excel | 14 | 10 | 1 | 40 | -80 | 10 | 10 | 30 |
| Python | 14 | 10 | 1 | 40 | -80 | 10 | 10 | 30 |

*Таблица 3. Результаты тестирования программы для «игры с Природой»*

После тестирования всех наших способов решения стало ясно, что все опробованные способы показывают точный результат без погрешностей. В ходе тестирования мы приняли решение, что самым оптимальным вариантом использования является MS Excel так как по быстроте вычислений и простоте работы с ним для рядового пользователя он подходит лучше всего.

**3.16. Вариант использования**

Теория игр как один из подходов в прикладной математике применяется для изучения поведения человека и животных в различных ситуациях. Первоначально теория игр начала развиваться в рамках экономической науки, позволив понять и объяснить поведение экономических агентов в различных ситуациях. Позднее область применения теории игр была расширена на другие социальные науки; в настоящее время теория игр используется для объяснения поведения людей в политологии, социологии и психологии.

Теоретико-игровой анализ был впервые использован для описания поведения животных Рональдом Фишером в 30-х годах XX века (хотя даже Чарльз Дарвин использовал идеи теории игр без формального обоснования). В работе Рональда Фишера не появляется термин «теория игр». Тем не менее, работа, по существу, выполнена в русле теоретико-игрового анализа.

Разработки, сделанные в экономике, были применены Джоном Майнардом Смитом в книге «Эволюция и теория игр». Теория игр используется не только для предсказания и объяснения поведения; были предприняты попытки использовать теорию игр для разработки теорий этичного или эталонного поведения. Экономисты и философы применяли теорию игр для лучшего понимания хорошего (достойного) поведения.

**3.17. Заключение**

Зачастую человек, осуществляя какую-либо деятельность, сталкивается с проблемой принятия решения в условиях множества факторов, влияющих на само решение. В подобных случаях эффективно пользоваться матричными играми, которые помогают упростить сложившуюся ситуацию и полностью оценить важность каждого фактора.

Математическая теория игр применяется во многих областях человеческой деятельности, таких как таких как экономика и менеджмент, промышленность и сельское хозяйство, военное дело и строительство, социология, психология, и т.д.

**4. Список используемых источников**

# Симплексный метод решения задач линейного программирования [Электронный ресурс]. -Режим доступа: habr.com/ru/post/565068/

# Алгоритм Дейкстра [Электронный ресурс]. -Режим доступа: https://studfile.net/preview/5616813/

1. Алгоритм Флойда [Электронный ресурс]. -Режим доступа: neerc.ifmo.ru/wiki/index.php?title=Алгоритм\_Флойда
2. Принятие решений в условиях неопределенности [Электронный ресурс].Режим доступа: https://studref.com/663941/menedzhment/prinyatie\_resheniy\_usloviyah\_neopredelennosti
3. Антагонистическая игра [Электронный ресурс].Режим доступа: http://mathemlib.ru/mathenc/item/f00/s00/e0000218/index.shtml
4. Теория принятия решений. Биматричные игры [Электронный ресурс]. Режим доступа: https://moodle.kstu.ru/mod/book/view.php?id=11478
5. Критерий пессимизма [Электронный ресурс].Режим доступа: https://studbooks.net/29424/ekonomika/kriteriy\_pessimizma
6. Критерий оптимизма [Электронный ресурс].Режим доступа: https://studfile.net/preview/4289476/page:23/