МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет информационных технологий и робототехники (ФИТР)

Кафедра: «Программное обеспечение информационных систем и технологий»

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «Методы и алгоритмы принятия решений»

на тему: «Принятие решений в условиях неопределенности»

Выполнил: cт. гр. 10702221 Парфенова У. Д.

Приняла: ст. преподаватель И.М. Борисова

Минск 2023

Белорусский Национальный Технический Университет

Кафедра программного обеспечения информационных систем и технологий

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовому проекту**

**по дисциплине** «Методы и алгоритмы принятия решений»

Тема «Принятие решений в условиях неопределенности»

**Исполнитель:** Парфенова Ульяна Дмитриевна (фамилия, инициалы)

(подпись)

**Студент**  3 **курса** 10702221 **группы**

**Руководитель:** Борисова Ирина Михайловна (фамилия, инициалы)

(подпись)

Минск 2023

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 4](#_Toc152842839)

[1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 5](#_Toc152842840)

[1.1. Теория игр 5](#_Toc152842841)

[1.2. История возникновения теории игр 5](#_Toc152842842)

[1.3. Применение теории игр 6](#_Toc152842843)

[1.4 Методы принятия решений 6](#_Toc152842844)

[2. ПРАКТИЧЕСКИЙ РАЗДЕЛ 11](#_Toc152842845)

[2.1. Постановка задачи 11](#_Toc152842846)

[2.2. Математическая модель 11](#_Toc152842847)

[2.3. Выбор метода принятия решения 17](#_Toc152842848)

[2.3.1 Критерий Вальда 17](#_Toc152842849)

[2.3.2 Критерий Сэвиджа 19](#_Toc152842850)

[2.3.3 Критерий Гурвица 21](#_Toc152842851)

[2.3.4 Критерий Байеса 26](#_Toc152842852)

[2.3.5 Вычисление рентабельности исследований 28](#_Toc152842853)

[2.4 Блок-схема алгоритмов решения 31](#_Toc152842854)

[2.5 Результаты 42](#_Toc152842855)

[2.6 Реализация программного продукта 43](#_Toc152842856)

[ЗАКЛЮЧЕНИЕ 49](#_Toc152842857)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 50](#_Toc152842858)

[ПРИЛОЖЕНИЕ А 51](#_Toc152842859)

ВВЕДЕНИЕ

Практически любое решение в жизни принимается человеком в условиях неопределенности, то есть при недостатке информации о существующих фактах и предстоящих событиях. Это справедливо и для фирм, принимающих решения о своих дальнейших действиях.

Первое, что нужно сделать для систематизации процесса выбора из нескольких альтернатив, это оценить выигрыши и потери, к которым приведет выбор каждой альтернативы, при условии реализации каждого из рассматриваемых сценариев будущего.

Наиболее актуально сейчас стоит вопрос о разработке решения, поэтому целью данной работы является изучение методологии разработки управленческих решений в условиях неопределенности. Для достижения данной цели необходимо выполнить следующие задачи:

1. Изучить понятие неопределенности и риска
2. Привести классификацию решений
3. Привести матрицу решений
4. Рассмотреть методы разработки принятия решения в условиях неопределенности

В данной работе рассматриваются принципы оптимальности уже давно и прочно вошедшие в теорию принятия решений − критерии Лапласа, Гурвица и Сэвиджа, а также максиминный критерий Вальда (принцип гарантированного результата).

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ
   1. Теория игр

Теория игр, раздел математики, изучающий формальные модели принятия оптимальных решений в условиях конфликта. В условиях конфликта стремление противника скрыть свои предстоящие действия порождает неопределённость. Наоборот, неопределённость при принятии решений (например, на основе недостаточных данных) можно интерпретировать как конфликт принимающего решения субъекта с природой. Поэтому Теория игр рассматривается также как теория принятия оптимальных решений в условиях неопределённости. Она позволяет математизировать некоторые важные аспекты принятия решений в технике, сельском хозяйстве, медицине и социологии. Перспективен подход с позиций Теории игр к проблемам управления, планирования и прогнозирования.

Основным в теории игр является понятие игры, являющееся формализованным представлением о конфликте. Точное описание конфликта в виде игры состоит в указании того, кто и как участвует в конфликте, каковы возможные исходы конфликта, а также кто и в какой форме заинтересован в этих исходах. Участвующие в конфликте стороны называются коалициями действия; доступные для них действия — их стратегиями; возможные исходы конфликта — ситуациями (обычно каждая ситуация понимается как результат выбора каждой из коалиций действия некоторой своей стратегии); стороны, заинтересованные в исходах конфликта, — коалициями интересов; их интересы описываются предпочтениями тех или иных ситуаций (эти предпочтения часто выражаются численными выигрышами). Конкретизация перечисленных объектов и связей между ними порождает разнообразные частные классы игр.

1.2. История возникновения теории игр

Основы теории игр зародились еще в 18 веке, с началом эпохи просвящения и развитием экономической теории. Впервые математические аспекты и приложения теории были изложены в классической книге 1944 года Джона фон Неймана и Оскара Моргенштерна «Теория игр и экономическое поведение». Первые концепции теории игр анализировали антагонистические игры, когда есть проигравшие и выигравшие за их счет игроки. Несмотря на то, что теория игр рассматривала экономические модели, вплоть до 50-х годов 20 века она была всего лишь математической теорией. После, в результате резкого скачка экономики США после второй мировой войны, и, как следствие, большего финансирования науки, начинаются попытки практического применения теории игр в экономике, биологии, кибернетике, технике, антропологии. Во время Второй мировой войны и сразу после нее теорией игр серьезно заинтересовались военные, которые увидели в ней мощный аппарат для исследования стратегических решений. В начале 50-х [Джон Нэш](http://ru.wikipedia.org/wiki/%D0%9D%D1%8D%D1%88,_%D0%94%D0%B6%D0%BE%D0%BD_%D0%A4%D0%BE%D1%80%D0%B1%D1%81) разрабатывает методы анализа, в которых все участники или выигрывают, или терпят поражение. Эти ситуации получили названия [«равновесие по Нэшу»](http://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B2%D0%BD%D0%BE%D0%B2%D0%B5%D1%81%D0%B8%D0%B5_%D0%9D%D1%8D%D1%88%D0%B0). По его теории, стороны должны использовать оптимальную стратегию, что приводит к созданию устойчивого равновесия. Игрокам выгодно сохранять это равновесие, так как любое изменение ухудшит их положение. Эти работы Нэша сделали серьезный вклад в развитие теории игр, были пересмотрены математические инструменты экономического моделирования. Джон Нэш показывает, что классический подход к конкуренции А.Смита, когда каждый сам за себя, неоптимален. Более оптимальны стратегии, когда каждый старается сделать лучше для себя, делая лучше для других. За последние 20 — 30 лет значение теории игр и интерес значительно растет, некоторые направления современной экономической теории невозможно изложить без применения теории игр. Большим вкладом в применение теории игр стала работа [Томаса Шеллинга](http://ru.wikipedia.org/wiki/%D0%A8%D0%B5%D0%BB%D0%BB%D0%B8%D0%BD%D0%B3,_%D0%A2%D0%BE%D0%BC%D0%B0%D1%81), нобелевского лауреата по экономике 2005 г. «Стратегия конфликта».

1.3. Применение теории игр

Двумя основными областями применения являются военное дело и экономика. Теоретико-игровые разработки применяются при проектировании автоматических систем управления для ракетного/противоракетного оружия, выборе форм аукционов по продаже радиочастот, прикладном моделировании закономерностей денежного обращения в интересах центральных банков, и т.п. Международные отношения и стратегическая безопасность обязаны теории игр (и теории принятия решений) в первую очередь концепцией гарантированного взаимного уничтожения.

Теорию игр применяют:

* В военном деле;
* В управлении;
* В биологии;
* В компьютерных технологиях;
* В играх(шахматах).

1.4 Методы принятия решений

Ситуация, когда результаты принимаемых решений неизвестны, считается условиями неопределённости. В общем виде, задача обоснования решения в условиях неопределенности сводится к сужению исходного множества альтернатив на основе информации, которой располагает ЛПР (лицо, принимающие решение). При принятии решений учитывается как состояние объективных условий, в которых существует фирма, так и внутренние факторы, характеризующие его потенциал.

Первое, что нужно сделать для систематизации процесса выбора из нескольких альтернатив, это оценить выигрыши и потери, к которым приведет выбор каждой альтернативы, при условии реализации каждого из рассматриваемых сценариев будущего. Все выигрыши и потери нужно свести в таблицу (или матрицу) выигрышей и потерь. В этой таблице столько строк, сколько рассматривается альтернатив, и столько столбцов, сколько сценариев будущего, определяющих результат каждой альтернативы, принимается во внимание.

Матрицу в задаче принятия решений с *m* возможными действиями и *n* состояниями природы можно представить следующим образом.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *s1* | *s2* | *…* | *sn* |
| *а1* | *ν (а1, s1)* | *ν(а1, s2)* | *…* | *ν(а1, sn)* |
| *а2* | *ν (а2, s1)* | *ν(а2, s2)* | *…* | *ν(а2, sn)* |
| *.* | *.* | *.* | *.* | *.* |
| *.* | *.* | *.* | *.* | *.* |
| *аm* | *ν (аm, s1)* | *ν(аm, s2)* | *…* | *ν(аm, sn)* |

Существует 4 критерия для анализа ситуации, связанной с принятием решений.

1. Критерий Лапласа.
2. Критерий Сэвиджа.
3. Критерий Гурвица.
4. Минимаксный (максиминный) критерий

**Критерий Лапласа** опирается на принцип недостаточного обоснования, который гласит, что поскольку распределение вероятностей состояния  неизвестно, нет причин считать их различными. Следовательно, используется оптимистическое предположение, что вероятности всех состояний природы равны между собой, то есть . Если при этом  представляет получаемую прибыль, то наилучшим решением является то, которое обеспечивает:

,

если величина  представляет собой расходы лица, принимающего решение, то оператор “max” заменяется на “min”.

**Критерий Сэвиджа** стремится смягчить консерватизм минимаксного (максиминного) критерия путем замены матрицы платежей (выигрышей или проигрышей)  матрицей потерь , которая определяется следующим образом:

**Критерий Гурвица** охватывает ряд различных подходов к принятию решений – от наиболее оптимистичного до наиболее пессимистичного (консервативного). Для описания склонности лица к оптимизму используется параметр оптимизма  Пусть величины  представляют доходы. Тогда решению, выбранному по критерию Гурвица, соответствует:

,

если , критерий Гурвица становится консервативным, так как его применение эквивалентно применению обычного минимаксного критерия. Если , критерий Гурвица становится слишком оптимистичным, ибо рассчитывает на наилучшее из наилучших условий. Степень оптимизма (или пессимизма) можно конкретизировать надлежащим выбором величины  из интервала . При отсутствии ярко выраженной склонности к оптимизму или пессимизму выбор  представляется разумным. Если величины  представляют потери, то критерий принимает следующий вид:

.

**Минимаксный (максиминный) критерий**

Процедура выбора сводится к следующему: рассматриваются результаты и в каждой альтернативе выбирается наихудший из возможных результатов, затем выбирается наилучший из них. Если величина  представляет получаемую прибыль, то в соответствии с максиминным критерием, в качестве оптимального выбирается решение, обеспечивающее

,

если величина  представляет потери, используется минимаксный критерий, который определяется следующим соотношением:

.

1. ПРАКТИЧЕСКИЙ РАЗДЕЛ
   1. Постановка задачи

Маленькая кондитерская в курортном городе продает выпечку собственного производства. Фирменные торты выпекаются каждое утро и продаются по цене $7 (при себестоимости - $3). Если торт не продается в день изготовления, он выбрасывается. Записи, которые ведет хозяйка, показывают, что за последние 100 дней спрос на эти торты имел следующее распределение.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Кол-во проданных тортов | 8 | 9 | 10 | 11 | 12 |
| Кол-во дней | 15 | 25 | 30 | 20 | 10 |

* 1. Подскажите хозяйке, какое количество тортов нужно выпекать, чтобы максимизировать прибыль?
  2. Хозяйка водит дружбу с гадалкой, которая каждый вечер предсказывает ей какое количество тортов нужно выпекать на следующий день и берет за услугу $2. Стоит ли хозяйке тратиться на гадалку?
  3. Используйте критерии максимина, минимаксного риска и максимума ожидаемой прибыли для принятия решения о партии тортов.
  4. Математическая модель

Составим таблицу решения в MS Exsel для данной задачи. В ней будет пять строк, так как хозяйка магазина может выбирать из пяти вариантов действий (приготовить 8, 9, 10, 11 или 12 тортов), и пять столбцов, так как последствия принятого решения будут определяться тем, по какому из пяти возможных сценариев станут развиваться события (составит спрос 8, 9, 10, 11 или 12 единицы продукта). В клетках таблицы укажем финансовые последствия каждого варианта решения в условиях реализации различных сценариев (прибыль хозяйки магазина). Эти последствия (результаты) рассчитаем по формуле:

Прибыль = (количество проданных продуктов×цена продажи) – (количество закупленных продуктов×себестоимость).

Если же спрос превышает сделанный заказ или соответствует ему, то мы продаем все что у нас запасено и не больше этого. В таком случае будем считать по следующей формуле:

Прибыль = (количество закупленных продуктов×цена продажи) – (количество закупленных продуктов×себестоимость).

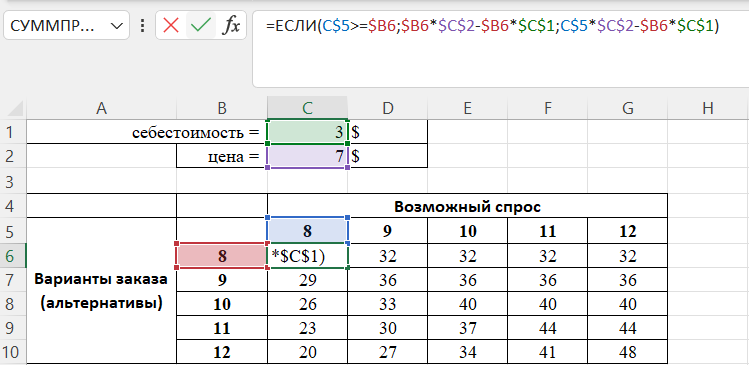
Чтобы записать одну формулу для всех случаев, используем функцию =ЕСЛИ(;;;).

В ячейку B4 запишем:

=ЕСЛИ(C$5>=$B6;$B6\*$C$2-$B6\*$C$1;C$5\*$C$2-$B6\*$C$1),

т.е. если спрос больше заказа или равен ему, используем формулу =$B7\*$C$1, а если нет – формулу = C$3\*$C$1+($B7-C$3)\*$F$1[3].

После всех вычислений получим следующую таблицу показанную на рисунке 2.1.



*Рисунок 2.1 – Платежная матрица с формулами*

Итого:

[1,1] – 7$\*8-3$\*8=32. При изготовлении 8 тортов и спросе 8 единиц, прибыль составит 32$ (восемь единиц продукции проданы по цене 7 $/шт., что в сумме составляет 56 $, и на изготовку затрачено 24$).

[1,2] – Спрос превысил сделанный заказ или в точности соответствовал ему. В этом случае мы продадим все, что у нас запасено на данный день и не больше этого и получили прибыл 32$.

[1,3]-[1,5] – аналогично.

[2,1] – 7$\*8-3$\*9 = 29. При изготовлении 9 тортов и спросе 8 единиц, прибыль составит 29$ (восемь единиц продукции проданы по цене 7 $/шт., что в сумме составляет 56 $, и на изготовку затрачено 27$).

[2,2] – 7$\*9-3$\*9 = 36. При изготовлении 9 тортов и спросе 9 единиц, прибыль составит 36$ (девять единиц продукции проданы по цене 7 $/шт., что в сумме составляет 63 $, и на изготовку затрачено 27$).

[2,3] – Спрос превысил сделанный заказ или в точности соответствовал ему. В этом случае мы продадим все, что у нас запасено на данный день и не больше этого и получили прибыл 36$.

[2,4]-[2,5] – аналогично.

[3,1] – 7$\*8-3$\*10 = 26. При изготовлении 10 тортов и спросе 8 единиц, прибыль составит 26$ (восемь единиц продукции проданы по цене 7 $/шт., что в сумме составляет 56 $, и на изготовку затрачено 30$).

[3,2] – 7$\*9-3$\*10 = 33. При изготовлении 10 тортов и спросе 9 единиц, прибыль составит 33$ (девять единиц продукции проданы по цене 7 $/шт., что в сумме составляет 63 $, и на изготовку затрачено 30$).

[3,3] – 7$\*10-3$\*10 = 40. При изготовлении 10 тортов и спросе 10 единиц, прибыль составит 40$ (десять единиц продукции проданы по цене 7 $/шт., что в сумме составляет 70 $, и на изготовку затрачено 30$).

[3,4] – Спрос превысил сделанный заказ или в точности соответствовал ему. В этом случае мы продадим все, что у нас запасено на данный день и не больше этого и получили прибыл 40$.

[3,5] – аналогично.

[4,1] – 7$\*8-3$\*11 = 23. При изготовлении 11 тортов и спросе 8 единиц, прибыль составит 23$ (восемь единиц продукции проданы по цене 7 $/шт., что в сумме составляет 56 $, и на изготовку затрачено 33$).

[4,2] – 7$\*9-3$\*11 = 30. При изготовлении 11 тортов и спросе 9 единиц, прибыль составит 30$ (девять единиц продукции проданы по цене 7 $/шт., что в сумме составляет 63 $, и на изготовку затрачено 33$).

[4,3] – 7$\*10-3$\*11 = 37. При изготовлении 11 тортов и спросе 10 единиц, прибыль составит 37$ (десять единиц продукции проданы по цене 7 $/шт., что в сумме составляет 70 $, и на изготовку затрачено 33$).

[4,4] – 7$\*11-3$\*11 = 44. При изготовлении 11 тортов и спросе 11 единиц, прибыль составит 44$ (одиннадцать единиц продукции проданы по цене 7 $/шт., что в сумме составляет 77 $, и на изготовку затрачено 33$).

[4,5] – Спрос превысил сделанный заказ или в точности соответствовал ему. В этом случае мы продадим все, что у нас запасено на данный день и не больше этого и получили прибыл 44$.

[5,1] – 7$\*8-3$\*11 = 20. При изготовлении 12 тортов и спросе 8 единиц, прибыль составит 20$ (восемь единиц продукции проданы по цене 7 $/шт., что в сумме составляет 56 $, и на изготовку затрачено 33$).

[5,2] – 7$\*9-3$\*11 = 27. При изготовлении 12 тортов и спросе 9 единиц, прибыль составит 27$ (девять единиц продукции проданы по цене 7 $/шт., что в сумме составляет 63 $, и на изготовку затрачено 33$).

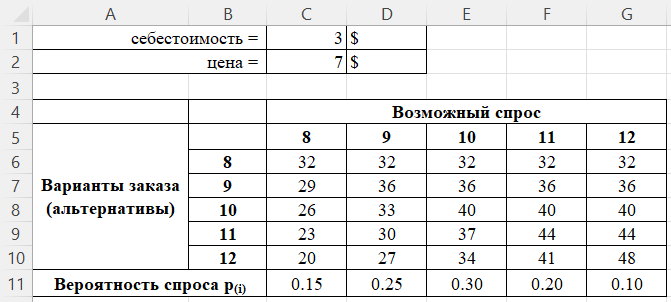
[5,3] – 7$\*10-3$\*11 = 34. При изготовлении 12 тортов и спросе 10 единиц, прибыль составит 34$ (десять единиц продукции проданы по цене 7 $/шт., что в сумме составляет 70 $, и на изготовку затрачено 33$).

[5,4] – 7$\*11-3$\*11 = 41. При изготовлении 12 тортов и спросе 11 единиц, прибыль составит 41$ (одиннадцать единиц продукции проданы по цене 7 $/шт., что в сумме составляет 77 $, и на изготовку затрачено 33$).

[5,5] – 7$\*12-3$\*12 = 48. При изготовлении 12 тортов и спросе 12 единиц, прибыль составит 48$ (двенадцать единиц продукции проданы по цене 7 $/шт., что в сумме составляет 84 $, и на изготовку затрачено 36$).

В тексте задачи имеются данные о том, сколько раз наблюдался тот или иной сценарий (спрос), и по ним можно рассчитать относительную частоту, с которой каждый из них реализуется, и таким образом оценить вероятность каждого варианта развития событий, *p(j)*, :

Таким образом получаем следующую платежную матрицу, (матрицу прибыли) представленную на рисунке 2.2.



*Рисунок 2.2 – Платежная матрица*

Согласно вышеприведенной таблице, если мы приготовим, например, 11 тортов, то с вероятностью 0,15 получим 23$. С вероятностью 0,25 получим 30$, с вероятностью 0,30 – 37$, с вероятностью 0,20 спрос будет соответствовать количеству приготовленных тортов, и мы получим 44$, и, наконец, с вероятностью 0,10 спрос превысит количество приготовленных тортов, и мы получим те же 44$.

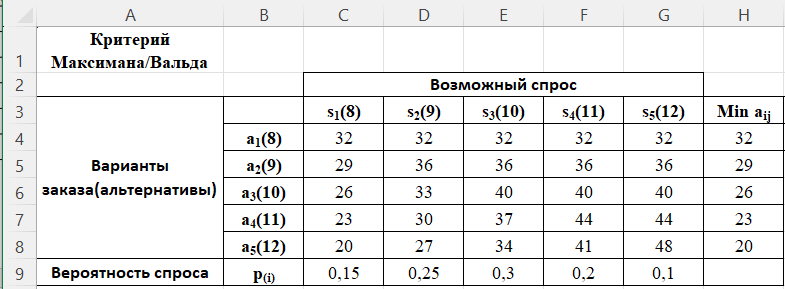
* 1. Выбор метода принятия решения
     1. Критерий Вальда

По критерию Вальда за оптимальную принимается чистая стратегия, которая в наихудших условиях гарантирует максимальный выигрыш, т.е.

a = max(min aij).

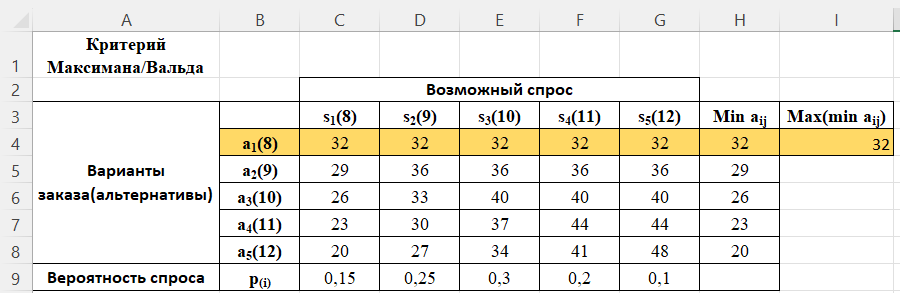
Критерий Вальда ориентирует статистику на самые неблагоприятные состояния природы, т.е. этот критерий выражает пессимистическую оценку ситуации. Также этот критерий называется критерий «Максимина», так как мы из худшего выбираем лучшее.

Вычислим минимальные элементы в строках нашей платежной матрицы с помощью функции МИН(), результат на рисунке 2.3, данные занесем в столбец G.



*Рисунок 2.3 – Вычисление минимального элемента в строках матрицы*

Находим максимальный элемент среди минимальных с помощью функции МАКС(). Результат поиска показан на рисунке 2.4.



*Рисунок 2.4 – Результат поиска максимального элемента среди минимальных*

Вывод: соответственно, оптимальным в данном случае является решение производить 8 единиц продукта (использовать стратегию а1), причем ожидаемая прибыль составляет в этом случае 32$.

2.3.2 Критерий Сэвиджа

Критерий минимакса настолько “пессимистичен”, что иногда может приводить к нелогичным выводам. Рассмотрим следующую матрицу потерь, которая обычно приводится в качестве классического примера для обоснования необходимости использования “менее пессимистичного” критерия Сэвиджа или по-другому критерий минимаксного риска.

.

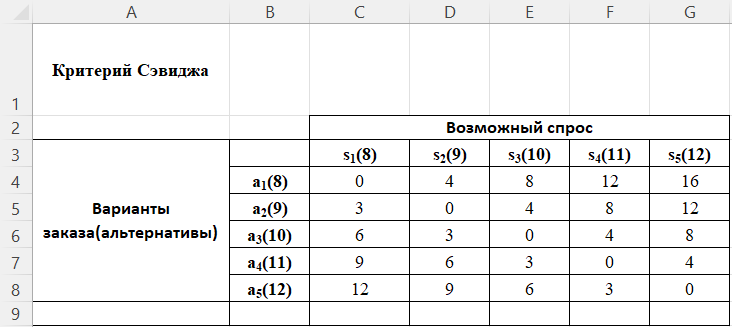
Критерий Сэвиджа ориентирует статистику на самые неблагоприятные состояния природы, т.е. этот критерий выражает пессимистическую оценку ситуации.

Находим матрицу рисков.

;

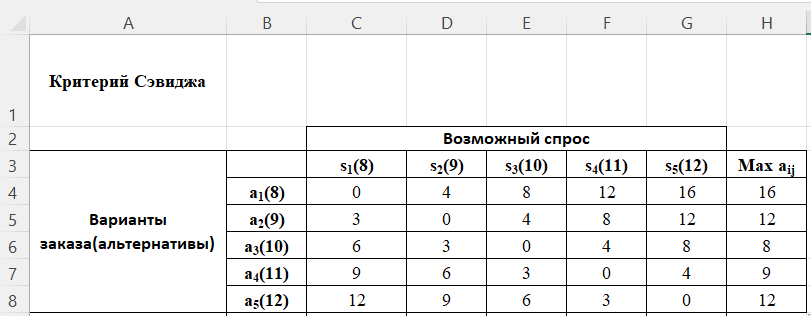
.

Для того, чтобы построить таблицу рисков необходимо в каждом столбце найти наибольший элемент. Затем от максимального элемента в столбце вычитаем значение элемента исходной матрицы и, таким образом, заполняем матрицу рисков (рисунок 2.5).



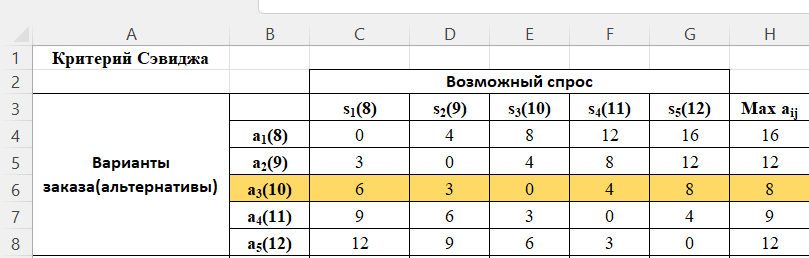
*Рисунок 2.5 – Создание новой матрицы рисков для критерия Гурвица*

После построения матрица рисков ищем максимальный элемент в каждой строке с помощью функции МАКС() результат представлен на рисунке 2.6.



*Рисунок 2.6 – Поиск максимального элемента в матрице рисков*

Далее ищем минимальный элемент среди максимальных элементов (рисунок 2.7).



*Рисунок 2.7 – Поиск минимального элемента среди максимальных для критерия Сэвиджа*

Вывод: итак, согласно критерию Сэвиджа оптимальной стратегией является стратегия a3, при которой хозяйка будет изготавливать по 10 тортов. Критерий Сэвиджа тоже крайне пессимистический, и в смысле «пессимизма» он сходен с критерием Вальда.

2.3.3 Критерий Гурвица

Этот критерий охватывает ряд различных подходов к принятию решений: от наиболее оптимистичного до наиболее пессимистичного. Стараясь занять наиболее уравновешенную позицию, можно ввести оценочный коэффициент, называемый коэффициентом пессимизма, который находится в интервале [0, 1] и отражает ситуацию, промежуточную между точкой зрения крайнего оптимизма и крайнего пессимизма. За оптимальную принимается та стратегия, для которой выполняется соотношение:

.

Здесь α − некоторое число, 0 ≤α ≤1 , характеризующее степень оптимизма игрока. Тогда число 1−α можно понимать как степень пессимизма. При α = 0 в каждой строке во всех столбцах стоит одно и то же число, равное максимальному выигрышу при выборе соответствующей строке стратегии, то есть самый оптимистичный результат. При α = 1, то новая матрица совпадает с матрицей выигрышей, и критерий Гурвица превращается в критерий Вальда; при α = 0 – в критерий «крайнего оптимизма», при других значениях α получается нечто среднее – ближе к оптимизму, если α близко к 0, или ближе к пессимизму, если α близко к 1.

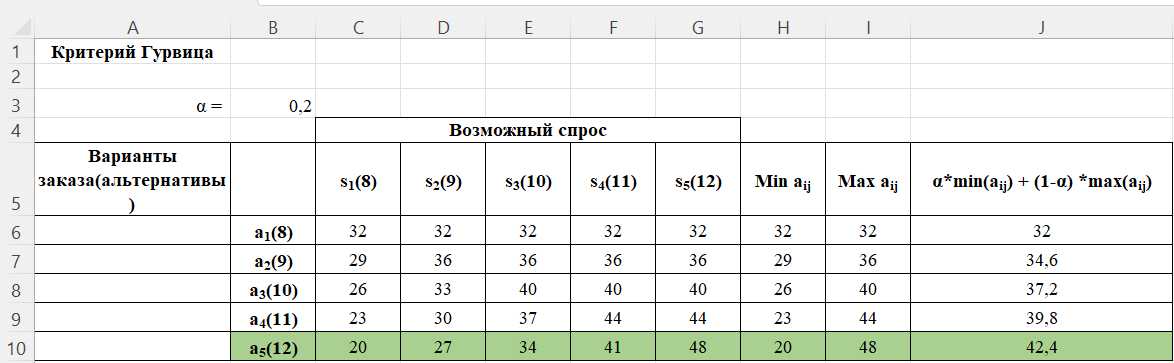
Рассмотрим несколько вариантов применения критерия Гурвица при различных значениях θ .

При θ = 0.2 матрица имеет следующий вид, показанный на рисунке 2.8:



*Рисунок 2.8 – Решение задачи с помощью критерия Гурвица при коэффициенте оптимизма 0,2*

Находим максимальный элемент в последнем столбце нашей матрицы (рисунок 2.9).



*Рисунок 2.9 – Поиск решения при помощи критерия Гурвица*

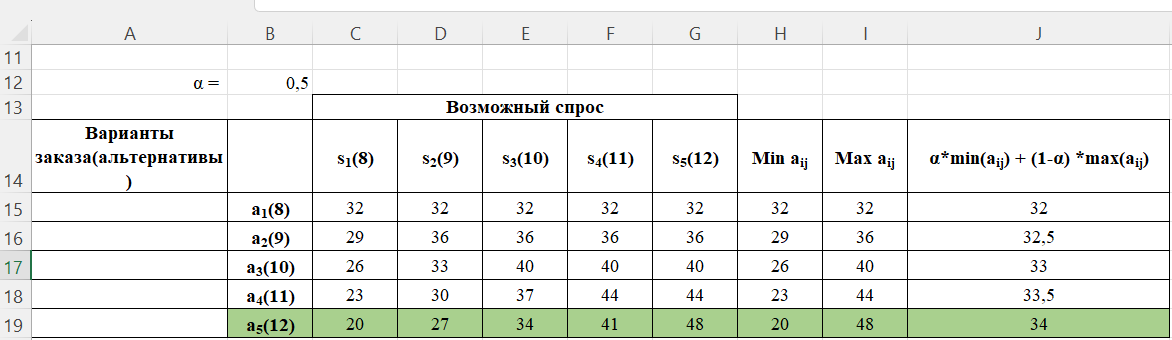
Вывод: максимальное значение находиться в последней строке матрицы и , значит оптимальной по данному критерию является стратегия a5 – изготавливать по 12 тортов.

При θ = 0.5 матрица имеет следующий вид, матрица представлена на рисунке 2.10:



*Рисунок 2.10 - Решение задачи с помощью критерия Гурвица при коэффициенте оптимизма 0,5*

Находим максимальный элемент в последнем столбце нашей матрицы (рисунок 2.11).



*Рисунок 2.11 – Поиск решения при помощи критерия Гурвица*

Вывод: максимальное значение находиться в последней строке матрицы и , значит оптимальной по данному критерию является стратегия a5 – изготавливать по 12 тортов.

При θ = 0.8 матрица имеет следующий вид, представленная на рисунке 2.12:



*Рисунок 2.12 - Решение задачи с помощью критерия Гурвица при коэффициенте оптимизма 0,8*

Находим максимальный элемент в последнем столбце нашей матрицы (рисунок 2.13).



*Рисунок 2.13 – Поиск решения при помощи критерия Гурвица*

Вывод: максимальное значение находиться в последней строке матрицы и , значит оптимальной по данному критерию является стратегия a1 – изготавливать по 8 тортов.

Таким образом, мы рассмотрели критерий Гурвица с различными коэффициентами оптимизма. При α = 0,8 критерий слишком оптимистичный, при α = 0,2 он слишком пессимистичный. Значение α между 0 и 1 может определяться в зависимости от склонности нашей задачи к пессимизму или оптимизму. При отсутствии ярко выраженной склонности α=0,5 представляется наиболее разумным.

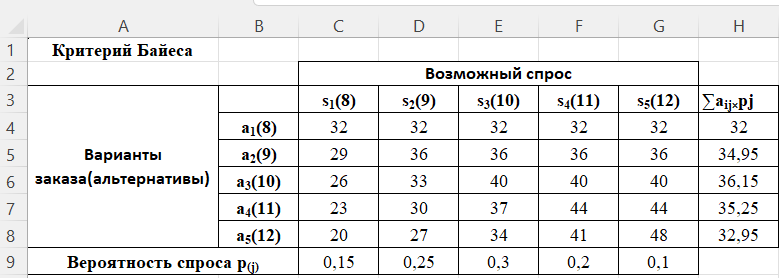
Вывод: учитывая, что самым разумным в нашей задаче будет считаться α = 0,5, то по критерию Гурвица следует придерживаться стратегии a5, и выпекать по 12 тортов.

2.3.4 Критерий Байеса

По критерию Байеса за оптимальные принимается та стратегия (чистая) ai, при которой максимизируется средний выигрыш. Критерий Байеса (принцип математического ожидания) предполагает полное доверие ЛПР известным вероятностям состояний окружающей среды. Следовательно, данная задача – это задача принятия решения в условиях риска. Также критерий Байеса является частным случаем критерия Лапласа. Лучше решение определяется по формуле:

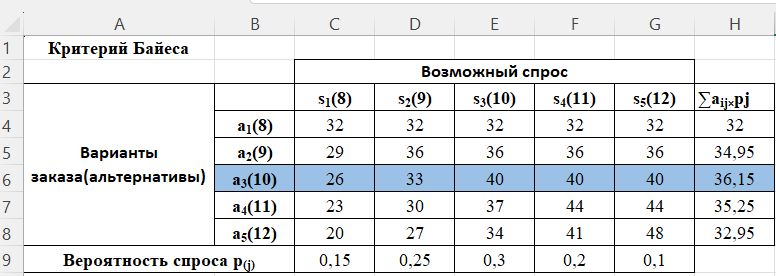
.

Считаем значения ∑(aijpj), полученная матрица представлена на рисунке 2.14.



*Рисунок 2.14 – Матрица для решения задачи методом Байеса*

Теперь находим максимальный элемент из столбца H (рисунок 2.15):



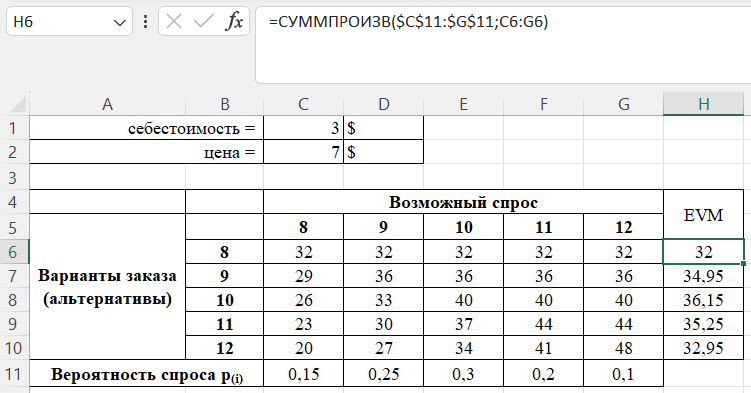
*Рисунок 2.15 – Решение матрицы с помощью критерия Байеса*

Вывод: проанализировав выбираем стратегию изготовления 10 тортов, максимальная ожидаемая прибыль 36,15$.

2.3.5 Вычисление рентабельности исследований

Используя выше полученные данные, можно оценить средний взвешенный финансовый результат EMV для каждой альтернативы (значения размера заказа). Рассчитаем величину EMV для каждой альтернативы, используя функцию =СУММПРОИЗВ(..). Для заказа в 8 тортов функция будет иметь вид: =СУММПРОИЗВ($C$11:$G$11;C6:G6). Ссылка на строку вероятностей фиксирована. Поместим эту формулу в ячейку H6 и протянем вниз до ячейки H10.

Величина EMV, таблица с подсчетами представлена на рисунке 2.16, с ростом заказа меняется немонотонно: сначала растет от 32$ до 36$, а затем уменьшается до 32,95$. Максимальная величина средней прибыли – 36,15$ – соответствует заказу 10 тортов.

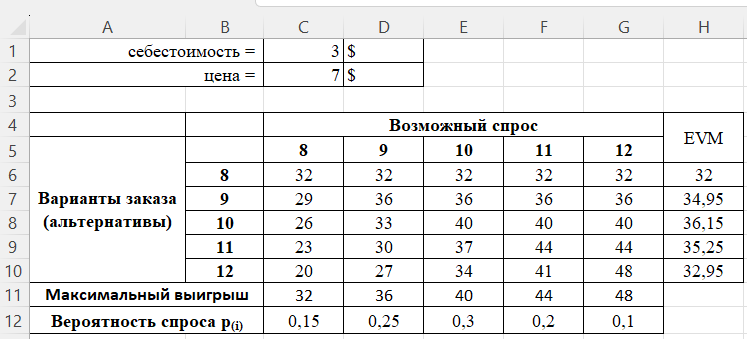


*Рисунок 2.16 – Таблица с расчетами EVM*

Таким образом для получения максимальной прибыли хозяйке кондитерской необходимо выпекать каждый день по 10 тортов.

Для того чтобы определить стоит ли хозяйке кондитерской платить гадалке по 2$, посчитаем стоимость совершенной информации.

Сначала в сроке С11:G11 определим максимальные выигрыши при каждом сценарии будущего, используя функцию =МАКС(..). Формула будет выглядеть следующим образом: =МАКС(C6:C10). Результат выполнения представлен на рисунке 2.17.



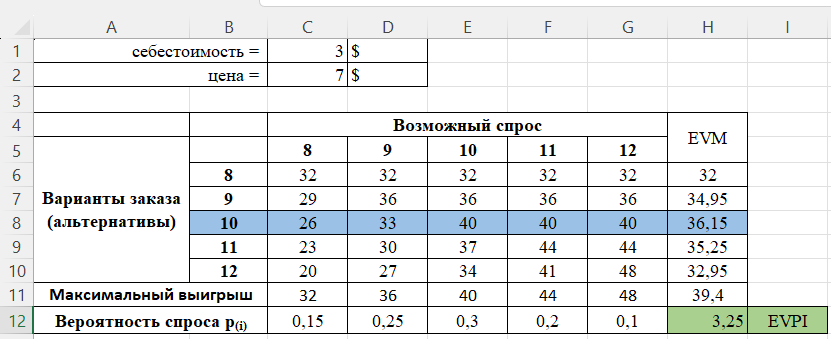
*Рисунок 2.17 – Поиск максимальной прибыли в каждом столбце*

Затем считаем ожидаемую монетарную ценность (так как вероятности не меняются). Для этого просто протянем формулу в ячейке H10 вниз (рисунок 2.18).



*Рисунок 2.18 – Считаем ожидаемую монетарную ценность*

Высчитываем на основе максимального значения в EVM, которое мы берем за 100%, и ожидаемой монетарной ценности. В результате, представленном на рисунке 2.19, увеличивается наша ожидаемая прибыль всего на 8,9% (получим 36,15$ вместо 39,4$). Эта величина и есть наш источник совершенной информации (EVPI) – верхняя граница цены, которую мы готовы платить.



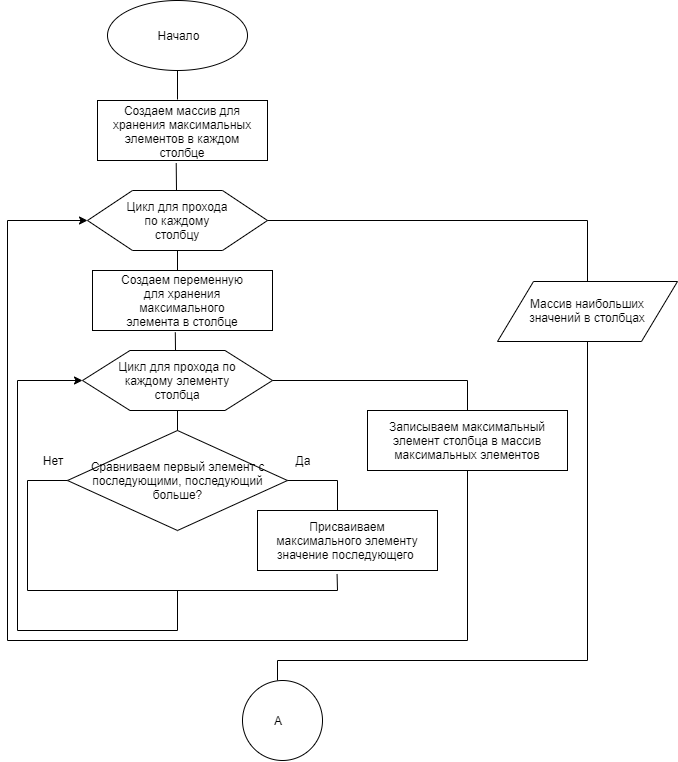
*Рисунок 2.19 – Считаем верхнюю границу, которая равна источнику совершенной информации*

А так как гадалке надо платить каждый день по 2$, то можно сделать вывод, что пользоваться её услугами не стоит.

* 1. Блок-схема алгоритмов решения

**Блок-схема для критерия Сэвиджа:**

* Находим максимальные элементы в столбцах матрицы.



*Рисунок 2.20 – Блок-схема для критерия Сэвиджа(поиск максимальных элементов в столбцах)*

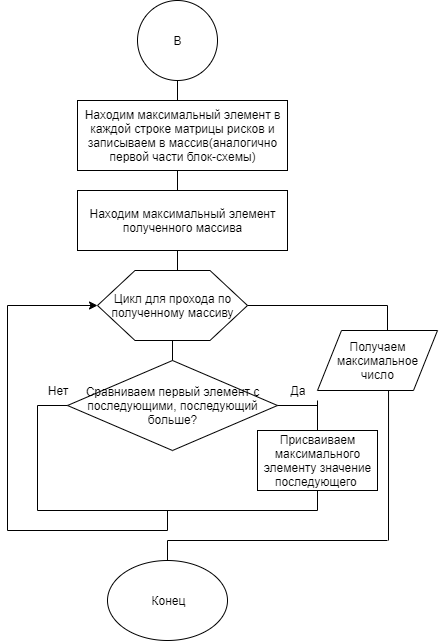
* Построим матрицу рисков, каждый элемент находим по формуле:

*.*



*Рисунок 2.21 - Блок-схема для критерия Сэвиджа(построение матрицы рисков)*

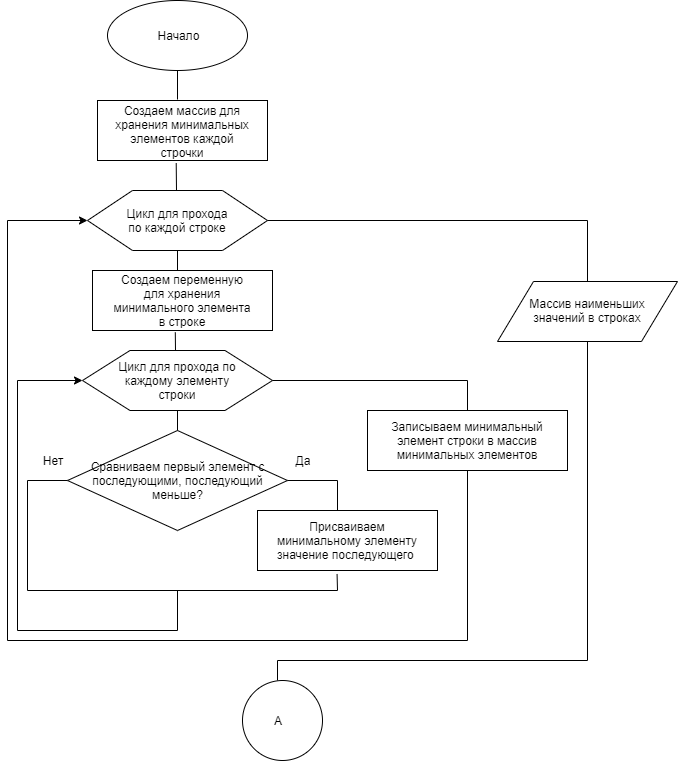
* Находим в матрице рисков максимальный элемент в каждой строке;
* Находим максимальный элемент среди максимальных в каждой строке



*Рисунок 2.22 - Блок-схема для критерия Сэвиджа(нахождение максимального элемента в каждой строке матрицы рисков и максимального числа из них)*

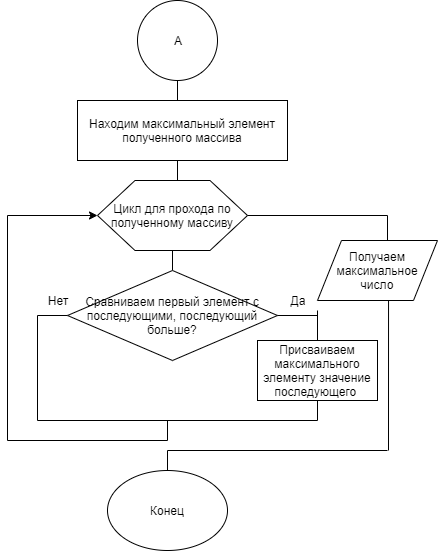
**Блок-схема для максиминного критерия/критерия Вальда:**

* Находим минимальные элементы в каждой строке



*Рисунок 2.23 – Блок-схема для максиминного критерия(поиск минимальных элементов каждой строки)*

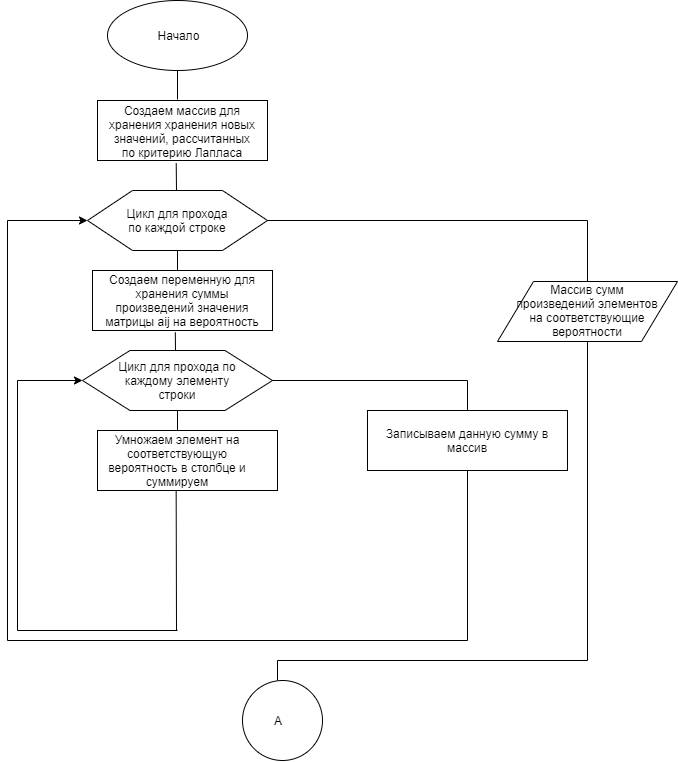
* Находим максимальный элемент из полученного массива максимальных

****

*Рисунок 2.24 – Блок-схема для максиминного критерия(поиск максимального элемента из минимальных)*

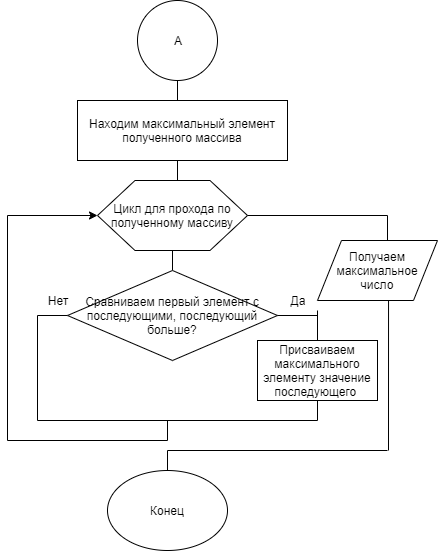
**Блок-схема для критерия Байеса:**

* Создаем масив для хранения сумм произведений элементов строк на вероятности

****

*Рисунок 2.25 – Блок-схема для критерия Байеса(считаем суммы произведений элементов строки на вероятность)*

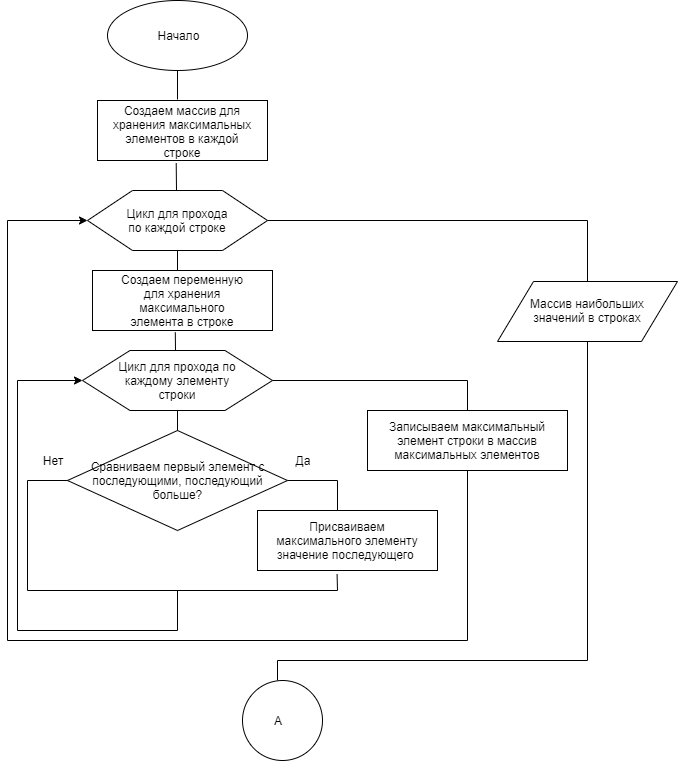
* Находим максимальное значение среду полученных сумм

****

*Рисунок 2.26 – Блок-схема для критерия Байеса (поиск максимального элемента из полученных сумм)*

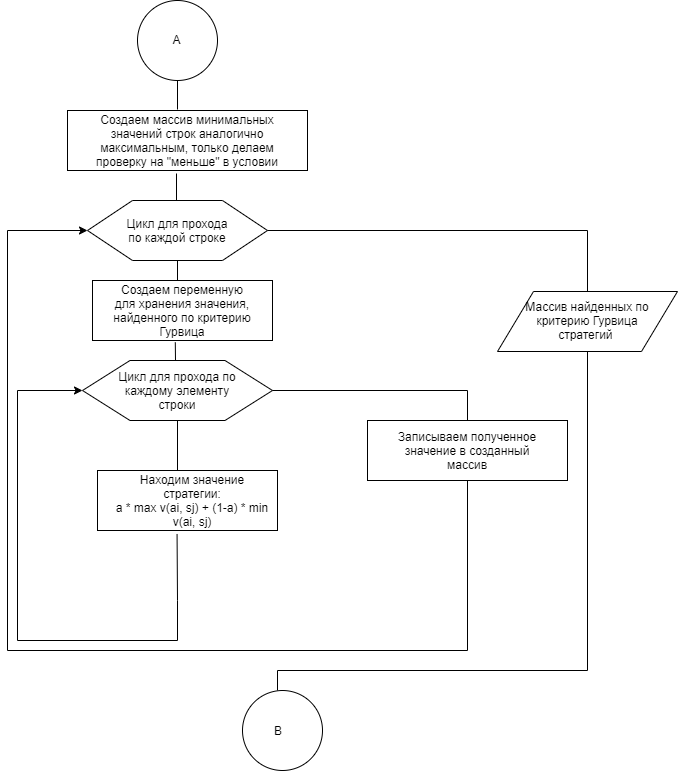
**Блок-схема для критерия Гурвица:**

* Находим максимальные элементы в строках

****

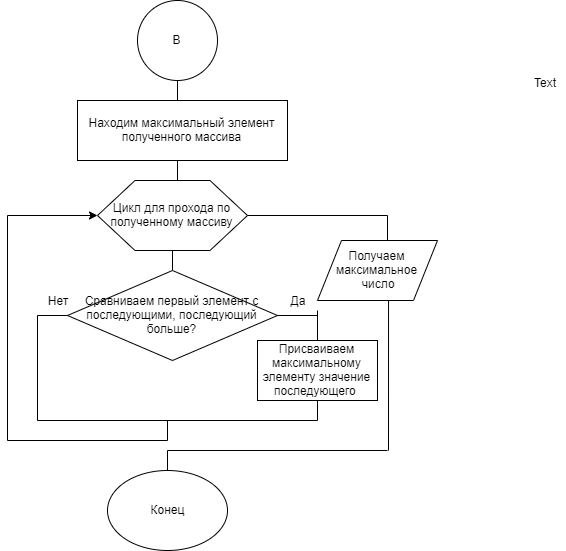
*Рисунок 2.27 – Блок-схема для критерия Гурвица(поиск максимальных элементов в строках)*

* Находим минимальные элементы в строках
* Считаем новые значения стратегий по критерию Гурвица, по формуле:

****

*Рисунок 2.28 – Блок-схема для критерия Гурвица(поиск минимальных элементов в строках и расчет значений по формуле)*

* Находим максимальный элемент из полученных стратегий

****

*Рисунок 2.29 – Блок-схема для критерия Гурвица(поиск максимального элемента из полученных значений)*

* 1. Результаты

В результате выполнения данной задачи различными методами, получаем следующие результаты:

• По критерию Вальда оптимальной стратегией является формирование заказа величиной 8 тортов, т.е первая стратегия;

• По критерию Байеса оптимальными стратегиями являются формирования заказа величиной 10 тортов, т.е третья стратегия;

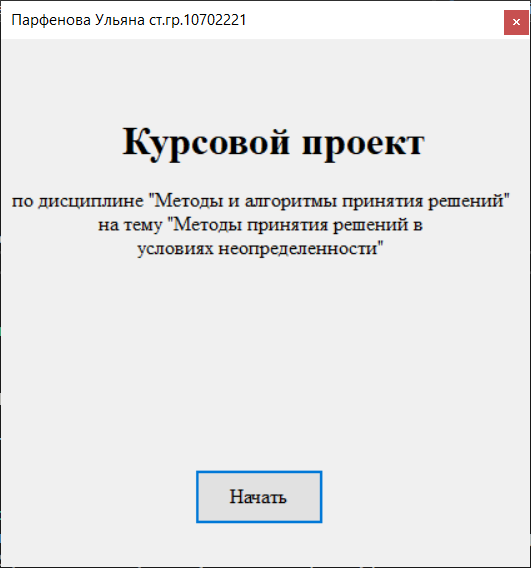
• По критерию Сэвиджа оптимальной стратегией является формирование заказа величиной 10 тортов, т.е третья стратегия;

• По критерию Гурвица оптимальной стратегией является формирование заказа величиной 12 тортов, т.е пятая стратегия.

Исходя из полученных результатов, сделаем вывод, что оптимальной стратегией является формирование заказа величиной 10 тортов, так как данная стратегия оптимальна по 2 критериям: Байеса и Сэвиджа.

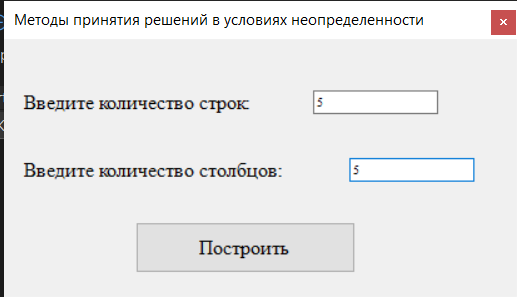
* 1. Реализация программного продукта

Реализация программного продукта осуществлялась на языке C# с помощью .Net Framework. Главное окно программы имеет следующий вид, представленный на рисунке 2.30. Полный код программы представлен в ПРИЛОЖЕНИИ А.



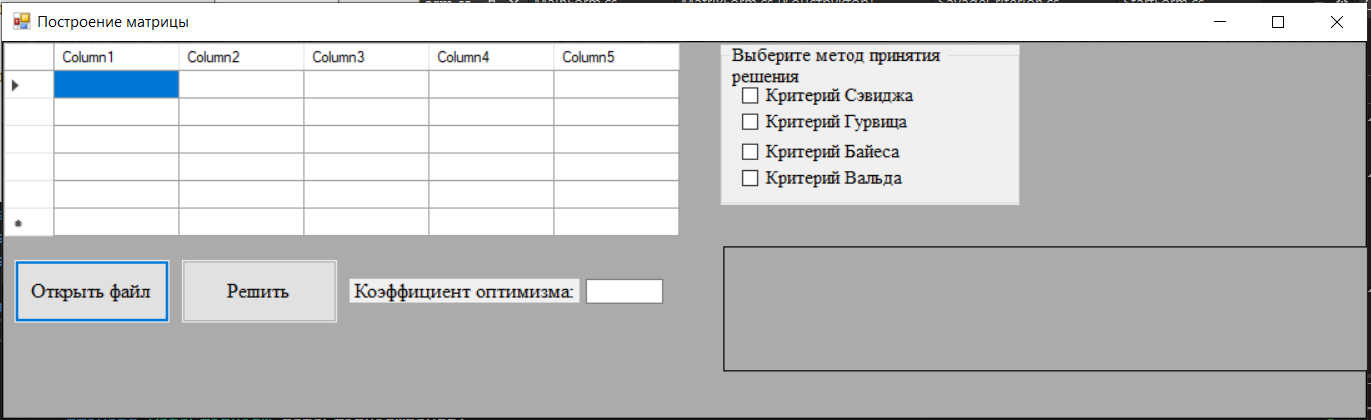
*Рисунок 2.30 – Главное окно программы*

После этого пользователю необходимо ввести размер нашей будущей матрицы, окно для ввода представлено на рисунке 2.31.

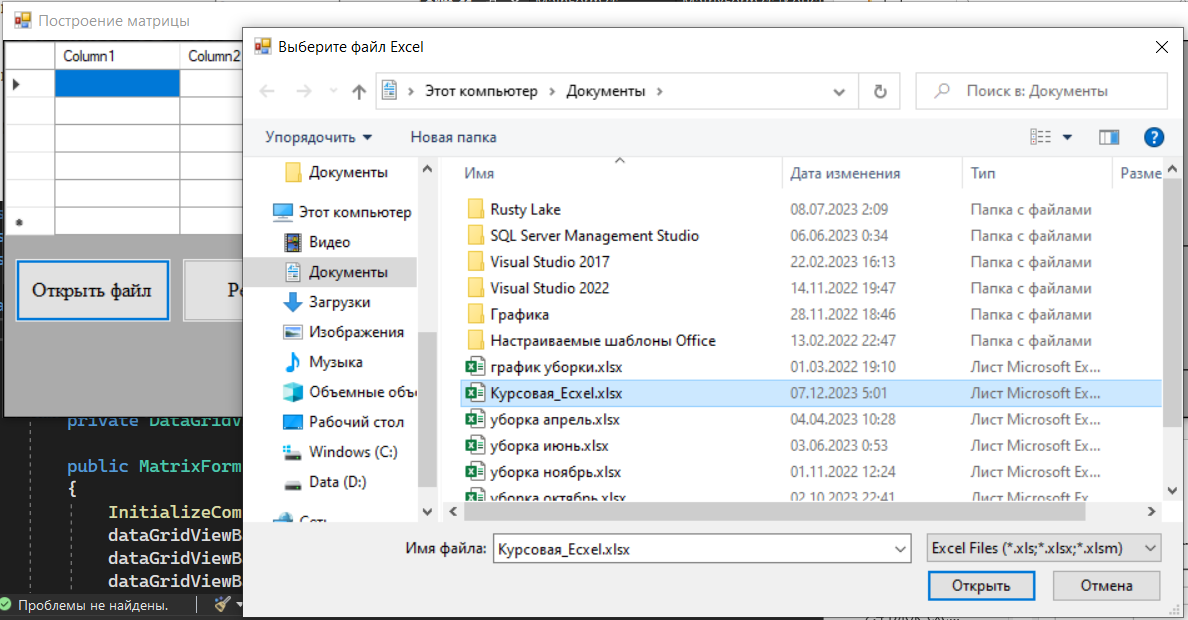


*Рисунок 2.31 – Окно для ввода размера платежной матрицы*

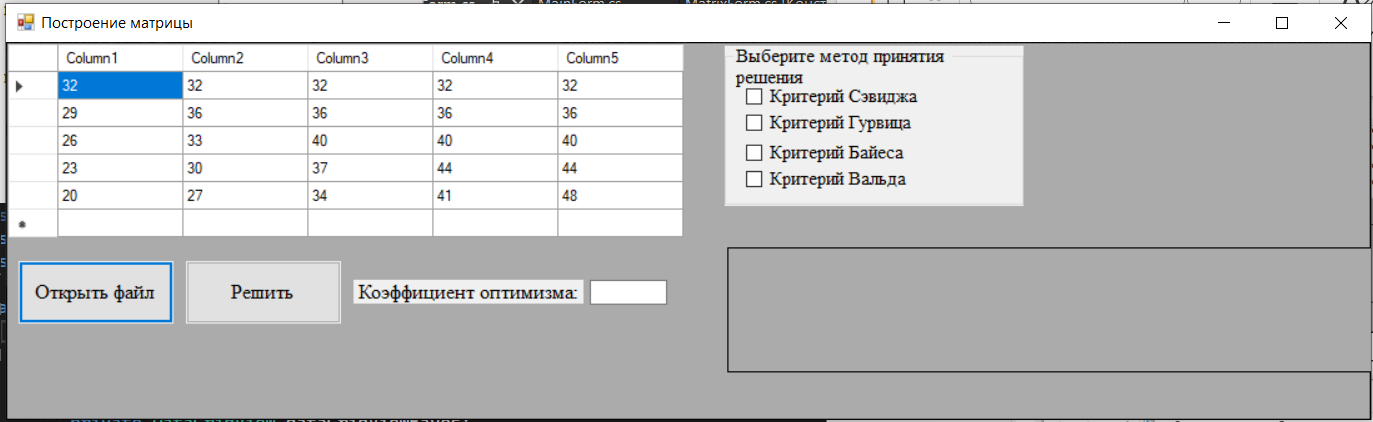
Далее открывается окно для построения матрицы (рисунок 2.32), в котором пользователь может вести матрицу или же загрузить данные из Microsoft Excel (рисунок 2.33). Также пользователь выбирает каким методом необходимо решить нашу матрицу и при необходимости вводит дополнительные параметры (рисунок 2.34).



*Рисунок 2.32 – Окно для построения и расчета платежной матрицы с помощью различных критериев*

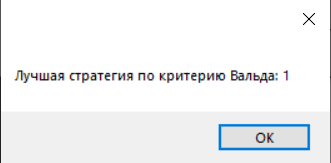


*Рисунок 2.33 – Выбор файла для заполнения матрицы*



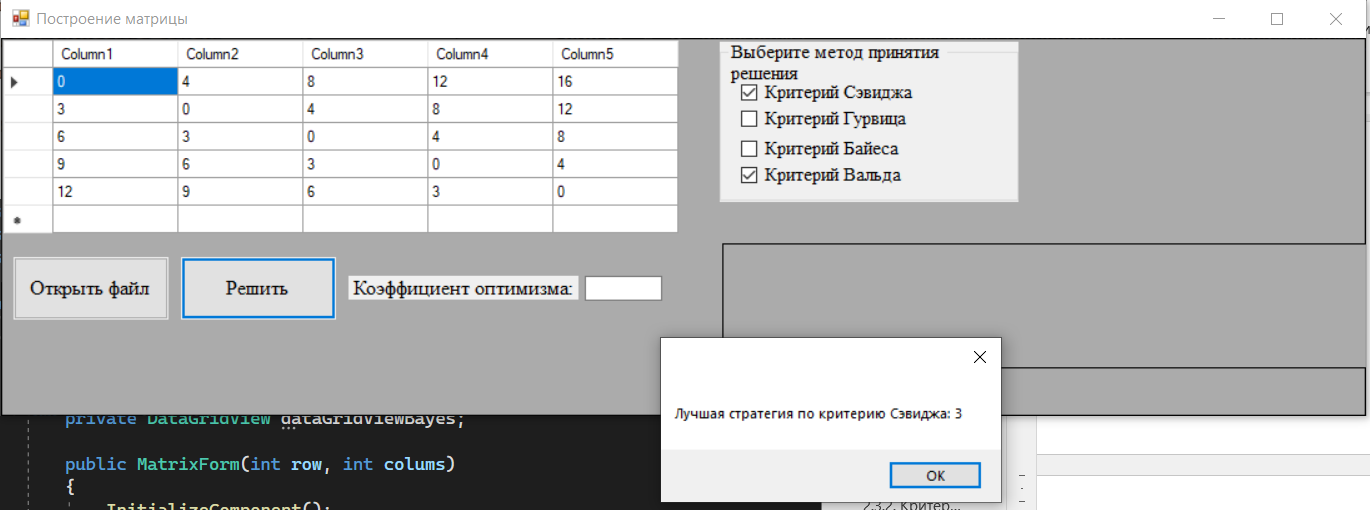
*Рисунок 2.34 – Выбор критерия для решения задачи и ввода дополнительных данных*

Решение матрицы с помощью критерия Вальда, результат показан на рисунке 2.35.



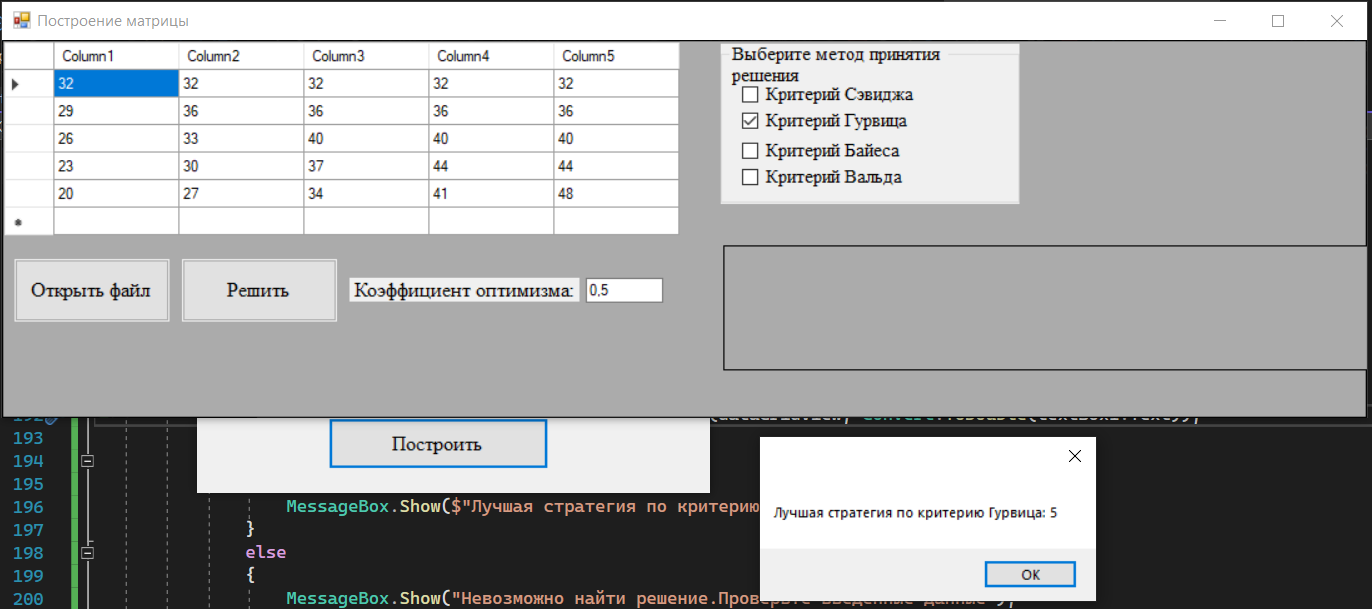
*Рисунок 2.35 – Ответ к задаче при помощи критерия Вальда*

Решение матрицы с помощью критерия Сэвиджа, результат показан на рисунке 2.36.



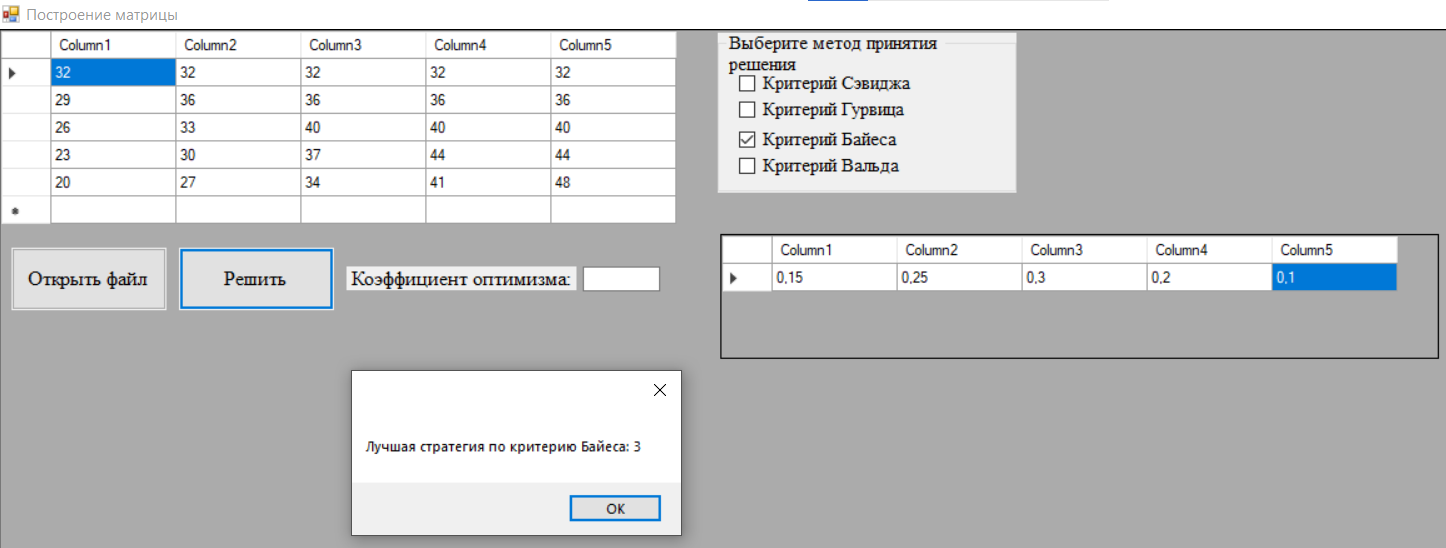
*Рисунок 2.36 – Ответ к задаче при помощи критерия Сэвиджа*

Решение матрицы с помощью критерия Гурвица, результат представлен на рисунке 2.37.



*Рисунок 2.37 – Ответ к задаче при помощи критерия Гурвица*

Решение матрицы с помощью критерия Байеса, результат представлен на рисунке 2.38.



*Рисунок 2.38 – Ответ к задаче при помощи критерия Байеса*

ЗАКЛЮЧЕНИЕ

Теория игр – математический метод изучения оптимальной стратегии в играх. Данный предмет имеет множество экономических приложений и это знание может помочь в оценке многих ситуаций. Теория игр позволяет оценивать игровые стратегии участников и выбрать лучший из предлагаемых вариантов.

Именно теория матричных игр позволяет нам рассматривать и с легкостью решать задачи принятия решений в ситуациях с несколькими участниками, когда значение целевой функции для каждого зависит также от решений принимаемых остальными участниками. Поэтому важную роль в матричных играх отводится конфликтам и совместным действиям.

Теория матричных игр широко нашла свое применение для анализа проблем микроэкономики, а также и в других сферах.

В результате выполнения данной курсовой работы были изучены основные понятия теории игр, классификация игр, история возникновения самой теории и отрасли ее применения. Рассмотрены основные критерии нахождения оптимальной стратегии, такие как максиминный(Вальда), критерий Байеса, критерий Гурвица, критерий Сэвиджа. Была решена задача с помощью каждого решения, из решения которой был сделан вывод, что оптимальной величиной формирования заказа является 10 тортов. Также был разработан программный продукт, реализующий алгоритм нахождения оптимальной стратегии на базе изученных критерией.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Андрианова А.А., Хабибуллин Р.Ф. Принятие решений в условиях неопределенности / А.А. Андрианова, Р.Ф. Хабибуллин. – Казань: Казан. ун-т, 2015. – 25 с.
2. Дадаков, И. А. Методы и модели принятия решений: учебное пособие / И. А. Дадаков, Л. В. Васильева. – Казань: Изд-во Казан. ун-та, 2017. – 120 с.
3. Зайцев М.Г., Варюхин С.Е. Методы оптимизации управления и принятия решений: примеры, задачи, кейсы: учебное пособие. - 2-е изд., испр. - М.: Издательство "Дело" АНХ, 2008. - 664 с.
4. Критерий Байеса в теории принятия решений [Электронный ресурс]. – Режим доступа: https://scicenter.online/prinyatiya-resheniy-teoriya-scicenter/kriteriy-bayesa-145942.html (дата обращения: 10.01.2023).
5. Макаров, А. А. Методы исследования в теории принятия решений / А. А. Макаров. – М.: Изд-во МГУ, 2005. – 300 с.
6. Самойлюкович, В. В. С17 Количественные методы принятия управленческих решений: методическое пособие для студентов спецшльности 1-26 02 01 «Бизнес-администрирование» : в 3 ч. / В. В. Самойлюкович. - Минск: БИТУ, 2013- . - Ч. 1 ; Принятие решений в условиях неопределенности и риска. — 2013. - 89 с.
7. Силкина Г.Ю. Теория принятия решений и управление рисками. Модели конфликтов, неопределенности, риска.: Учеб. пособие. СПб.: Изд-во СПбГПУ, 2003. 72 с.
8. Смирнов, А. Н. Теория принятия решений: учебник / А. Н. Смирнов, А. В. Широков. – М.: Эксмо, 2009. – 240 с.
9. Петров, В. Б. Принятие решений в условиях неопределенности / В. Б. Петров, Г. В. Сидоров. – СПб.: БХВ-Петербург, 2010. – 180 с.
10. Принятие решений в условиях неопределенности: учебное пособие / под ред. А. С. Ильина. – М.: Финансы и статистика, 2015. – 160 с.

ПРИЛОЖЕНИЕ А

**Файл StartForm.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Kursework

{

public partial class StartForm : Form

{

public StartForm()

{

InitializeComponent();

}

private void openMainFormButton\_Click(object sender, EventArgs e)

{

MainForm mainForm = new MainForm();

mainForm.FormClosed += MainForm\_FormClosed;

this.Hide();

mainForm.Show();

}

private void MainForm\_FormClosed(object sender, FormClosedEventArgs e)

{

Application.Exit();

}

}

}

**Файл MainForm.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Kursework

{

public partial class MainForm : Form

{

public MainForm()

{

InitializeComponent();

}

private void closeButton\_Click(object sender, EventArgs e)

{

// Закройте вторую форму. Это вызовет событие FormClosed.

this.Close();

}

private void createMatrixButton\_Click(object sender, EventArgs e)

{

// Проверка корректности введенных данных.

if (int.TryParse(textBoxRow.Text, out int rows) && int.TryParse(textBoxColum.Text, out int columns))

{

// Создание новой формы с матрицей и передача параметров.

MatrixForm matrixForm = new MatrixForm(rows, columns);

matrixForm.ShowDialog();

}

else

{

MessageBox.Show("Введите корректные значения для строк и столбцов.");

}

}

}

}

**Файл MatrixForm.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using Excel = Microsoft.Office.Interop.Excel;

namespace Kursework

{

public partial class MatrixForm : Form

{

private DataGridView dataGridView;

private DataGridView dataGridViewBayes;

public MatrixForm(int row, int colums)

{

InitializeComponent();

dataGridViewBayes = new DataGridView();

dataGridViewBayes.Location = new System.Drawing.Point(groupBox1.Location.X+365, groupBox1.Location.Y-20);

dataGridViewBayes.Size = new System.Drawing.Size(colums\*115, 100);

dataGridViewBayes.AllowUserToAddRows = false;

Controls.Add(dataGridViewBayes);

dataGridView = new DataGridView();

dataGridView.Dock = DockStyle.Fill;

Controls.Add(dataGridView);

// Создание и отображение матрицы.

CreateAndDisplayMatrix(row, colums);

buttonFile.Location = new System.Drawing.Point(10, row\*35);

groupBox1.Location = new System.Drawing.Point(colums\*115, dataGridView.Location.Y+3);

buttonSolution.Location = new System.Drawing.Point(buttonFile.Width + buttonFile.Location.X + 10, buttonFile.Location.Y);

label1.Location = new System.Drawing.Point(buttonFile.Width + buttonFile.Location.X + buttonSolution.Width+20, buttonFile.Location.Y+15);

textBox1.Location = new System.Drawing.Point(buttonFile.Width + buttonFile.Location.X + buttonSolution.Width + label1.Width+25, buttonFile.Location.Y + 15);

}

private void CreateAndDisplayMatrix(int rows, int columns)

{

// Создание DataTable для хранения данных матрицы.

var dataTable = new System.Data.DataTable();

// Добавление столбцов в DataTable.

for (int i = 0; i < columns; i++)

{

dataTable.Columns.Add($"Column{i + 1}", typeof(int));

}

// Добавление строк в DataTable.

for (int i = 0; i < rows; i++)

{

dataTable.Rows.Add();

}

// Назначение DataTable элементу DataGridView.

dataGridView.DataSource = dataTable;

}

private void ReadExcelButton\_Click(object sender, EventArgs e)

{

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "Excel Files|\*.xls;\*.xlsx;\*.xlsm";

openFileDialog.Title = "Выберите файл Excel";

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

string filePath = openFileDialog.FileName;

// Чтение данных из Excel файла.

DataTable dataTable = ReadDataFromExcel(filePath);

// Проверка соответствия размеров матрицы и таблицы.

if ((dataTable.Rows.Count+1) == dataGridView.Rows.Count &&

dataTable.Columns.Count == dataGridView.Columns.Count)

{

// Запись данных в таблицу DataGridView.

for (int i = 0; i < dataTable.Rows.Count; i++)

{

for (int j = 0; j < dataTable.Columns.Count; j++)

{

dataGridView.Rows[i].Cells[j].Value = dataTable.Rows[i][j];

}

}

}

else

{

MessageBox.Show("Размеры матрицы в файле Excel не соответствуют размерам таблицы.");

}

}

}

private DataTable ReadDataFromExcel(string filePath)

{

DataTable dataTable = new DataTable();

Excel.Application excelApp = new Excel.Application();

Excel.Workbook workbook = excelApp.Workbooks.Open(filePath);

Excel.Worksheet worksheet = (Excel.Worksheet)workbook.Sheets[1];

int rowCount = worksheet.UsedRange.Rows.Count;

int colCount = worksheet.UsedRange.Columns.Count;

for (int i = 1; i <= rowCount; i++)

{

if (i == 1)

{

for (int j = 1; j <= colCount; j++)

{

dataTable.Columns.Add("Column" + j, typeof(int));

}

}

DataRow row = dataTable.NewRow();

for (int j = 1; j <= colCount; j++)

{

row[j - 1] = worksheet.Cells[i, j].Value;

}

dataTable.Rows.Add(row);

}

workbook.Close(false);

excelApp.Quit();

return dataTable;

}

private void CheckBox3\_CheckedChanged(object sender, EventArgs e)

{

// Проверяем состояние CheckBox

if (checkBox3.Checked)

{

// Создаем таблицу с 1 строкой и количеством столбцов, равным столбцам изначальной матрицы

CreateBayesTable();

}

else

{

// Если CheckBox не выбран, очищаем таблицу

dataGridViewBayes.DataSource = null;

}

}

private void CreateBayesTable()

{

// Очищаем существующую таблицу, если она есть

dataGridViewBayes.DataSource = null;

// Создаем DataTable с количеством столбцов, равным столбцам изначальной матрицы

DataTable dataTable = new DataTable();

foreach (DataGridViewColumn column in dataGridView.Columns)

{

dataTable.Columns.Add(column.HeaderText);

}

// Добавляем пустую строку

dataTable.Rows.Add(dataTable.NewRow());

// Устанавливаем таблицу данных для DataGridView

dataGridViewBayes.DataSource = dataTable;

}

private void CheckBox\_CheckedChanged(object sender, EventArgs e)

{

// Обработчик события изменения состояния для всех CheckBox'ов

// Проверяем состояние каждого CheckBox и вызываем соответствующий метод

if (checkBox1.Checked)

{

int decision = SavageSolver.FindBestStrategy(dataGridView);

if (decision != -1)

{

MessageBox.Show($"Лучшая стратегия по критерию Сэвиджа: {decision}");

}

else

{

MessageBox.Show("Невозможно найти решение.Проверьте введенные данные");

}

}

if (checkBox2.Checked)

{

int decision = GurvicaSolver.FindBestStrategy(dataGridView, Convert.ToDouble(textBox1.Text));

if (decision != -1)

{

MessageBox.Show($"Лучшая стратегия по критерию Гурвица: {decision}");

}

else

{

MessageBox.Show("Невозможно найти решение.Проверьте введенные данные");

}

}

if (checkBox3.Checked)

{

int decision = BayesSolver.FindBestStrategy(dataGridView,dataGridViewBayes);

if (decision != -1)

{

MessageBox.Show($"Лучшая стратегия по критерию Байеса: {decision}");

}

else

{

MessageBox.Show("Невозможно найти решение.Проверьте введенные данные");

}

}

if (checkBox4.Checked)

{

int decision = WaldSolver.FindBestStrategy(dataGridView);

if (decision != -1)

{

MessageBox.Show($"Лучшая стратегия по критерию Вальда: {decision}");

}

else

{

MessageBox.Show("Невозможно найти решение.Проверьте введенные данные");

}

}

}

}

}

**Файл SavageCriterion.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Kursework

{

using System;

using System.Data;

using System.Linq;

using System.Windows.Forms;

public class SavageSolver

{

public static int FindBestStrategy(DataGridView dataGridView)

{

if (dataGridView == null || dataGridView.Rows.Count == 0 || dataGridView.Columns.Count == 0)

{

throw new ArgumentException("Неправильно заполнена таблица");

}

int rowCount = (dataGridView.Rows.Count-1);

int colCount = dataGridView.Columns.Count;

// Создаем массив для хранения максимальных значений в каждом столбце

double[] maxValues = new double[colCount];

// Находим максимальные значения в каждом столбце

for (int j = 0; j < colCount; j++)

{

double maxValue = double.MinValue;

for (int i = 0; i < rowCount; i++)

{

if (dataGridView.Rows[i].Cells[j].Value != null &&

double.TryParse(dataGridView.Rows[i].Cells[j].Value.ToString(), out double cellValue))

{

if (cellValue > maxValue)

{

maxValue = cellValue;

}

}

}

maxValues[j] = maxValue;

}

// Отнимаем максимальные значения от каждого элемента в соответствующем столбце

for (int j = 0; j < colCount; j++)

{

for (int i = 0; i < rowCount; i++)

{

if (dataGridView.Rows[i].Cells[j].Value != null &&

double.TryParse(dataGridView.Rows[i].Cells[j].Value.ToString(), out double cellValue))

{

dataGridView.Rows[i].Cells[j].Value = maxValues[j]-cellValue;

}

}

}

// Находим стратегию, минимизирующую максимальные потери

double[] maxRowValues = new double[rowCount];

// Находим максимальные значения в каждой строке

for (int i = 0; i < rowCount; i++)

{

double maxRowValue = double.MinValue;

for (int j = 0; j < colCount; j++)

{

if (dataGridView.Rows[i].Cells[j].Value != null &&

double.TryParse(dataGridView.Rows[i].Cells[j].Value.ToString(), out double cellValue))

{

if (cellValue > maxRowValue)

{

maxRowValue = cellValue;

}

}

}

maxRowValues[i] = maxRowValue;

}

// Находим максимальное значение в массиве максимальных значений строк

double minMaxRowValue = double.MaxValue;

int bestStrategy = -1;

for (int i = 0; i < rowCount; i++)

{

if (maxRowValues[i] < minMaxRowValue)

{

minMaxRowValue = maxRowValues[i];

bestStrategy = i+1;

}

}

return bestStrategy;

}

}

}

**Файл WaldSolve.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Kursework

{

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Windows.Forms;

public class WaldSolver

{

public static int FindBestStrategy(DataGridView dataGridView)

{

if (dataGridView == null || dataGridView.Rows.Count == 0 || dataGridView.Columns.Count == 0)

{

throw new ArgumentException("Неверно построена таблица");

}

int rowCount = dataGridView.Rows.Count-1;

int colCount = dataGridView.Columns.Count;

// Создаем массив для хранения минимальных значений в каждой строке

double[] minValuesInRows = new double[rowCount];

// Находим минимальные значения в каждой строке

for (int i = 0; i < rowCount; i++)

{

double minValueInRow = double.MaxValue;

for (int j = 0; j < colCount; j++)

{

if (dataGridView.Rows[i].Cells[j].Value != null &&

double.TryParse(dataGridView.Rows[i].Cells[j].Value.ToString(), out double cellValue))

{

if (cellValue < minValueInRow)

{

minValueInRow = cellValue;

}

}

}

minValuesInRows[i] = minValueInRow;

}

// Находим максимальное минимальное значение в строках

double maxMinValue = double.MinValue;

int bestStrategy = -1;

for (int i = 0; i < rowCount; i++)

{

if (minValuesInRows[i] > maxMinValue)

{

maxMinValue = minValuesInRows[i];

bestStrategy = i+1;

}

}

return bestStrategy;

}

}

}

**Файл GurvicaSolve.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Kursework

{

using System;

using System.Data;

using System.Linq;

using System.Windows.Forms;

public class GurvicaSolver

{

public static int FindBestStrategy(DataGridView dataGridView, double optimismCoefficient)

{

if (dataGridView == null || dataGridView.Rows.Count == 0 || dataGridView.Columns.Count == 0)

{

throw new ArgumentException("Неверно заполнена таблица");

}

if (optimismCoefficient < 0 || optimismCoefficient > 1)

{

throw new ArgumentException("Коэффициент оптимизма должен быть от 0 до 1");

}

int rowCount = dataGridView.Rows.Count-1;

int colCount = dataGridView.Columns.Count;

// Создаем массив для хранения взвешенных значений в каждой строке

double[] weightedValuesInRows = new double[rowCount];

// Находим взвешенные значения в каждой строке

for (int i = 0; i < rowCount; i++)

{

double minValueInRow = double.MaxValue;

double maxValueInRow = double.MinValue;

for (int j = 0; j < colCount; j++)

{

if (dataGridView.Rows[i].Cells[j].Value != null &&

double.TryParse(dataGridView.Rows[i].Cells[j].Value.ToString(), out double cellValue))

{

if (cellValue < minValueInRow)

{

minValueInRow = cellValue;

}

if (cellValue > maxValueInRow)

{

maxValueInRow = cellValue;

}

}

}

// Взвешенное значение: optimismCoefficient \* maxValue + (1 - optimismCoefficient) \* minValue

weightedValuesInRows[i] = optimismCoefficient \* minValueInRow + (1 - optimismCoefficient) \* maxValueInRow;

}

// Находим стратегию с максимальным взвешенным значением

double maxWeightedValue = double.MinValue;

int bestStrategy = -1;

for (int i = 0; i < rowCount; i++)

{

if (weightedValuesInRows[i] > maxWeightedValue)

{

maxWeightedValue = weightedValuesInRows[i];

bestStrategy = i+1;

}

}

return bestStrategy;

}

}

}

**Файл BayesSolve.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Kursework

{

using System;

using System.Data;

using System.Linq;

using System.Windows.Forms;

public class BayesSolver

{

public static int FindBestStrategy(DataGridView dataGridView, DataGridView dataGridViewProbabilities)

{

if (dataGridView == null || dataGridView.Rows.Count == 0 || dataGridView.Columns.Count == 0)

{

throw new ArgumentException("Неверно задана таблица");

}

if (dataGridViewProbabilities == null || dataGridViewProbabilities.Rows.Count != 1 || dataGridViewProbabilities.Columns.Count != dataGridView.Columns.Count)

{

throw new ArgumentException("Неверно введены вероятности");

}

int rowCount = dataGridView.Rows.Count-1;

int colCount = dataGridView.Columns.Count;

// Инициализируем массив для хранения суммы вероятностей умноженных на значения в каждой строке

double[] sumProbabilities = new double[rowCount];

// Читаем вероятности из DataGridViewProbabilities

double[] probabilities = new double[colCount];

for (int j = 0; j < colCount; j++)

{

if (dataGridViewProbabilities.Rows[0].Cells[j].Value != null &&

double.TryParse(dataGridViewProbabilities.Rows[0].Cells[j].Value.ToString(), out double probability))

{

probabilities[j] = probability;

}

else

{

throw new ArgumentException($"Invalid probability value in column {j}");

}

}

// Рассчитываем сумму вероятностей умноженных на значения в каждой строке

for (int i = 0; i < rowCount; i++)

{

for (int j = 0; j < colCount; j++)

{

if (dataGridView.Rows[i].Cells[j].Value != null &&

double.TryParse(dataGridView.Rows[i].Cells[j].Value.ToString(), out double cellValue))

{

sumProbabilities[i] += cellValue \* probabilities[j];

}

}

}

// Находим стратегию с максимальной суммой

double maxSumProbability = double.MinValue;

int bestStrategy = -1;

for (int i = 0; i < rowCount; i++)

{

if (sumProbabilities[i] > maxSumProbability)

{

maxSumProbability = sumProbabilities[i];

bestStrategy = i+1;

}

}

return bestStrategy;

}

}

}