# Содержание

# Введение

Работа АПК для достаточного обеспечения населения страны продуктами питания всегда была важной составляющей деятельности правительства РФ. Для эффективной работы этой сферы нужно всегда вырабатывать грамотные стратегии по производству или переработке. Например, выработка лучшей стратегии становится особенно важной, когда ставится задача о выработке такого стратегического продукта как сахар, который в нашей стране изготавливается из сахарной свеклы. Объем ее переработки в сутки даже средним по мощности сахарным заводом равен приблизительно 3 тысячи тонн. Большие объемы перерабатываемого сырья при этом способствует тому, что применение квазиоптимальных стратегий переработки дает существенный выигрыш относительно переработки в произвольном порядке. Данная лабораторная работа посвящена разработке программы системы поддержки принятия решений, целью которой будет являться анализ различных стратегий по плане переработки сахарной свеклы, максимизирующего выход сахара.

В теоретической части будет изучен такой тип задач, как задача о назначениях, а также некоторые её разновидности. Кроме этого, будут рассмотрены некоторые алгоритмы решения данной задачи.

Практическая часть будет представлять из себя реализацию изученных ранее стратегий на языке программирования Python. Реализация будет проходить в контексте написания программы системы поддержки принятия решений, которая, помимо алгоритмов, будет иметь интерфейс. Программа будет моделировать процесс переработки сахара в зависимости от исходных данных. На выходе пользователь получит отчет по каждой стратегии, а именно: потери каждой стратегии, график зависимости количества сахара по дням. Кроме того, программа, опираясь на полученные результаты, предложит пользователю лучшую стратегию переработки.

# 1. Теоретическая часть

## 1.1. Задачи о назначениях и методы их решения

### 1.1.1. Постановка задачи о назначениях

Пусть дана матрица C размера n×n с неотрицательными компонентами $c_{ij}$, где элемент $c_{ij}$ с соответствует стоимости выполнения j-го вида работ i-ым работником. Необходимо найти такое соответствие работ исполнителям (одну работу одному исполнителю), чтобы расходы на оплату труда были наименьшими. Если цель состоит в нахождении назначения с наибольшей стоимостью, то решение сводится к решению аналогичной задачи путём замены каждой стоимости $c_{ij}$ на разность между максимальной стоимостью и $c_{ij}$ , т.е. матрица себестоимостей требует преобразования по правилу $c'_{ij} = \max\{c_{ij}\} - c_{ij}$. Матрицу C обычно называют весовой (или платежной), мы далее будем называть ее матрицей состояний, так как в различных задачах о назначениях матрица C определяет различные составляющие, например, эффективность. Выбор строки относительно столбца описывается перестановкой:

Целевая функция имеет вид:

$$S(\sigma) = c_{\sigma(1)1} + c_{\sigma(2)2} + \cdots + c_{\sigma(n)n} \tag{1}$$

Оптимизационная задача состоит в том, чтобы найти такую перестановку чисел от 1 до n (последовательность переработки имеющихся партий сырья), при которой значение S($\sigma$ ) , будет минимальным (максимальным). Назовем перестановку , при которой функция S($\sigma$ ) принимает максимальное значение – максимальной, а принимает минимальное значение – минимальной. Перестановка гарантирует выбор 11 элементов, такой, что в каждой строке и в каждом столбце выбран только один элемент, входящий в формулу S($\sigma$ ).

### 1.1.2. Венгерский алгоритм решения задачи о назначениях

Задача о назначениях является частным случаем транспортной задачи. При ее решении методом полного перебора потребуется O(n!) операций. Но есть и более приспособленные для этой задачи алгоритмы. Одним из первых и самых известных является венгерский алгоритм. Он был открыт в 1955 году, чуть позже было доказано, что его полиномиальная сложность $O(n^4)$ , а еще спустя время этот алгоритм был модифицирован до полиномиальной сложности $O(n^3)$.

Алгоритм.

1. Если задача решается на максимум, то в каждой строке матрицы необходимо найти максимальный элемент, его же вычесть из каждого элемента соответствующей строки и умножить всю матрицу на (-1) . Если задача решается на минимум, то этот шаг необходимо пропустить.

2. В каждой строке матрицы состояний находят минимальный элемент и вычитают его из всех элементов строки.

3. В каждом столбце матрицы состояний находят минимальный элемент и вычитают его из всех элементов столбца. 12

4. Находят строку с одним 0, заключают этот 0 в квадрат и называют отмеченным. В столбце, содержащем отмеченный 0, все 0 зачёркивают и дальше не рассматривают. Продолжают шаг, пока возможно.

5. Находят столбец с одним 0 и отмечают найденный 0. В строке, содержащей отмеченный 0, все остальные 0 зачёркивают. Продолжают шаг, пока возможно.

6. Если после выполнения шагов 4-5 остались неотмеченные 0, то отмечают любой из них, а в строке и столбце, содержащих отмеченный 0, все остальные 0 зачёркивают.

7. Оптимальное решение достигнуто в том случае, если в каждой строке и в каждом столбце матрицы есть ровно один отмеченный 0. В противном случае проводят минимальное число пересекающихся вертикальных и горизонтальных прямых через все 0.

8. Среди не зачёркнутых прямыми чисел ищут минимальное, вычитая его из всех не зачёркнутых чисел и прибавляя ко всем числам, стоящим на пересечении двух (вертикальной и горизонтальной) прямых. К полученной матрице применяют алгоритм, начиная с шага 4.

### 1.1.3. Постановка нечеткой задачи о назначениях

Поставим модифицированную задачу о назначениях. Для известной матрицы C найти перестановку $\sigma$ чисел от 1 до n , такую, что значение целевой функции S($\sigma$ ) было бы максимальным. Кроме того, значения $\sigma$ (s) выбираются последовательно (сначала $\sigma$ (1) , затем $\sigma$ (2) и т.д.). Выбор значения $\sigma$ (j) происходит, основываясь на знании только первых j столбцов матрицы C (в момент выбора $\sigma$ (j) , остальные n-j столбцов матрицы C неизвестны).

Поставим нечеткую задачу о назначениях. Предположим, заданы константы $c_{min}, c_{max}, b_{min}, b_{max}$ . Назовем матрицу C допустимой, если $c_{i1} \in [c_{min,} c_{max}]$, $i = \overline{1,n}$, $b_{ij} \in [b_{min,} b_{max}]$, $i = \overline{1,n}$, $j = \overline{1,n-1}$,, а сама она строится по формуле:

$$c_{ij} = c_{i1} \cdot b_{i1} \dots b_{ij-1},\ i = \overline{1,n}, j = \overline{2,n}$$

Если в модифицированной задаче допустимая матрица C задается альтернативным способом, элементы $c_{i1}$, $i = \overline{1,n}$ , известны, а параметры $b_{ij-1}$, $i = \overline{1,n}$, становятся известными в момент выбора $\sigma$ (j) , то задача преобразуется в модифицированную нечеткую, или в просто нечеткую, задачу о назначениях.

### 1.1.4. Методы решения нечеткой задачи о назначениях

Для поставленной нечеткой задачи необходимо осуществлять поиск других, эвристических методов. Ниже приведены известные эвристические методы:

1. «Жадный» (Greedy) метод заключается в использовании «жадного» алгоритма. Выбирается $c_{ij}$ – наибольший элемент j -го столбца матрицы C из «невыбранных» строк, то есть $i \neq \sigma(s)$ для всех $s = \overline{1, j-1}$.

2. «Бережливый» (Thrifty) метод использует противоположный по смыслу жадному – «бережливый» алгоритм: выбирается $c_{ij}$ – наименьший 25 элемент j столбца матрицы из «невыбранных» строк, то есть $i \neq \sigma(s)$ для всех $s = \overline{1, j-1}$.

3. «Жадно-Бережливый» и «Бережливо-Жадный» методы – смешанные методы, когда, например, с 1 по $\nu - 1$ столбец используется один из методов, приведенных выше, а с $\nu$ по n столбец – другой, $2 \leq \nu \leq n-1$. Если это непонятно из текста (например, предложенной задачи), то будем указывать число $\nu$ рядом с названием этого метода. Например, жадно-бережливый-$\nu$ метод.

4. Метод GTC. Опишем его с помощью теоремы:

Пусть для параметров матрицы С выполняются следующие условия:

1) $c_{11} = c_{21} = \cdots = c_{n1}$
2) $b_{ij} = b_i,\ i = \overline{1,n},\ 2 \leq \nu \leq \left[\frac{n}{2}\right],\ j = \overline{1, \nu - 1}$
3) $b_{ij} = \overline{b_i},\ \overline{b_i} < 1,\ i = \overline{1,n},\ \ j = \overline{\nu, n}$
4) $1 < b_1 < b_2 < \cdots < b_n < \left(\frac{n-1}{n-2}\right)^{\frac{n}{n+1}}$
5) Изначально известен порядок 4)
6) $\overline{b_k} = \varphi_k b_k^{-1},\ \varphi_k = b_k^{-s}$, где фиксированная константа $s \in \left[0, \frac{1}{n}\right]$

При данных условиях перестановка $\omega$ – оптимальная, здесь $\omega$ (1) = n - 2$\nu$ +3,

$\omega$ (2) = n - 2$\nu$ +5,..., ω($\nu$ -1) = n -1,$\omega$ ($\nu$ ) = n,

$\omega$ ($\nu$ +1) = n - 2, $\omega$ ($\nu$ + 2) = n - 4...,$\omega$ (2$\nu$ -1) = n - 2$\nu$ + 2 , $\omega$ (2$\nu$ )= n - 2$\nu$ +1,...,$\omega$ (n -1) = 2, $\omega$ (n) =1

# 2. Практическая часть

## 2.1. Постановка задачи о нахождении плана переработки сахарной свеклы, максимизирующего выход сахара

Предположим, что за единицу времени (период, этап) завод перерабатывает фиксированное количество (партию) сырья массы M. Величина M определяется производственными мощностями завода. Имеется сырье (сахарная свекла разных сортов), и в каждую единицу времени (этап) перерабатывается сырье только одного сорта (одна партия). Обозначим n – количество партий свеклы, которые завод перерабатывает за «сезон». Предположим, что к началу сезона переработки все сырье находится на темпоральных хранилищах. Многие сахарные заводы особенно в начале сезона (при благоприятных погодных условиях) не хранят свеклу, а сразу после сбора везут ее для переработки на завод. Для облегчения математической модели мы будем считать, что к началу переработки вся свекла убрана и находится на кагатных полях, откуда и доставляется на сахарный завод.

Занумеруем партии сырья, некоторым образом, от 1 до n. Обозначим $c_{i1}$ – долю полезного ингредиента в i -й партии сырья в начале переработки, $c_{i2}$ – его содержание в i -й партии свеклы в начале второго этапа переработки, …, $c_{in}$ – содержание полезного ингредиента в i -й партии свеклы в начале n-го этапа. Для сахарной свеклы полезный ингредиент – это сахар.

Параметры $c_{ij}$ удобно определять через начальную долю полезного ингредиента $c_{i1} = a_i \in [a_{min},\ a_{max}]$ и коэффициенты деградации $b_{ij-1} = \frac{c_{ij}}{c_{ij-1}}$, (альтернативное задание матрицы C). Соответствующие формулы для $c_{ij}$ имеют следующий вид $c_{i1} = a_i$, $c_{ij} = a_i b_{i1} \dots b_{ij-1}$, $i = \overline{1,n}$, $j = \overline{2,n}$.

Параметры $c_{ij}$ составляют матрицу C , которая будем называть матрицей состояний. Если в течении хранения доля сахара постоянно уменьшается (свекла деградирует), то $b_{ij} \in [\beta_1, \beta_2] \subset (0,1)$ , где $[\beta_1, \beta_2]$ – некий отрезок, а матрица C – строчно-антитонная.

Для упрощения математической модели будем считать, что число этапов, во время которых происходит дозаривание одинаково для всех партий сахарной свеклы. Также считаем, что процесс дозаривания длится от первого этапа до начала $\nu$ этапа, $2 \le \nu \le \left[\frac{n}{2}\right]$. 58 На этапах дозаривания параметры $b_{ij} \in (1, \beta_{max}]$, $i = \overline{1,n}$, $j = \overline{1, \nu - 1}$, на этапах увядания параметры $b_{ij} \in [\beta_1, \beta_2] \subset (0,1)$, $j = \overline{1, \nu - 1}$. Рекомендуется брать $\beta_{max} = \frac{n-1}{n-2}$. В математической модели нужно определить: учитывается процесс дозаривания или нет. Этот случай соответствует строчно-изоантитонной матрице состояний.

Параметры $a_i$ и $b_{ij}$ могут распределяться по-разному на заданных отрезках. Особо важным является распределение коэффициентов деградации $b_{ij}$ . Будем различать два принципиальных момента:

1) равномерное распределение параметров $b_{ij}$ на отрезке $[\beta_1, \beta_2]$;

2) концентрированное распределение: при каждом i существует константа $\delta_i, \delta_i \leq \frac{|\beta_2-\beta_1|}{4}$ такая, что $b_{ij} \in [\beta_i^1, \beta_i^2] \subset [\beta_1, \beta_2]$, где $|\beta_i^1 - \beta_i^2| = \delta_i$ На отрезке $[\beta_i^1, \beta_i^2]$ параметры $b_{ij}$ распределены равномерно.

Иначе говоря, в этом случае коэффициенты деградации каждой партии лежат в маленькой окрестности некой точки, то есть деградация свеклы происходит «приблизительно одинаково» на каждом этапе.

Для этапов дозаривания концентрированное распределение определяется аналогичным образом.

Сформулируем задачу поиска оптимальной последовательности переработки имеющихся партий сырья, которая обеспечивает для перерабатывающего завода максимальный выход конечного продукта за весь сезон переработки сырья. Имеется матрица состояний $\overline{S} = C - \overline{L}$ с положительными элементами $s_{ij}$ . Матрица $\overline{L}$ с неотрицательными элементами $l_{ij}$ отвечает за те или иные потери сахара во время переработки.

Последовательность переработки партий сырья будем описывать перестановкой натуральных чисел от 1 до n. При реализации этой последовательности общий выход продукции после переработки всех n партий в течение n единиц времени будет пропорционален следующей величине:

$$S(\sigma) = c_{\sigma(1)1} + c_{\sigma(2)2} + \cdots + c_{\sigma(n)n}$$

А масса выхода конечного продукта равна M * S.

Оптимизационная задача состоит в том, чтобы найти такую перестановку $\sigma$ чисел от 1 до n (последовательность переработки имеющихся партий сырья), при которой S($\sigma$ ), будет максимальным. Назовем такую перестановку максимальной стратегий.

Математическая модель оптимального расписания (плана) переработки сахарной свеклы идентична нечеткой задаче о назначениях, поэтому эвристические методы решения нечеткой задачи могут быть использованы в качестве эвристических стратегий для плана переработки партий сырья.

## 2.2. Краткое описание программы системы поддержки и принятия решений

Фактически программа будет состоять из 2-х частей: алгоритмов, которые реализуют эвристические стратегии для решения задачи, а также интерфейса, который сделает куда более удобным для пользователя ввод данных, а также вывод результатов для их дальнейшей обработки и анализа.

На вход будут подаваться следующие данные: число партий, число дней дозаривания, минимальная и максимальная сахаристость, минимальный и максимальный коэффициент дозаривания, а также минимальный и максимальный коэффициент деградации.

На выходе пользователь получает график со сравнением стратегий. Помимо этого, пользователь сможет увидеть рекомендуемую стратегию в зависимости от входных данных, а также потери каждой стратегии.

## 2.2. Выбор эвристических стратегий и их реализация на Python

Для исследования можно предложить следующие стратегии:

1) «Жадная» (Greedy) стратегия заключается в использовании «жадного» алгоритма. Перерабатывается та партия сырья из оставшихся, которая в данный момент может дать наибольший выход продукции (сахара), то есть ту партию с номером i, у которой $s_{ij}$ в начале j -го периода наибольшее. Реализация стратегии принимает на вход матрицу, с которой будет работать. В процессе работы алгоритм ищет максимальные элементы в каждом столбце, выбирая их, если они ещё не были выбраны. По завершении работы алгоритм возвращает итоговую сумму, индексы выбранных элементов и сумму на каждом шаге.

2) **«Бережливая» (Thrifty) стратегия** использует противоположный по смыслу жадному – «бережливый» алгоритм: на каждом этапе перерабатывается партия сырья из оставшихся с наименьшей производственной ценностью (партию с номером i, у которой $s_{ij}$ в начале j - го периода – наименьшее). Реализация стратегии принимает на вход матрицу, с которой будет работать. В процессе работы алгоритм ищет мминимальные элементы в каждом столбце, выбирая их, если они ещё не были выбраны. По завершении работы алгоритм возвращает итоговую сумму, индексы выбранных элементов и сумму на каждом шаге.

3) **«Бережливая/жадная» стратегия** состоит в том, что во время первых $\nu - 1$ этапов реализуется «бережливый» алгоритм, а после, начиная с $\nu$ -го этапа, используется «жадный» алгоритм. Число $\nu,\ \nu = \overline{2, \left[\frac{n}{2}\right]}$, известно до начала сезона. Реализация стратегии принимает на вход матрицу, с которой будет работать, а также количество шагов, в течение которых используется бережливый подход. В процессе работы алгоритм в течение некоторого числа шагов, указанных на входе, ищет минимальные элементы в каждом столбце, выбирая их, если они ещё не были выбраны, а затем переходит к жадному подходу. По завершении работы алгоритм возвращает итоговую сумму, индексы выбранных элементов и сумму на каждом шаге.

4) **«Жадная/бережливая» стратегия** использует во время первых $\nu - 1$ этапов «жадный» алгоритм, а после, начиная с $\nu$ -го этапа, используется «бережливый» алгоритм. Реализация стратегии принимает на вход матрицу, с которой будет работать, а также количество шагов, в течение которых используется жадный подход. В процессе работы алгоритм в течение некоторого числа шагов, указанных на входе, ищет макисмальные элементы в каждом столбце, выбирая их, если они ещё не были выбраны, а затем переходит

к бережливому подходу. По завершении работы алгоритм возвращает итоговую сумму, индексы выбранных элементов и сумму на каждом шаге.

5) **Стратегия типа БkЖ (T(k)G)** состоит в том, что во время первых $\nu - 1$ этапов на переработку отправляется партия, которая в данный момент находится на k-ой позиции по сахаристости, считая от наименьшей (1 позиция) до наибольшей, для корректности этого условия необходимо, чтобы $k = \overline{1, n - \nu + 1}$. Далее, начиная с $\nu$ этапа, используется «жадный» алгоритм. Например, «бережливая/жадная» стратегия – это стратегия Б1Ж (T(1)G). Реализация стратегии принимает на вход матрицу, с которой будет работать, количество шагов, в течение которых используется бережливый подход, а также параметр $k$, влияющий на выбор элементов. В процессе работы алгоритм в течение некоторого числа шагов, указанных на входе, ищет минимальные элементы в каждом столбце, выбирая их, если они ещё не были выбраны, а затем максимальные, используя значение $k$. По завершении работы алгоритм возвращает итоговую сумму, индексы выбранных элементов и сумму на каждом шаге.

6) **Стратегия CTG** состоит в перестановке $\omega$ , которая определяет расписание стратегии CTG по описанному правилу (см. метод CTG), здесь $b_i = b_{i1}$, $i = \overline{1, n}$. Реализация стратегии принимает на вход матрицу, с которой будет работать, количество шагов, в течение которых используется бережливый подход, а также параметр $k$, влияющий на выбор элементов. В процессе работы алгоритм использует функцию $gamma$ для выбора индексов на основе значения $k$. По завершении работы алгоритм возвращает итоговую сумму, индексы выбранных элементов и сумму на каждом шаге.

7) **Стратегия Жk (Gk)** является предлагаемой нами стратегией для решения задачи о назначениях. Реализация стратегии принимает на вход матрицу, с которой будет работать, а также параметр $k$, влияющий на выбор элементов. В процессе работы алгоритм сначала проходится по строкам каждого столбца, выбирая элемент в соответствии с параметром $k$, после чего массив выбранных элементов сортируется в порядке убывания, и на каждой итерации выбирается элемент с максимальным значением. Можно сказать, что стратегия является некоторым дополнением Жадной стратегии, а именно использует параметр $k$ для выбора элементов, что в некоторых случаях может дать наиболее подходящий результат. По завершении работы алгоритм возвращает итоговую сумму, индексы выбранных элементов и сумму на каждом шаге.

## 2.4. Проверка работы программы СППР и проведение малого эксперимента

Интерфейс программы СППР при запуске имеет следующий вид (рис 2.1):

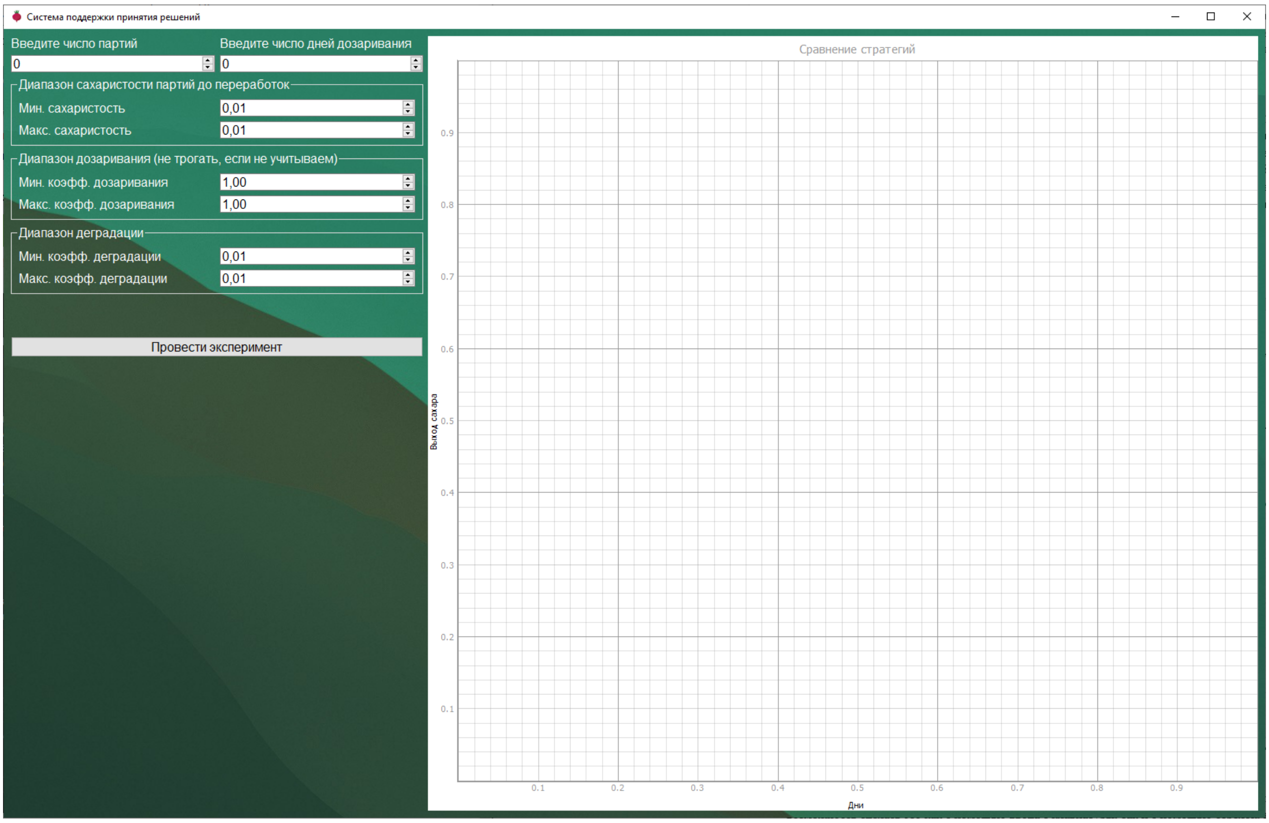


Рисунок 2.1. Интерфейс СППР при запуске

В левой верхней части пользователь задает вводные данные. Интерфейс предоставляет возможность сделать это как с помощью ввода с клавиатуры, так и с помощью стрелок, которые находятся внутри строчки. Помимо этого, интерфейс не позволит пользователю ввести, например, буквенные символы вместо цифр, а сама программа проверит введенные данные на корректность и, в случае ошибки, выведет об этом сообщение. Для демонстрации остального функционала программы проведем на ней малый эксперимент, задав следующие вводные:

Число партий: 15

Число дней дозаривания: 7

Минимальная сахаристость: 0,12

Максимальная сахаристость: 0,22

Минимальный коэффициент дозаривания: 1

Максимальный коэффициент дозаривания: 1,15

Минимальный коэффициент деградации: 0,85

Максимальный коэффициент деградации: 1

Теперь проведем эксперимент, нажав соответствующую кнопку(рис 2.2):

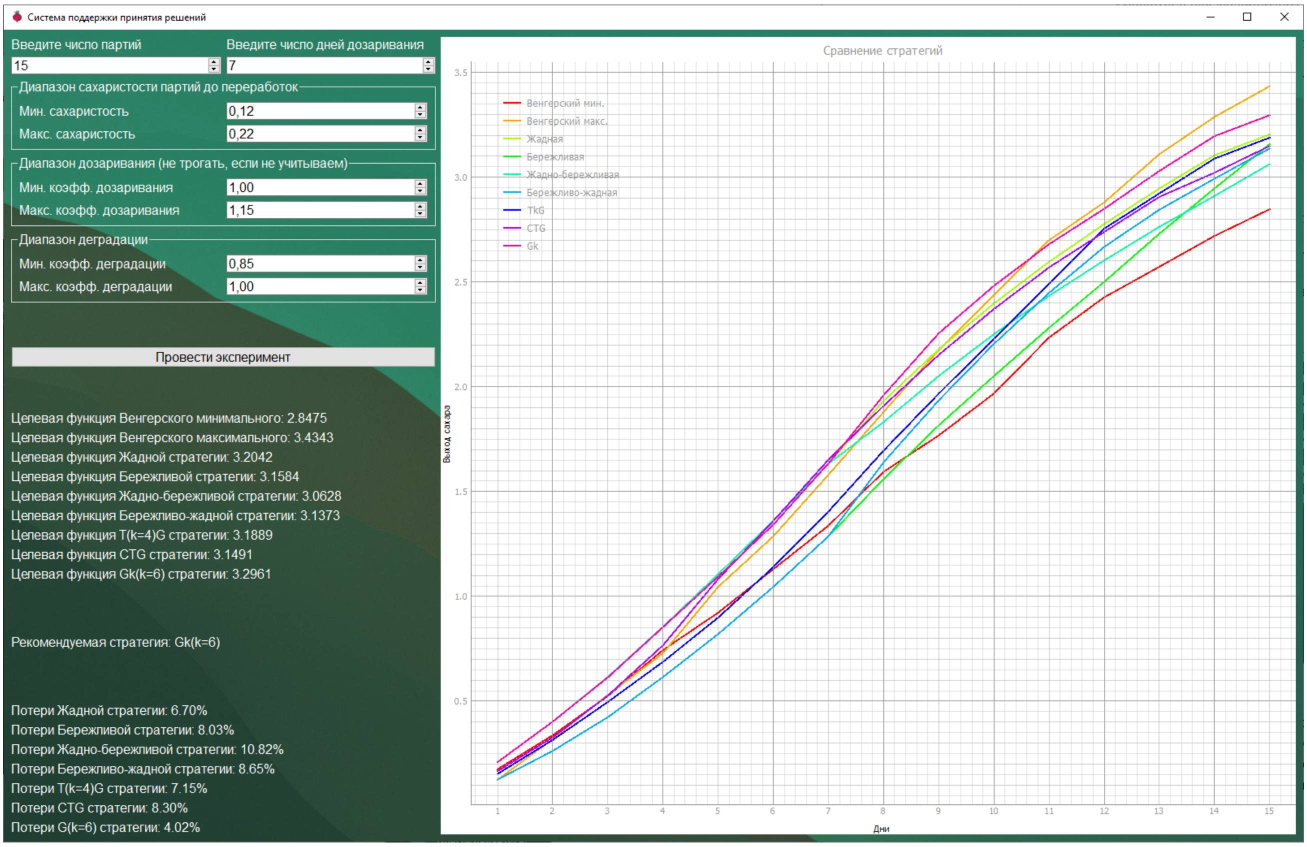

Рисунок 2.2. Интерфейс СППР после проведения эксперимента

На выходе мы получаем графики зависимости выхода сахара от дней для каждой стратегии для отражения динамики. Кроме того, программа выводит относительные потери каждой стратегии и предлагает лучшую в зависимости от исходных данных. В случае малого эксперимента программа советует использовать Жk(k=6) стратегию.

# Заключение

В ходе данной лабораторной работы была изучена задача о назначениях, а также некоторые её разновидности, такие как нечеткая задача о назначениях и нечеткая задача о назначениях со строчно-изоантитонной матрицей. Кроме того, были рассмотрены некоторые эвристические стратегии решения подобных задач, а именно: жадная, бережливая, жадно-бережливая, бережливо-жадная, CTG и TkG стратегии.

В практической части лабораторной работы была поставлена задача написать программу системы поддержки принятия решений для анализа различных стратегий, которые могут использованы в плане переработки сахарной свеклы, максимизирующего выход сахара.  Изученные раннее стратегии были реализованы на Python. В ходе реализации мы узнали новый функционал данного языка программирования. Помимо этого, для программы был написан интерфейс, который сделает работу с программой куда более комфортной для пользователя. В ходе реализации интерфейса нами были изучены некоторые новые конструкции и возможности Python. Используя написанную нами программу, пользователь сможет увидеть потери , а также посмотреть результат применения каждой стратегии в динамике, взглянув на графики.

## Список литературы


1. Баландин, Д.В., Кузенков, О.А., Малышев, Д.С., Приставченко, О.В., Эгамов, А.И. Эвристические методы для решения нечеткой задачи о назначениях. Учебное пособие. – Нижний Новгород: Нижегородский госуниверситет, 2024 г.
2. Лутц, Марк Изучаем Python / пер. с англ. Ю.Н.Артеменко. 5-е изд. – М.: Диалектика, 2019 г.
3. Хайнеман, Джордж Алгоритмы. С примерами на Python / пер. с англ. Г. Курячего. – СПб.: Питер, 2023 г.
4. Метиз, Эрик Изучаем Python: программирование игр, визуализация данных, веб-приложения. 3-е изд. / пер. с англ. Е. Матвеева. – СПб.: Питер, 2020 г.