

Национальный исследовательский Нижегородский
государственный университет имени Н. И. Лобачевского

(Институт информационных технологий, математики и механики)

Отчёт по лабораторной работе

«Оптимизация последовательности переработки партий сахарной свёклы»

Выполнили:
Студенты группы 3823Б1ФИ1
Гусев Д.А.
Дорофеев И.Д.
Федосеев С.Н.
Никитин А.А.
Савва Д.А.

Научный руководитель:
Эгамов А.И.

2025 г.

Содержание

| | |
|---|----------|
| Постановка задачи | 2 |
| Математическая модель деградации свёклы | 2 |
| Процесс дозаривания свёклы | 2 |
| Распределение коэффициентов | 2 |
| Матрица состояний и потери | 2 |
| Задача оптимизации | 3 |
| Особенности задачи | 3 |
| Как работает программа | 3 |
| Архитектура системы | 3 |
| Процесс работы программы | 4 |
| Генерация данных | 4 |
| Алгоритмы оптимизации | 4 |
| Анализ результатов | 5 |
| Интерфейс и валидация | 5 |
| Заключение | 5 |
| Список литературы | 6 |

Постановка задачи

Математическая модель деградации свёклы

Рассматривается задача оптимизации последовательности переработки n партий сахарной свёклы на сахарном заводе. За единицу времени завод перерабатывает партию сырья массой M одного сорта.

Пусть $i = \overline{1, n}$ — номер партии, $j = \overline{1, n}$ — этап переработки. Обозначим через c_{ij} количество полезного ингредиента (сахара) в начале этапа j для партии i .

Параметры c_{ij} определяются через начальную долю полезного ингредиента $c_{i1} = a_i \in [a_{\min}, a_{\max}]$ и коэффициенты деградации $b_{i,j-1} = \frac{c_{ij}}{c_{i,j-1}}$ для $j = 2, \dots, n$. Соответствующие формулы имеют вид:

$$c_{i1} = a_i, \tag{1}$$

$$c_{ij} = a_i b_{i1} \dots b_{i,j-1}, \quad i = 1, \dots, n, \quad j = 2, \dots, n. \tag{2}$$

Процесс дозаривания свёклы

Опционально учитывается процесс дозаривания (дозревания) свёклы:

- Число этапов дозаривания ограничено и равно v , где $2 \leq v \leq \lfloor \frac{n}{2} \rfloor$.
- На этапах дозаривания параметры $b_{ij} \in (1, \beta_{\max}]$ для $j = \overline{1, v-1}$.
- На этапах увядания параметры $b_{ij} \in [\beta_1, \beta_2] \subset (0, 1)$ для $j = \overline{v, n-1}$.
- Рекомендуется брать $\beta_{\max} = \frac{n-1}{n-2}$.

Распределение коэффициентов

Коэффициенты деградации b_{ij} могут распределяться двумя способами:

1. **Равномерное распределение:** b_{ij} распределены равномерно на отрезке $[\beta_1, \beta_2]$ (для увядания) или $(1, \beta_{\max}]$ (для дозаривания).
2. **Концентрированное распределение:** Для каждой партии i существует константа $\delta_i \leq \lfloor \frac{\beta_2 - \beta_1}{4} \rfloor$, 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 этом подотрезке параметры распределены равномерно.

Матрица состояний и потери

Имеется матрица состояний $\tilde{S} = C - \tilde{L}$, где:

- C — матрица коэффициентов содержания сахара для каждой партии (строки) на каждом этапе (столбцы).
- \tilde{L} — матрица с неотрицательными элементами потерь сахара.
- \tilde{S} — матрица коэффициентов содержания сахара после учёта потерь.

Элемент l_{ij} матрицы потерь (в %) вычисляется по формуле:

$$l_{ij} = 1.1 + 0.1541 \times (K_i + Na_i) + 0.2159 \times N_i + 0.9989 \times I_{ij} + 0.1967, \tag{3}$$

где:

- $K_i \in [4.8; 7.05]$ — содержание калия;
- $Na_i \in [0.21; 0.82]$ — содержание натрия;
- $N_i \in [1.58; 2.8]$ — содержание азота;
- $I_{ij} = I_{i0} \times (1.029)^{7j-7}$, где $I_{i0} \in [0.62; 0.64]$ — начальный индекс, j — номер этапа (нумерация с 1), 7 — количество дней в этапе.

Потери учитываются при расчёте итоговой матрицы: $\tilde{S} = C - \frac{L}{100}$.

Задача оптимизации

Последовательность переработки сырья описывается перестановкой σ натуральных чисел от 1 до n :

$$\sigma = \begin{pmatrix} 1 & 2 & \dots & n-1 & n \\ \sigma(1) & \sigma(2) & \dots & \sigma(n-1) & \sigma(n) \end{pmatrix}$$

Общий выход сахара рассчитывается по формуле:

$$S(\sigma) = \tilde{s}_{\sigma(1),1} + \tilde{s}_{\sigma(2),2} + \dots + \tilde{s}_{\sigma(n),n} \quad (4)$$

Масса конечного продукта вычисляется как:

$$S = S(\sigma) \times M \times d, \quad (5)$$

где $d = 7$ — число дней в этапе.

Цель оптимизации: найти такую перестановку σ чисел от 1 до n (последовательность переработки имеющихся партий сырья), при которой $S(\sigma)$ будет максимальным.

Особенности задачи

В начале этапа j известны только первые j столбцов матрицы, причём только для партий, ещё не переработанных. Это делает задачу нечёткой задачей о назначениях, что исключает применение точных алгоритмов (например, венгерского алгоритма) в реальном времени. Однако в рамках компьютерного эксперимента возможно сравнение различных эвристических стратегий с оптимальным решением, полученным венгерским алгоритмом.

Как работает программа

Разработанная система является инструментом для проведения вычислительных экспериментов по оптимизации процесса переработки сахарной свёклы. Она реализована в виде desktop-приложения с использованием современных веб-технологий.

Архитектура системы

Система построена по клиент-серверной архитектуре и включает в себя следующие компоненты:

1. **Клиент (Frontend):** Графический интерфейс пользователя, созданный с помощью фреймворка **Electron**. Это позволяет использовать для разработки интерфейса стандартные веб-технологии (HTML, CSS, JavaScript). Задача клиента — предоставить пользователю удобный инструмент для ввода параметров симуляции, отправить их на сервер и наглядно отобразить полученные результаты.

2. **Сервер (Backend):** Ядро системы, реализованное на языке **Python** с использованием микро-фреймворка **Flask**. Сервер предоставляет REST API для взаимодействия с клиентом.
3. **Модули вычислений:** Набор скриптов на Python, использующих библиотеки **NumPy** для эффективных матричных операций и **SciPy**.

Процесс взаимодействия следующий: пользователь вводит параметры в интерфейсе Electron, клиент формирует из них JSON-объект и отправляет POST-запрос на сервер Flask. Сервер обрабатывает запрос, запускает модули вычислений, получает результаты и возвращает их клиенту в виде JSON-ответа. Клиент парсит ответ и выводит данные в виде таблиц и графиков.

Процесс работы программы

Генерация данных

При запуске симуляции система последовательно генерирует математическую модель процесса:

1. **Генерация партий свёклы:** Для каждой из n партий создается вектор начальных условий: сахаристость a_i и химические параметры (K_i, Na_i, N_i, I_{i0}) , взятые случайным образом из заданных диапазонов.
2. **Генерация коэффициентов деградации:** Формируется матрица переходов B , описывающая изменение сахаристости со временем, с учётом опции дозаривания и выбранного типа распределения (равномерное или концентрированное).
3. **Расчёт матриц состояний и потерь:** На основе сгенерированных данных вычисляются полная матрица сахаристости C и матрица технологических потерь L .
4. **Формирование итоговой матрицы \tilde{S} :** Финальная матрица, используемая для оптимизации, получается путем вычета нормированных потерь из матрицы сахаристости: $\tilde{S} = C - L/100$.

Алгоритмы оптимизации

Система реализует и сравнивает несколько стратегий оптимизации, от простых эвристик до точного метода.

1. **Жадная стратегия (Greedy):** Наиболее очевидный подход. На каждом этапе j выбирается та доступная партия, которая имеет максимальную сахаристость \tilde{s}_{ij} именно в этот момент. Стратегия является локально-оптимальной, но часто приводит к субоптимальному итоговому результату, так как "сжигает" лучшие партии в самом начале.
2. **Бережливая стратегия (Thrifty):** Контринтуитивный, но в некоторых случаях эффективный метод. На каждом этапе выбирается партия с *минимальной* текущей сахаристостью. Идея состоит в том, чтобы сохранить партии с высоким потенциалом на более поздние этапы, где их вклад в общую сумму будет более весомым по сравнению с остальными.
3. **Гибридные стратегии (Thrifty/Greedy, Greedy/Thrifty):** Комбинированные подходы, пытающиеся взять лучшее от двух базовых стратегий. Например, стратегия Thrifty/Greedy первые $\lfloor n/2 \rfloor$ этапов работает как бережливая, сохраняя лучшие партии, а затем переключается на жадную, чтобы извлечь из них максимальную выгоду.

4. **Сложные эвристики ($T(k)G, Gk$):** Вариации вышеописанных методов, предназначенные для исследования более тонких закономерностей в задаче.
5. **Венгерский алгоритм:** Точный алгоритм, решающий задачу о назначениях за полиномиальное время ($O(n^3)$). В контексте данной работы он выступает в роли "оракула": зная всю матрицу \tilde{S} заранее, он находит глобально оптимальную последовательность σ^* с максимальным выходом сахара S^* . Этот результат недостижим в реальности, но служит абсолютно точным эталоном для оценки качества работы эвристических стратегий.

Анализ результатов

Ключевой задачей после проведения симуляции является анализ полученных данных. Для каждой стратегии вычисляется итоговый выход сахара $S(\sigma)$. Главным показателем эффективности эвристики является её относительное отклонение от эталонного значения, полученного Венгерским алгоритмом:

$$\text{Efficiency} = \frac{S(\sigma_{\text{heuristic}})}{S(\sigma_{\text{optimal}})} \times 100\%$$

Анализ позволяет определить, какая из эвристических стратегий является наиболее робастной и эффективной для различных наборов входных параметров (например, для разных типов распределения коэффициентов деградации).

Интерфейс и валидация

Графический интерфейс позволяет пользователю гибко настраивать параметры эксперимента и немедленно видеть результат. В систему встроена валидация входных данных для предотвращения некорректного ввода (например, проверка того, что $a_{\min} \leq a_{\max}$).

Заключение

В рамках лабораторной работы была успешно спроектирована и реализована система поддержки принятия решений для задачи оптимизации последовательности переработки сырья. Разработанный программный комплекс позволяет моделировать сложный технологический процесс с учётом множества параметров и проводить вычислительные эксперименты для сравнения эффективности различных управляющих стратегий.

Сравнение эвристических методов с точным решением, полученным с помощью Венгерского алгоритма, показало, что простые стратегии (жадная, бережливая) могут быть эффективны лишь в узком диапазоне условий, в то время как гибридные подходы демонстрируют более стабильные и близкие к оптимальным результаты.

Перспективы развития проекта:

- **Калибровка модели:** Использование реальных производственных данных для уточнения формулы потерь и коэффициентов деградации.
- **Стохастическое моделирование:** Учёт случайного фактора в параметрах модели и применение методов стохастического программирования или обучения с подкреплением для поиска оптимальной стратегии в условиях неопределенности.
- **Расширение функционала:** Добавление в систему других технологических и экономических факторов, таких как стоимость хранения, логистические ограничения и т.д.

Список литературы

1. Кнут Д. Э. Искусство программирования. Том 3. Сортировка и поиск / Д. Э. Кнут. — 2-е изд. — М.: Вильямс, 2000. — 832 с.
2. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ / Т. Кормен [и др.]. — 3-е изд. — М.: Вильямс, 2013. — 1328 с.
3. Kuhn H. W. The Hungarian method for the assignment problem / H. W. Kuhn // Naval Research Logistics Quarterly. — 1955. — Vol. 2, No. 1-2. — P. 83-97.
4. Munkres J. Algorithms for the Assignment and Transportation Problems / J. Munkres // Journal of the Society for Industrial and Applied Mathematics. — 1957. — Vol. 5, No. 1. — P. 32-38.
5. SciPy Community. `scipy.optimize.linear_sum_assignment` — SciPy v1.11.0 Manual [Электронный ресурс]. — Режим доступа: https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.linear_sum_assignment.html (дата обращения: 2024).
6. Flask Development Team. Flask — Web Development, One Drop at a Time [Электронный ресурс]. — Режим доступа: <https://flask.palletsprojects.com/> (дата обращения: 2024).
7. Electron Contributors. Electron — Build cross-platform desktop apps with JavaScript, HTML, and CSS [Электронный ресурс]. — Режим доступа: <https://www.electronjs.org/> (дата обращения: 2024).
8. NumPy Developers. NumPy — The fundamental package for scientific computing with Python [Электронный ресурс]. — Режим доступа: <https://numpy.org/> (дата обращения: 2024).