**Текст доклада**

**Вступление (1 слайд)**

Здравствуйте, уважаемый председатель и члены государственной экзаменационной комиссии.

Представляю доклад по выпускной квалификационной работе бакалавра на тему “Разработка веб-приложения для решения задач смешано-целочисленного линейного программирования“

**Цель и задачи (2 слайд)**

Цель работы - разработка веб-приложения, которое упростит процесс решения задач MILP для конечных пользователей.

Поставленные задачи:

* создать интуитивно понятный интерфейс для ввода линейных целевых функций и ограничений
* обеспечить возможность загрузки параметров задач из файла
* исполнять и сохранять задачи на сервере
* просматривать итоговое решение и анализ чувствительности
* определить алгоритмы обмена данными между клиентом и сервером

**Математическая постановка задачи (3 слайд)**

Задачи смешано-целочисленного линейного программирования представляют собой класс оптимизационных задач, где целевая функция и ограничения выражены линейными зависимостями, а переменные могут быть как непрерывными, так и дискретными. Формальная модель MILP-задачи имеет следующий общий вид:

1. Целевая линейная функция, которую необходимо оптимизировать, то есть либо минимизировать, либо максимизировать. Она представляет собой сумму произведений i-го коэффициента на i-ую переменную
2. Оптимизация функции происходит при выполнении m условий. Каждое условие - это линейное ограничение, в левой части которого стоит сумма произведений j-го коэффициента на j-ую переменную, справа - число ограничивающее эту сумму, между ними знак сравнения
3. Часть переменных в таких задачах принадлежит множеству целых чисел, другая часть - множеству действительных чисел

**Структура данных (4 слайд)**

Структура данных приложения представляет собой одну таблицу tasks, в которой находится 9 полей:

1. id - id записи в таблице
2. task\_id - id задачи
3. conditions - условие задачи в формате json, имеет вложенную структуру, изображенную на слайде
4. solution - решение задачи, тоже json формат с вложенной структурой
5. solver - решатель, который был выбран пользователем
6. conditions\_excel - условие задачи в виде excel файла, закодированное в строку base64
7. upload\_time - время загрузки
8. solve\_time - время решения
9. canceled - булевый флаг, если пользователь отменил решение

**Архитектура проекта (5 слайд)**

Архитектурно проект представляет из себя:

1. Фронтенд - визуальная часть проекта, монолит, в котором используется vite (инструмент сборки веб-приложения), React (веб фреймворк), TypeScript (типизированный язык программирования), rtk query (стейт менеджер)
2. Серверная часть проекта - монолит, написанный на Python.

При общении клиента с бэкендом REST API используется в случае загрузки задачи или отмены решения, а статус решения и само решение клиент получает по подключенному SSE соединению, в котором бэкенд в одностороннем порядке отправляет сообщения.

В качестве контроллера выступает Flask.

Если на сервер задача поступила в формате Excel, то она проходит через дополнительный модуль - Pandas, где она парсится и приводится к объектному виду, привычному для Python.

После этого задача поступает в решатель Pyomo, в котором происходит решение задачи. После чего формируется ответ и отправляется на клиент.

На каждом из этапов информация сохраняется в базу данных, посредником между приложением и ней выступает ORM SqlAlchemy. В качестве базы данных выбрана PostgreSQL

**Ввод задачи (6 слайд)**

Пользователь, заходя на сайт, в левой его части видит форму, в которой он может ввести данные задачи MILP:

1. Выбрать количество переменных
2. Указать на целочисленность переменных
3. Ввести параметры целевой функции и линейных ограничений
4. Выбрать один из 3 представленных решателей

**Загрузка файла (7 слайд)**

Как я говорил ранее, пользователю доступен еще один формат загрузки задачи - Excel файл. При нажатии на соответствующую кнопку, открывается модальное окно, в котором указаны требования к составлению такого файла.

Для удобства пользователю для скачивания доступны шаблон файла и пример заполненной задачи.

**Процесс решения (8 слайд)**

В случае успешной загрузки задачи на сервер, тот отправляет на клиент ответ с id задачи, по которому устанавливается SSE соединение. Такой подход необходим, потому что задачи могут быть большими и на их решение могут уходить часы и даже дни. Поэтому обычный формат запрос-ответ не подойдет, так как запрос в таком случае, после небольшого ожидания ответит с ошибкой.

Пока по этому соединению сервер отправляет сообщения о том, что задача находится в процессе решения, пользователю показывается состояние загрузки.

Так же, если решение задачи затянулось или были введены неверные параметры, юзеру предоставлена возможность отменить решение задачи, нажав на соответствующую кнопку.

**Результат решения (9 слайд)**

Если пользователь дождался успешного решения задачи. То в правой части экрана появляется следующая информация:

1. Условие задачи. Есть возможность скачать Excel файлом загруженное условие
2. Решатель, который был выбран для решения этой задачи
3. Сообщение от решателя
4. Информация о продолжительности решения
5. Значение целевой функции
6. Значения переменных
7. А также информация об анализе чувствительности. В последнем столбце содержится информация о том, насколько изменится значение целевой функции, при изменении правого столбца ограничения на единицу

Для удобства, можно настроить количество знаков после запятой для не целых значений.

**Результат (10 слайд)**

Разработано веб-приложение, которое позволяет пользователям с удобным и интуитивно понятным интерфейсом загружать условия MILP-задачи в виде Excel‑файлов или вводить их вручную, выбирать нужный решатель и получать оптимальное решение задачи с анализом чувствительности. Вся информация о задачах сохраняется в базе данных. Поддержана возможность решения тяжелых задач.

Таким образом все задачи решены и цель разработать веб-приложение, которое упростит процесс решения задач MILP для конечных пользователей достигнута.

Доклад окончил. Спасибо за внимание. (переключить на слайд с qr-кодом)