Краткое описание решения.

Основные работы на которые я опирался указаны в репозитории, а также в прошлых отчетах.

1 Построение модели ЦОД

В данной секции описан процесс построения модели ЦОД.

1.1 Общее описание модели

Модель ЦОД создавалась с использованием OpenStudio и EnergyPlus. Сама модель описывается в файле формата `.idf`, с помощью специального языка разметки, этот файл подается в EnergyPlus для запуска процесса моделирования. Также использовалось приложение OpenStudio Application, которое позволяет сделать некоторую часть

работы с помощью приложения с графическим интерфейсом. Некоторые рисунки ниже это скриншоты из этого приложения.

ЦОД состоит из помещения 31x4x18 метров. Используемые материалы и конструкции подбирались под климатическую зону 6A (ASHRAE CLIMATE ZONE). Помещение разделено на 11 температурных зон. Первая зона – горячая, далее типы зон чередуются. Для горячих зон установлена нагрузка 1000 Вт/м^2 (исключение – первая и последняя горячие зоны, для них установлено значение в 500 Вт/м^2). Для холодных зон нагрузка отсутствует (за исключением минимальной осветительной нагрузки). Между зонами стоит воздушная стена (air boundary construction), через нее зоны могут обмениваться воздухом (и соответственно температурой). Ширина холодных коридоров – 2 метра. Крайних горячих – 3 метра, промежуточных – 4 метра.

Под основным помещением установлен фальшпол. Внутри ЦОД стоит система CRAH, которая ориентируется на температуру внутри холодных зон. Холодный воздух подается внутрь фальшпола под управлением системы CRAH, а затем из фальшпола холодный воздух попадает в холодные коридоры. Модель здания представлена на Рис. 1.

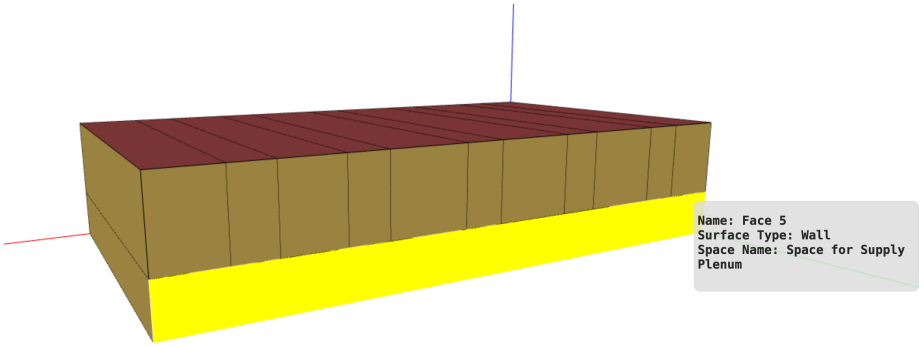

Рис. 1: Модель здания ЦОД.

Для каждой поверхности указан ее тип, а для каждого типа установлена конструкция (состоящая из слоев материалов), из которой эта поверхность состоит (Рис. 2)

Визуализация очередности холодных коридоров внутри здания представлена на Рис. 3. Между холодными коридорами расположены горячие.

Компонентная визуализация системы CRAH представлена на Рис 4. Воздух, попадающий в систему (в том числе наружный), проходит через змеевик водяного охлаждения (CRAH Water Cooling Coil), который охлаждается в Chilled Water Loop (Рис. 5) (внутри него используется Condenser Water Loop (Рис. 6)). После происходит контроль влажности воздуха, и воздух через CRAH Fan, попадает в фальшпол, в VAV Terminal, а затем в холодные коридоры.

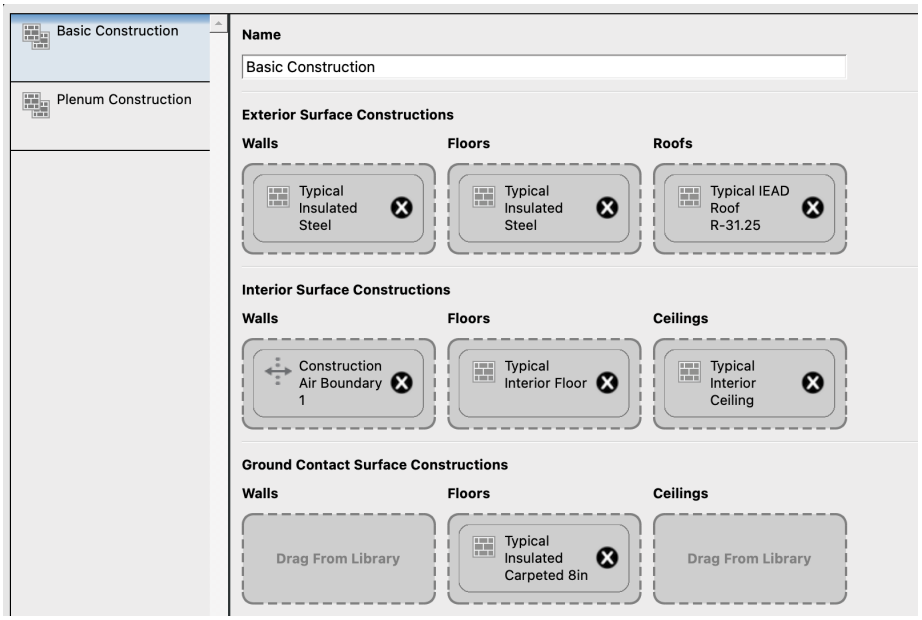

Рис. 2: Конструкции основного помещения модели ЦОД.

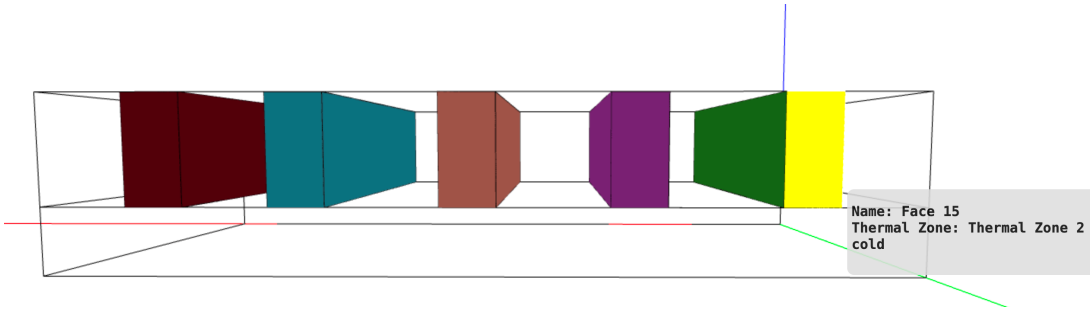

Рис. 3: Холодные коридоры в модели ЦОД.

Для каждого компонента каждой системы подобраны параметры.

В модель добавлены данные о погоде в Москве из источников ASHRAE (ссылка), информация о уровне CO_2, информация о осве-

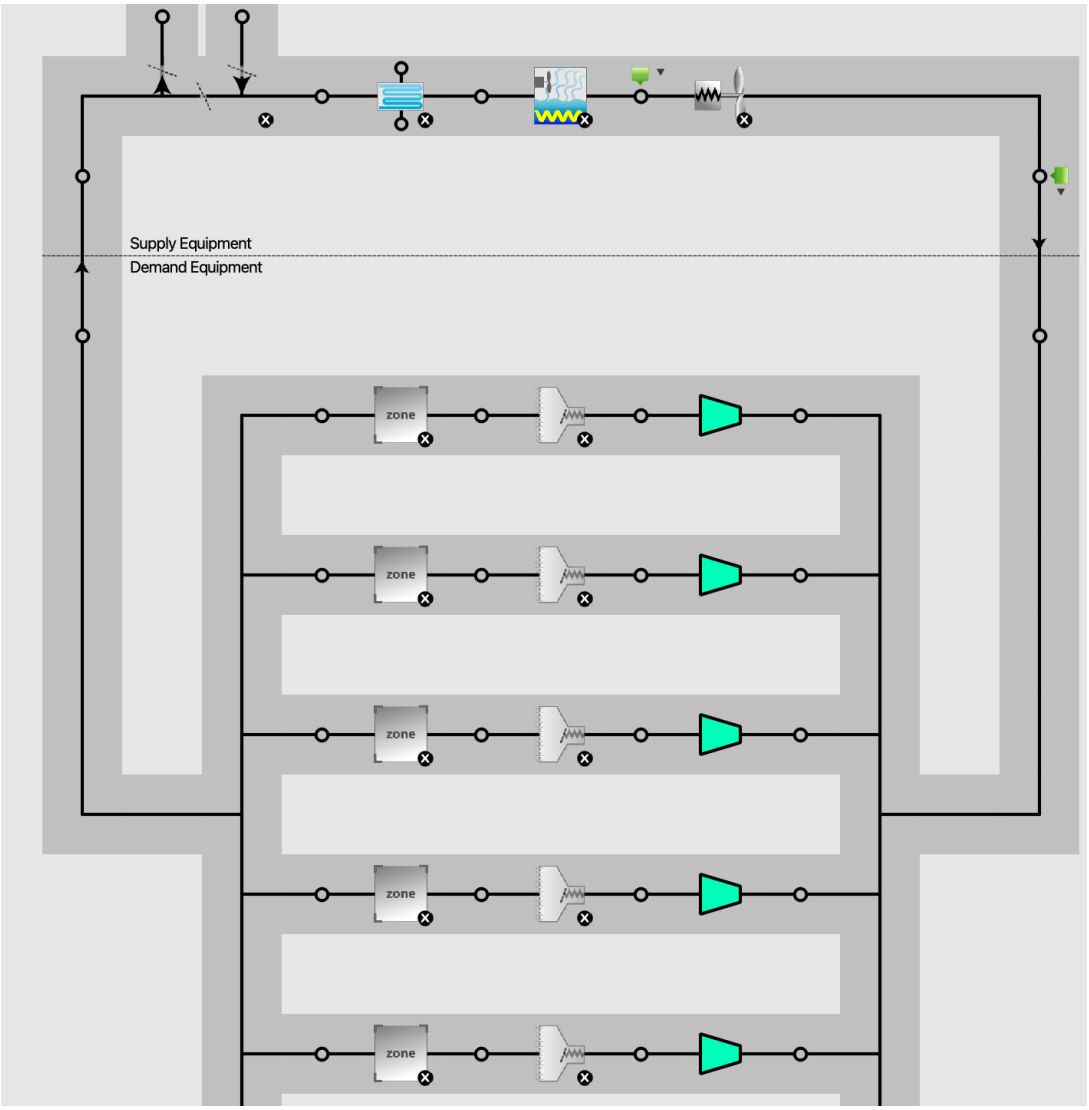

Рис. 4: Компоненты системы CRAH.

тительной нагрузке и другое.

Далее модель дорабатывалсь уже в самом файле с кодом описания модели.

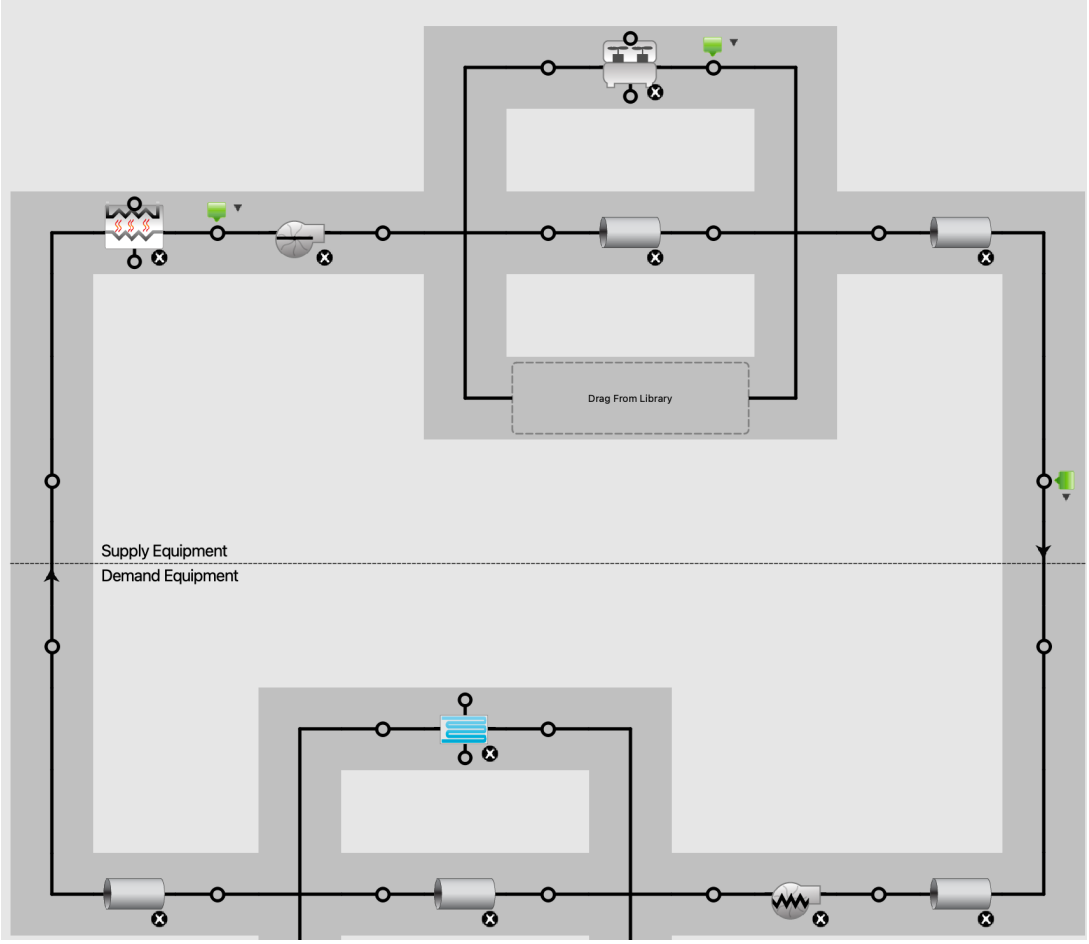

Рис. 5: Компоненты системы Chilled Water Loop.

Это очень общее описание модели, каждый компонент настраивался отдельно, полное описание модели на языке разметики, который читает EnergyPlus, представлен здесь.

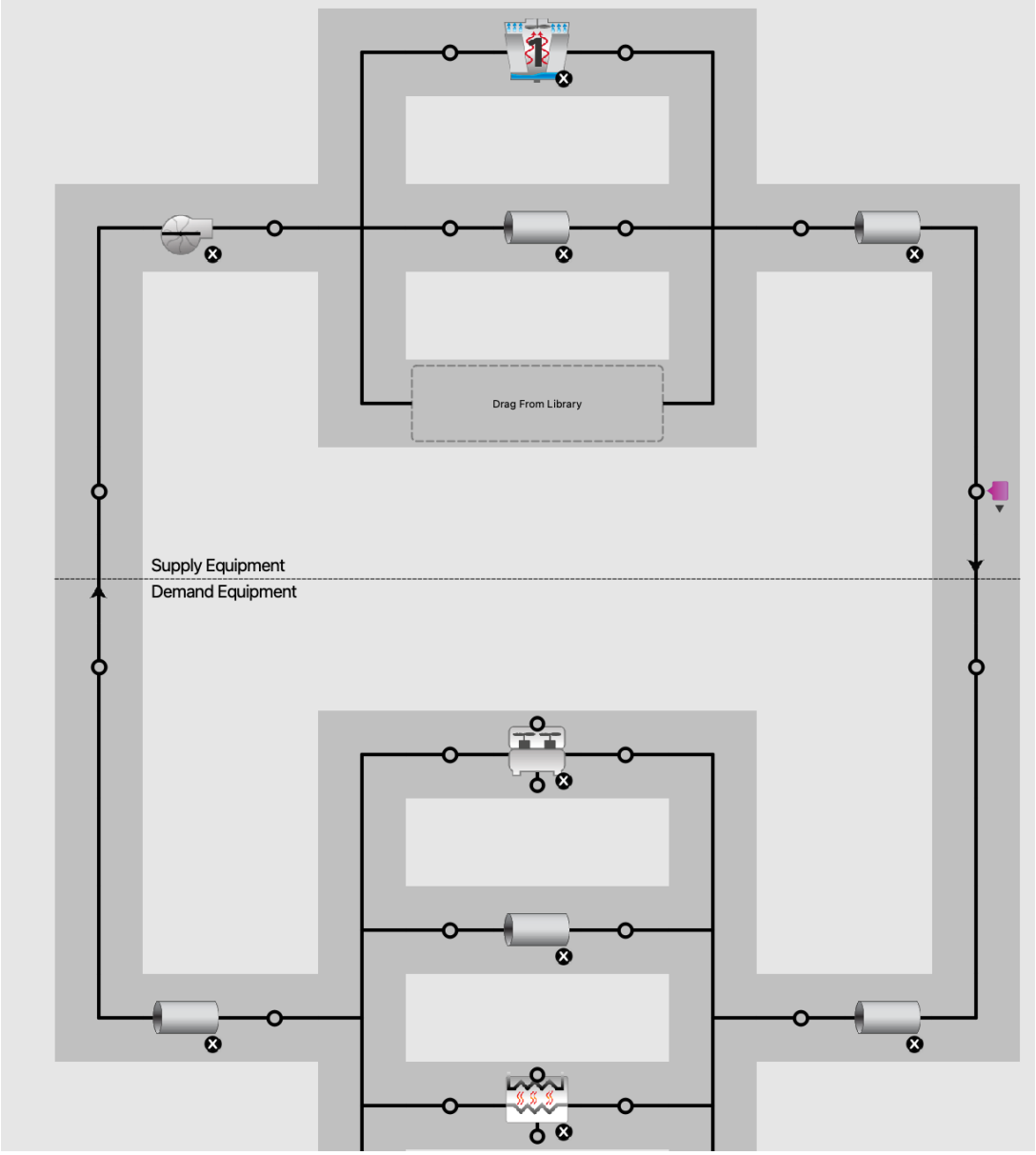

Рис. 6: Компоненты системы Condenser Water Loop.

1.2 Взаимодействие с Python и RL-агентом

Необходимо было добавить возможность влиять на симуляцию в процессе исполнения (чтобы RL-агент мог управлять уставками). Для этого применялись ExternalInterface на расписаниях (Schedule) управляемых объектов, управление ими проводилось посредством Ptolemy2 и Building Controls Virtual Test Bed (bctvb). Передача и получение информации от процесса EnergyPlus велось поверх сокетов.

2 Получение RL-агента

2.1 Среда для обучения

Создавалась среда ЦОД, в которой RL-агент может делать шаги, получать награду и т.п. Среда напрямую давала агенту возможность взаимодействовала с моделю ЦОД, описанной выше, запускала ее симуляцию, совершала действия агента, получала информацию на основе которой строила награждение. Среда наследовала базовый класс среды от gymnasium. В коде описан как и общий абстрактный класс для всех сред ЦОД, так и конкретный для среды, которая взаимодействует с EnergyPlus. Более полно можно посмотреть в исходном

коде, в нем описаны докстринги и типизация каждого метода.

2.2 Обучение агента

Проводилось много экспериментов, обучались агенты с помощью Ray и stable baselines 3 (SB3). Исследовались модели PPO, A2C, SAC. Сейчас как модель для теста инференса стоит PPO от SB3, которая обучалась с использованием transfer learning, сначала модель обучалась в симуляции на 1 день, потом на 3 и тд, также в процессе обучения изменялся learning rate. Большая проблема заключалась в том, что модели обучается достаточно долго, и большое время занимает один шаг агента. Так, SAC обучался более устойчиво, но одна эпоха длилась в 2 раза больше, чем у PPO, поэтому модель SAC не очень подходила для обучения бейзлайн модели на одном процессоре. Из-за этого таже сокращена размерность observation space агента, он видит не всю информацию о зонах, а только информацию о температурах в крайних и центральном коридорах, влажность в центральной зоне, наружную температуру воздуха, и предыдущие уставки охлаждения и температуры воздуха AHU (эксперименты показали, что если агенту поступает информация о предыдущих уставках, то он обучается лучше). Также в процессе обучения стало ясно, что агент лучше

обучается, если он принимает от среды и отдает ей нормализованные числа.

2.3 Сравнение агента и костантных моделей

Сравнение данных симуляции при контроле уставок RL-агентом оказалось эффективнее, чем при контроле константными уставками. Наглядно это можно посмотреть в ноутбуке.

2.4 HTTP-API

С использованием Flask построена HTTP-API для RL-агента. Запускается сервер, который отлавливает json'ы приходящие на POST запрос в в `/predict`, из json берется список наблюдений (observations), который подается в модель. Модель возвращает набор действий; действия добавляется в json, который возвращается клиенту.

3 Логирование и дэшборды

Добавлен код для создания и заполнения БД и таблицы ClickHouse. Добавлен архив с дэшбордом Superset.

Также добавлена возжность сделать симуляцию инференса: запустить HTTP-API, запустить среду ЦОД, каждый шаг которой отправляется с использованием API, создать таблицу ClickHouse и каждый шаг добавлять туда информацию от среды.

4 Описание структуры проекта

Ниже описаны самые важные компоненты структуры проекта.

`main.py` управляет запуском главных функций.

`Makefile` вызывает `main.py` с нужными аргументами в более user-friendly виде, а также имеет различные удобные правила.

`config.py` позволяет указать путь до EnergyPlus и настроить ожидаемый процесс симуляции в коде (нужно, чтобы он соответствовал параметрам в `.idf` файле).

`data/` хранит parquet файлы, полученные в результате тестирования агента и константных моделей.

`models/` хранит предобученные модели.

`notebooks/` хранит ноутбук с сравнением агента и константных моделей.

`external/DC_dashboard` – дэшборд для Superset.

`src/` весь основной код.

`src/eplus_controll/` весь необходимый функционал для работы с EnergyPlus.

`src/dc_env/buildings/MultiZone_DC.idf` хранит описание модели ЦОД

`src/dc_env/data_center_env.py` хранит класс среды для ЦОД.

`src/models/` весь функционал для обучения и теста модели (внутри разделение на константные модели, модели из Ray и модели из SB3).

`src/inference/model_api.py` описание HTTP-API агента.

`src/inference/DBApi.py` класс дя взаимодействия с ClickHouse.

`src/inference/inference_agent.py` для симуляции инференса RL-агента.