**Общее описание**

Environment.py - модуль, реализующий среду симуляции для системы LTE. Он обеспечивает взаимодействие между пользовательскими устройствами, алгоритмами планирования ресурсов и ресурсной сеткой LTE.

**Класс SimulationEnvironment**

**Инициализация**

**def** \_\_init\_\_(self, bandwidth\_mhz: float = 10, num\_frames: int = 10, bs\_position: tuple = (500, 500))

* bandwidth\_mhz - полоса частот в МГц (1.4, 3, 5, 10, 15, 20)
* num\_frames - количество кадров для симуляции
* bs\_position - координаты базовой станции (x, y) в метрах

**Основные методы**

**add\_user**

Добавляет пользовательское устройство в симуляцию.

**run\_simulation**

Запускает симуляцию на заданную длительность с указанным шагом времени.

**visualize\_resource\_grid**

Визуализирует ресурсную сетку LTE.

**visualize\_throughput**

Визуализирует пропускную способность пользователей.

**visualize\_resource\_allocation**

Визуализирует распределение ресурсных блоков по времени.

**print\_statistics**

Выводит статистику симуляции.

**Функция test\_round\_robin**

Создает тестовую симуляцию с алгоритмом планирования Round Robin.

Возможности симуляции и настраиваемые параметры

**Параметры симуляции**

* **Полоса частот (bandwidth\_mhz)**: Влияет на количество доступных ресурсных блоков
* **Количество кадров (num\_frames)**: Определяет размер ресурсной сетки
* **Длительность симуляции (duration\_ms)**: Время работы симуляции
* **Шаг времени (time\_step\_ms)**: Интервал между обновлениями состояния

**Параметры пользователей**

* **Количество пользователей**: Можно создать любое число пользователей
* **Положение пользователей**: Координаты x и y
* **Размер буфера**: Объем данных, доступных для передачи
* **Класс пользователя**: Влияет на модель движения (пешеход, автомобиль и т.д.)

**Пример изменения количества пользователей**

python

**def** test\_custom\_simulation():

*# Создаем среду симуляции*

env = SimulationEnvironment(bandwidth\_mhz=20, num\_frames=2)

*# Добавляем произвольное количество пользователей*

num\_users = 10 *# Можно изменить на любое число*

**for** i **in** range(1, num\_users + 1):

x = np.random.randint(300, 700)

y = np.random.randint(300, 700)

ue = UserEquipment(UE\_ID=i, x=x, y=y)

*# Настраиваем размер буфера*

ue.buffer.ADD\_PACKET(np.random.randint(5000, 20000), 0, np.random.randint(1, 5))

env.add\_user(ue)

*# Запускаем симуляцию*

env.run\_simulation(duration\_ms=20)

*# Визуализируем результаты*

env.visualize\_resource\_grid(tti\_start=0, tti\_end=20)

env.visualize\_throughput()

env.visualize\_resource\_allocation()

env.print\_statistics()

Рекомендации по дальнейшему улучшению проекта

1. **Реализация других алгоритмов планирования**:
   * Proportional Fair (PF)
   * Maximum Throughput (MT)
   * Best CQI
   * QoS-aware
2. **Улучшение моделей движения пользователей**:
   * Реализация различных паттернов перемещения
   * Моделирование препятствий
   * Учет скорости и траектории движения
3. **Реализация детальной модели канала**:
   * Учет замираний сигнала
   * Моделирование интерференции
   * Реализация MIMO
4. **Расширение возможностей анализа результатов**:
   * Экспорт статистики в CSV или Excel
   * Создание интерактивных графиков
   * Сравнение различных алгоритмов планирования
5. **Реализация механизмов управления мощностью**:
   * Адаптивное управление мощностью передачи
   * Учет энергоэффективности
6. **Реализация более сложных моделей трафика**:
   * VoIP
   * Потоковое видео
   * FTP
7. **Добавление возможности воспроизведения записанных сценариев**:
   * Загрузка трассировок реального трафика
   * Воспроизведение заданных паттернов движения
8. **Учет задержек и джиттера**:
   * Моделирование задержек при обработке данных
   * Учет джиттера при передаче данных
9. **Добавление GUI для управления симуляцией**:
   * Графический интерфейс для настройки параметров
   * Визуализация результатов в реальном времени
10. **Реализация распределенных вычислений**:
    * Параллельное выполнение симуляций
    * Распределение обработки по нескольким узлам