Обзор лабораторной работы: Что это такое и зачем?

Эта лабораторная работа посвящена изучению **системы массового обслуживания (СМО) с комбинированной дисциплиной обслуживания (ДО)**. Это комбинация из предыдущих: заявки ждут в очереди, если канал занят, но очередь (K мест, включая обслуживание), так что если очередь полная — заявка теряется (отказ). Модель для сетей с буфером ограниченного размера, как в пакетных сетях. Ключевой показатель — **вероятность потерь p** (доля потерянных заявок), которая зависит от нагрузки ρ = a (интенсивность поступления), длины очереди (K-1 мест ожидания) и распределений.

Цель: Освоить AnyLogic, построить модель M/M/1/K (марковская), сравнить симуляцию с теорией, затем модифицировать в G/G/1/K (общие распределения) и изучить влияние коэффициентов вариации (Ca для поступления, Cb для обслуживания).

План работы (по методичке, хотя в содержании акцент на G/G, а не многофазной, возможно опечатка):

1. Построить и валидировать модель M/M/1/K.
2. Сравнить имитацию и аналитику для M/M/1/K.
3. Модифицировать в многофазную (или G/G/1/K) и исследовать.
4. Выводы.

Чтобы защититься: Покажи модель в AnyLogic, таблицы, графики p от a для разных K. Объясни формулы, почему результаты сходятся/отличаются. Я разобью просто, с советами.

1. Что такое СМО с комбинированной ДО? (Теоретическая основа)

- **Заявки**: Приходят по пуассоновскому потоку (M — экспоненциальные интервалы, среднее τ=1/a).
- **Очередь**: Ограниченная, K-1 мест ожидания (общее K позиций: 1 на обслуживании + K-1 в очереди).
- **Обслуживание**: Один канал (1), время t — экспоненциальное (M, среднее 1) или общее (G).
- **Правила**:
  - Заявка приходит.
  - Если канал свободен — обслуживается.
  - Если занят, но очередь не полная — ждёт.
  - Если очередь полная — теряется (отказ).

- **Нагрузка ρ = a = λ / μ**: λ=a (интенсивность поступления), μ=1 (для t=1). Если ρ>1, потери высокие; ограниченная очередь добавляет потерь даже при ρ<1.
- **Коэффициенты вариации**: Ca = σ_τ / τ (для интервалов), Cb = σ_t / t (для обслуживания). Для M — Ca=Cb=1; для детерминированного — 0.
- **Ключевой показатель**: p = потери / всего заявок.

Это модель для сетей с有限ыми буферами (routers). Симулируешь, чтобы увидеть, как p зависит от a, K, Ca, Cb.

2. Построение модели в AnyLogic (Шаг 2.1–2.2)

- **Элементы модели** (из Process Modeling Library, можно из прошлой лабы):
  - **Source**: Генерирует заявки. Arrivals: exponential(1/a) (интервалы). New agent: Agent или Packet.
  - **SelectOutput**: Разветвитель. Condition: queue.size() < queue.capacity() (если место в очереди — в queue; else — в sink1, отказ).
  - **Queue**: Очередь. Capacity: v (K-1? В методичке queue.capacity=20 или slider). Type: Agent.
  - **Delay**: Обслуживание. Delay time: exponential(1). Capacity: 1.
  - **Sink**: Успех.
  - **Sink1**: Отказы. On enter: p = (double)sink1.count() / source.count().
  - **Параметр a**: double, нагрузка (default 0.9?).
  - **Переменная p**: double, вероятность потерь.
  - **Slider?**: Для изменения K в runtime.
- **Структура**: Source → SelectOutput → Queue → Delay → Sink (успех). Из SelectOutput вниз → Sink1 (отказ). (Рис.1).

Настройки по рисункам 2–5:

- Source: exponential(1/a).
- Queue: Capacity=20 (или v), infinite=false.
- Delay: exponential(1), capacity=1.
- Sink1: p = (double)sink1.count / source.count.

3. Валидация модели (Шаг 2.3)

- **Проверка работы (2.3.1)**: Запусти с a=0.9, max скорость. p ≈0.012 (для определённого K, вероятно queue.capacity=20 или по формуле). Если ошибки — проверь условия в selectOutput.
- **Сравнение с аналитикой (2.3.2)**:

- **Формула для M/M/1/K**: p = (1 - ρ) * ρ^K / (1 - ρ^{K+1}), где K — общее (очередь + сервер).
    - Запусти для a=0.1..0.99, K-1=0,1,5,10,20,50 (меняй queue.capacity = K-1).
    - Заполни таблицу 1: Для каждого a и K-1 — p из симуляции (ИМ), p из формулы (АМ).
    - Графики: p от a для разных K-1. Ожидай: p растёт с a, падает с ростом K (больше очереди — меньше потерь). При K→∞ — p=0 для ρ<1.

Результаты ИМ близки к АМ (разница от случайности, запусти на >10^5 заявок).

4. Исследование СМО G/G/1/K (Шаг 3)

- **Построение (3.1)**: Измени source и delay на общие распределения (G). Добавь slider для K? Структура та же (рис.7).
- **Настройки (3.2, рис.8–13)**:
    - Source: Arrivals — general, напр. triangular или uniform для Ca≠1.
    - Queue: Capacity — переменная, напр. slider.value (для изменения K-1).
    - Delay: Delay time — general, напр. uniform(0,2) для Cb=1/√3≈0.577.
    - Sink/Sink1: Как раньше.
    - Slider: Для capacity queue, min=0, max=50.
- **Эксперименты (3.2)**:
    - Варьируй a=0.1..0.99, K-1=5,10.
    - Для каждого: Выбери распределения, посчитай Ca (CV поступления), Cb (CV обслуживания), т (средний интервал=1/a), t (среднее обслуживание=1).
    - p из симуляции (им), из аппроксимации (ам): p ≈ [ρ^{2K} * (Ca^2 + Cb^2)/2] / [1 - ρ^{2(K+1)} * (Ca^2 + Cb^2)/2 + что-то], wait, формула в методичке: p ≈ (ρ^2 (Ca^2 + Cb^2)/2 * (1-ρ)) / (1 - ρ^{2K+1} (Ca^2 + Cb^2)/2) или точнее как дана.
    - Заполни таблицу 2: Для каждого a и K-1 — Ca, Cb, т, t, им, ам.
    - Графики: p от a для K-1=5,10.

Выводы: Для G/G p зависит от Ca, Cb — если Ca,Cb<1 (менее случайные), p ниже, чем в M/M (Ca=Cb=1). Аппроксимация работает для высоких ρ. Симуляция показывает реальные отличия от марковских моделей.

5. Выводы (Ответы на вопросы — готовься сказать преподавателю)

1. **По построению имитационной модели СМО с комбинированной ДО**: Модель на базе source-selectOutput-queue-delay-sink/sink1. SelectOutput проверяет место в очереди. Легко модифицировать из предыдущих лаб. Валидация подтверждает: p стабилизируется быстро.
2. **По имитации и аналитике M/M/1/K**: ИМ близко к АМ (разница <0.01 при больших runs). p растёт с a, но ограниченная очередь добавляет потерь даже при низкой нагрузке. Для

малого K p высокое; для большого K приближается к M/M/1 (бесконечная очередь, $p=0$ при $\rho<1$).

3. **По исследованию СМО G/G/1/K**: В общем случае p ниже, если распределения детерминированные (Ca,Cb→0), т.к. меньше вариабельности — меньше перегрузок. Аппроксимация даёт хорошее приближение, но ИМ точнее для не-марковских потоков. Полезно для реальных систем, где потоки не пуассоновские.