

**Національний Технічний Університет України КПІ**

Факультет Інформатики та Обчислювальної Техніки  
Кафедра інформатики та програмної інженерії

**Практична робота №3**

З дисципліни «Моделювання систем»

ПОБУДОВА ІМІТАЦІЙНОЇ МОДЕЛІ СИСТЕМИ З ВИКОРИСТАННЯМ

ФОРМАЛІЗМУ МОДЕЛІ МАСОВОГО ОБСЛУГОВУВАННЯ

**Перевірив:**

Асистент

Бернатович Анатолій Олександрович

Оцінка:\_\_\_\_\_\_\_\_\_\_\_**Виконав:**

Студент групи ІТ-92

Яцевський О. І.

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_

Завдання до практичної роботи

1. Реалізувати універсальний алгоритм імітації моделі масового обслуговування з багатоканальним обслуговуванням, з вибором маршруту за пріоритетом або за заданою ймовірністю. **30 балів.**
2. Для наступного тексту задачі скласти формалізовану модель масового обслуговування та реалізувати її з використанням побудованого універсального алгоритму **(30 балів)**:

У банку для автомобілістів є два віконця, кожне з яких обслуговується одним касиром і має окрему під'їзну смугу. Обидві смуги розташовані поруч. З попередніх спостережень відомо, що інтервали часу між прибуттям клієнтів у годину пік розподілені експоненційно з математичним очікуванням, рівним 0,5 од. часу. Через те, що банк буває переобтяжений тільки в годину пік, то аналізується тільки цей період. Тривалість обслуговування в обох касирів однакова і розподілена експоненційно з математичним очікуванням, рівним 0,3 од. часу. Відомо також, що при рівній довжині черг, а також при відсутності черг, клієнти віддають перевагу першій смузі. В усіх інших випадках клієнти вибирають більш коротку чергу. Після того, як клієнт в'їхав у банк, він не може залишити його, доки не буде обслугований. Проте він може перемінити чергу, якщо стоїть останнім і різниця в довжині черг при цьому складає не менше двох автомобілів. Через обмежене місце на кожній смузі може знаходитися не більш трьох автомобілів. У банку, таким чином, не може знаходитися більш восьми автомобілів, включаючи автомобілі двох клієнтів, що обслуговуються в поточний момент касиром. Якщо місце перед банком заповнено до границі, то клієнт, що прибув, вважається втраченим, тому що він відразу ж виїжджає. Початкові умови такі: 1) обидва касири зайняті, тривалість обслуговування для кожного касира нормально розподілена з математичним очікуванням, рівним 1 од. часу, і середньоквадратичним відхиленням, рівним 0,3 од. часу; 2) прибуття першого клієнта заплановано на момент часу 0,1 од. часу; 3) у кожній черзі очікують по два автомобіля.

Визначити такі величини: 1) середнє завантаження кожного касира; 2) середнє число клієнтів у банку; 3) середній інтервал часу між від'їздами клієнтів від вікон; 4) середній час перебування клієнта в банку; 5) середнє число клієнтів у кожній черзі; 6) відсоток клієнтів, яким відмовлено в обслуговуванні; 7) число змін під'їзних смуг.

1. Для наступного тексту задачі скласти формалізовану модель масового обслуговування та реалізувати її з використанням побудованого універсального алгоритму **(40 балів)**:

У лікарню поступають хворі таких трьох типів: 1) хворі, що пройшли попереднє обстеження і направлені на лікування; 2) хворі, що бажають потрапити в лікарню, але не пройшли повністю попереднє обстеження; 3) хворі, які тільки що поступили на попереднє обстеження. Чисельні характеристики типів хворих наведені в таблиці:

|  |  |  |
| --- | --- | --- |
| *Тип хворого* | *Відносна частота* | *Середній час*  *реєстрації, хв* |
| 1 | 0,5 | 15 |
| 2 | 0,1 | 40 |
| 3 | 0,4 | 30 |

При надходженні в приймальне відділення хворий стає в чергу, якщо обидва чергових лікарі зайняті. Лікар, який звільнився, вибирає в першу чергу тих хворих, що вже пройшли попереднє обстеження. Після заповнення різноманітних форм у приймальне відділення хворі 1 типу ідуть прямо в палату, а хворі типів 2 і 3 направляються в лабораторію. Троє супровідних розводять хворих по палатах. Хворим не дозволяється направлятися в палату без супровідного. Якщо всі супровідні зайняті, хворі очікують їхнього звільнення в приймальному відділенні. Як тільки хворий доставлений у палату, він вважається таким, що завершив процес прийому до лікарні.

Хворі, що спрямовуються в лабораторію, не потребують супроводу. Після прибуття в лабораторію хворі стають у чергу в реєстратуру. Після реєстрації вони ідуть у кімнату очікування, де чекають виклику до одного з двох лаборантів. Після здачі аналізів хворі або повертаються в приймальне відділення (якщо їх приймають у лікарню), або залишають лікарню (якщо їм було призначено тільки попереднє обстеження). Після повернення в приймальне відділення хворий, що здав аналізи, розглядається як хворий типу 1.

У наступній таблиці приводяться дані по тривалості дій (хв):

|  |  |
| --- | --- |
| *Величина* | *Розподіл* |
| Час між прибуттями в приймальне відділення | Експоненціальний з математичним сподіванням 15 |
| Час слідування в палату | Рівномірне від 3 до 8 |
| Час слідування з приймального відділення в лабораторію або з лабораторії в приймальне відділення | Рівномірне від 2 до 5 |
| Час обслуговування в реєстратуру  лабораторії | Ерланга з математичним сподіванням 4,5 і *k*=3 |
| Час проведення аналізу в лабораторії | Ерланга з математичним сподіванням 4 і *k*=2 |

Визначити час, проведений хворим у системі, тобто інтервал часу, починаючи з надходження і закінчуючи доставкою в палату (для хворих типу 1 і 2) або виходом із лабораторії (для хворих типу 3). Визначити також інтервал між прибуттями хворих у лабораторію.

Теоретичні відомості

Для данної лабораторної роботи була обрана універсальна мова програмування Python. Такий вибір був здійсненний через широку сферу застосування мови, обширний асортимент математичних фунцій у бібліотеках як numpy, а також наявність зручних інструментів візуалізаціїї і вводу / виводу.

Приблизну структуру об’єктно-орієнтованої програми представимо діаграмою класів:

Timeline

Description automatically generated

Модель складається з елементів, які беруть за основу клас Element. Таким чином, всі наступні класи матимуть всі наступні властивості :

|  |  |
| --- | --- |
| Назва | Коментар |
| tcurr | поточний момент часу |
| tnext | момент часу наступної події |
| delayMean | середеє значення часової затримки |
| delayDev | середнє квадратичне відхилення часової затримки |
| getDelay() | функція для розрахуноку часової затримками |
| inAct() | розрахунки, які відбуваються на вході в компонент |
| outAct() | виході з компоненту |
| nextElement | Наступний елемент моделі |

Важливим для з’єднання елементів є полу nextElement, яку встановлює зв’язки між елементами, об’єднуючи їх в одну модель.

Модельі відповідає клас Model, що містить в собі список елементів і здійсюнє імітацію на інтервалі часу time. Імітація здійснюється за наступним принципом: визначається момент найближчої події, просувається час в момент найближчої події та здійснюється відповідна подія.

Для оптимізації розрахунків ми можемо вести розрахунки для процесів, які відбуваються в поточний момент часу, паралельно.

Виконання лабораторної роботи

Ми створили дві моделі: одну з одним каналом і одну з двома каналами. Перша модель виглядає так:

c1 = Create(delay\_mean=5, name='CREATOR', distribution='exp')  
p1 = Process(max\_queue=3, delay\_mean=5, distribution='exp')  
  
c1.next\_element = [p1]  
elements = [c1, p1]  
model = Model(elements)  
model.simulate(1000)

* "delay\_mean" - це математичне сподівання
* «розподіл» — тип розподілу часу
* "max\_queue" — максимальний розмір черги на пристрій
* "ім'я" - це назва пристрою
* "c1.next\_element" вказує на наступний пристрій у моделі (у цьому випадку після "CREATE" буде "PROCESS")
* "model.simulate" - це виклик методу для запуску симуляції

Друга модель аналогічна, з додаванням параметра «n\_channel», який визначає кількість каналів.

c1 = Create(delay\_mean=5, name='CREATOR', distribution='exp')  
p1 = Process(max\_queue=3, n\_channel=2, delay\_mean=5, distribution='exp')  
  
c1.next\_element = [p1]  
elements = [c1, p1]  
model = Model(elements)  
model.simulate(1000)

Тепер ми запустимо обидві моделі та порівняємо результати. Результати моделювання для моделі без каналів такі:

Text

Description automatically generated

* "кількість" - кількість елементів, оброблених пристроєм
* «Середня довжина черги» — середня довжина черги пристрою
* «Імовірність відмови» — це ймовірність того, що пристрій не зможе обробити завдання
* «Середнє навантаження» — середнє навантаження пристрою
* «Глобальна максимальна спостережувана довжина черги» — це максимальний розмір черги, який спостерігається в моделі
* «Загальна середня довжина черги» — це середній розмір черги в усій моделі
* «Глобальна ймовірність відмови» — це ймовірність того, що вся модель не зможе обробити завдання
* «Загальне середнє навантаження» — це середнє навантаження всіх пристроїв у моделі

Результати моделювання для моделі з каналами аналогічні, але з додатковими каналами для обробки завдань

Text

Description automatically generated

Як можна побачити, через додавання ще одного каналу, у нас сильно знизилось завантаження пристрою а також ймовірність відмови. Також знизилась середня довжина черги. Тобто, з використанням канальної системи можна досягти кращого результату опрацювання елементів пристроєм.

Ми створили кілька моделей, щоб продемонструвати, як імовірність і пріоритет можна використовувати для маршрутизації подій у системі. У першій моделі ми встановили ймовірність вибору пристроїв p2 і p3 рівними 0,9 і 0,1 відповідно.

c1 = Create(delay\_mean=5, name='CREATOR', distribution='exp')  
p1 = Process(max\_queue=3, delay\_mean=5, distribution='exp')  
p2 = Process(max\_queue=3, delay\_mean=5, distribution='exp')  
p3 = Process(max\_queue=3, delay\_mean=5, distribution='exp')  
  
c1.next\_element = [p1]  
p1.next\_element = [p2, p3]  
p1.probability = [0.9, 0.1]  
elements = [c1, p1, p2, p3]  
model = Model(elements)  
model.simulate(1000)

Text

Description automatically generated

Результати моделювання показують, що третій пристрій з меншою ймовірністю вибору має менше оброблених елементів і менше навантаження, ніж другий пристрій.

У другій моделі ми встановили пріоритет для пристроїв p2 і p3 на 2 і 1 відповідно, в результаті чого третій пристрій вибирається частіше через його вищий пріоритет.

c1 = Create(delay\_mean=5, name='CREATOR', distribution='exp')  
p1 = Process(max\_queue=3, delay\_mean=5, distribution='exp')  
p2 = Process(max\_queue=3, delay\_mean=5, distribution='exp')  
p3 = Process(max\_queue=3, delay\_mean=5, distribution='exp')  
p1.priority = [2, 1]  
c1.next\_element = [p1]  
p1.next\_element = [p2, p3]  
elements = [c1, p1, p2, p3]  
model = Model(elements)  
model.simulate(1000)

Text

Description automatically generated

Також ми побудували модель автомобільного банку, як наведено у завданні із вказанням початкових значень. Для цього ми модифікували універсальний алгоритм, який ми реалізували у минулому завданні. У моделі автомобільного банку перша смуга має вищу перевагу, що призводить до того, що перший касир обробляє більше подій.

c1 = CreateBank(delay\_mean=0.5, name='CREATOR', distribution='exp')  
p1 = ProcessBank(max\_queue=3, delay\_mean=0.3, name='CASHIER\_1', distribution='exp')  
p2 = ProcessBank(max\_queue=3, delay\_mean=0.3, name='CASHIER\_2', distribution='exp')  
  
c1.next\_element = [p1, p2]  
  
# Обидва касири зайняті  
p1.state[0] = 1  
p2.state[0] = 1  
  
# Тривалість  
# обслуговування для кожного касира нормально розподілена з  
# математичним очікуванням, рівним 1 од. часу, і середньоквадратичним  
# відхиленням, рівним 0,3 од. часу  
p1.t\_next[0] = fun.norm(1, 0.3)  
p2.t\_next[0] = fun.norm(1, 0.3)  
  
# Прибуття першого клієнта заплановано на момент часу 0,1 од. часу  
c1.t\_next[0] = 0.1  
  
# У кожній черзі очікують по два автомобіля.  
p1.queue = 2  
p2.queue = 2  
  
element\_list = [c1, p1, p2]  
bank = ModelBank(element\_list, balancing=[p1, p2])  
bank.simulate(1000)

Де “fun.norm” – функція визначення нормально розподілених значень в залежності від параметрів. “CASHIER\_1” та “CASHIER\_2” – відповідні касири за номерами. Моделювання проходить на проміжку часу в 1000 часових одиниць

Результати моделювання надають наступну інформацію:

Text

Description automatically generated

Навантаження касира 1 0,5, касира 2 0,115.

Середня кількість клієнтів у банку 0,87.

Середній інтервал часу між відходами клієнтів від вікон становить 1,64.

Середній час перебування клієнта в банку становить 0,0014.

Середня кількість клієнтів у першій черзі – 0,21, у другій – 0,041.

Відсоток клієнтів, яким було відмовлено в обслуговуванні, становить 0,029%.

Кількість змін смуги – 21.

Побудуємо формалізовану модель масового обслуговування лікарні, як наведено у завданні. Для цього, як і у попередньому завданні, ми модифікували базовий алгоритм.

c1 = CreateHospital(delay\_mean=15.0, name='CREATOR\_1', distribution='exp')  
p1 = ProcessHospital(max\_queue=100, n\_channel=2, name='RECEPTION', distribution='exp')  
p2 = ProcessHospital(max\_queue=100, delay\_mean=3.0, delay\_dev=8, n\_channel=3, name='FOLLOWING\_TO\_THE\_WARD',  
 distribution='unif')  
p3 = ProcessHospital(max\_queue=0, delay\_mean=2.0, delay\_dev=5, n\_channel=10, name='FOLLOWING\_TO\_THE\_LAB\_RECEPTION',  
 distribution='unif')  
p4 = ProcessHospital(max\_queue=100, delay\_mean=4.5, delay\_dev=3, n\_channel=1, name='LAB\_REGISTRY',  
 distribution='erlang')  
p5 = ProcessHospital(max\_queue=100, delay\_mean=4.0, delay\_dev=2, n\_channel=1, name='EXAMINATION',  
 distribution='erlang')  
p6 = ProcessHospital(max\_queue=0, delay\_mean=2.0, delay\_dev=5, n\_channel=10, name='FOLLOWING\_TO\_THE\_RECEPTION',  
 distribution='unif')  
  
d1 = DisposeHospital(name='EXIT1')  
d2 = DisposeHospital(name='EXIT2')  
  
c1.next\_element = [p1]  
p1.next\_element = [p2, p3]  
p2.next\_element = [d1]  
p3.next\_element = [p4]  
p4.next\_element = [p5]  
p5.next\_element = [d2, p6]  
p6.next\_element = [p1]  
  
p1.prior\_types = [1]  
  
p1.required\_path = [[1], [2, 3]]  
p5.required\_path = [[3], [2]]  
  
elements = [c1, p1, p2, p3, p4, p5, p6, d1, d2]  
  
model = ModelHospital(elements)  
model.simulate(1000)

Де «CREATOR\_1» і «RECEPTION» - прибуття в приймальне відділення, «FOLLOWING\_TO\_THE\_WARD» - слідування в палату, «FOLLOWING\_TO\_THE\_LAB\_RECEPTION» - слідування в лабораторію, «'LAB\_RECEPTION» - обслуговування в реєстратурі лабораторії, «EXAMINATION» - проведення огляду. розбір, "НА\_РЕЦЕПЦІЮ" - переїзд в приймальне відділення.

Також для цієї роботи ми додали такий елемент як DisposeHospital. Необхідно вказати закінчення послуги, а також зібрати інформацію про кількість обслужених пацієнтів.

Text

Description automatically generated

Давайте запустимо програму і побачимо результати:

Дамо відповіді на запитання завдання:

1. Інтервал часу, починаючи з надходження і закінчуючи доставкою в палату – 51.42 (оскільки після здачі аналізів хворий типу два буде вважатися як хворий типу один, а тому значення одне) Інтервал часу виходу із лабораторії – 70.934.
2. Інтервал часу між прибуттям хворих у лабораторію – 38.53.

Висновок

Підводячи підсумок, ми створили кілька моделей, щоб продемонструвати, як імовірність і пріоритет можна використовувати для маршрутизації подій у системі. Результати моделювання дають змогу зрозуміти ефективність моделей і наслідки зміни ймовірності та пріоритету маршрутизації подій.