Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Моделі і методи оцінки продуктивності інформаційних систем
(тема)

Виконав:

студент II курсу, групи СПм-23-1

Холодний М.О.
(прізвище, ініціали)

Спеціальність

123 «Комп’ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма

Системне програмування
(повна назва освітньої програми)

Керівник: проф. Горбачов В.О.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп'ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 20___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Холодному Миколі Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Моделі і методи оцінки продуктивності інформаційних систем

затверджена наказом по університету від " 22 " 11 2024 р. № 1236 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 20.01.2025

3. Вхідні дані до роботи

3.1 Інформаційні системи з мережевою структурою;

3.2 Аналітичні і імітаційні методи аналізу продуктивності інформаційних систем

3.3 Експериментальний підхід до оцінки продуктивності інформаційних систем

3.4 Інструменти: імітаційні системи моделювання

4. Перелік питань, що потрібно опрацювати у роботі

4.1 Розробка клієнт-серверної моделі сайту

4.2 Розробка структурної моделі взаємодії компонентів сайту.

4.3 Розробка аналітичної моделі часу відповіді

4.4 Планування експерименту

4.5 Оцінка ефективності за допомогою аналітичного та імітаційного моделювання

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) _____

Слайд-презентація — 15 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз інформаційних систем та моделей	23 .11.24 -31.11.24	
2	Постановка задачі	23 .11.24-30.11.24	
3	Розробка клієнт-серверної моделі сайту	30.11.24-30.12.24	
4	Розробка аналітичної і імітаційної моделі	30.11.24-30.12.24	
5	Розробка програмного забезпечення	01.12.24-25.12.24	
6	Планування та проведення експерименту	01.11.24-05.01.25	
7	Проведення аналізу результату	01.11.24-16.01.25	
8	Оформлення роботи	05.01.25-20.01.25	

Дата видачі завдання 23 .11.2024 р.

Студент _____

Керівник роботи _____ проф. Горбачов В.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 68 с., 16 рис., 9 табл., 2 дод., 17 джерел.

КЛІЄНТ-СЕРВЕРНА МОДЕЛЬ, ЗБАЛАНСОВАНА СИСТЕМА, ОПЕРАЦІЙНИЙ АНАЛІЗ, ЧАС ВІДГУКУ

Метою кваліфікаційної роботи є розробка клієнт-серверної моделі сайту і оцінка ефективності за допомогою аналітичного та імітаційного моделювання.

У ході виконання кваліфікаційної роботи було розроблено структурну модель взаємодії компонентів сайту і запропоновано вимоги до їх моделей системного рівня. Розроблена аналітична модель часу відповіді. Реалізован план обчислювального експерименту і проведен порівняльний аналіз ефективності сайту з використанням аналітичних і імітаційних методів моделювання.

ABSTRACT

Master’s thesis: 68 pages, 16 figures, 9 tables, 2 appendices, 17 sources.

CLIENT-SERVER MODEL, BALANCED SYSTEM, OPERATIONAL ANALYSIS, RESPONSE TIME

The purpose of the qualification work is to develop a client-server model of the site and evaluate its effectiveness using analytical and simulation modeling.

In the course of the qualification work, a structural model of the interaction of site components was developed and requirements for their system-level models were proposed. An analytical model of the response time was developed. A computational experiment plan was implemented and a comparative analysis of the site's effectiveness was carried out using analytical and simulation modeling methods

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

DES - дискретно-подійна система (англ. Discrete – Event System)

CS – складна система (англ. complex system)

SM - імітаційна модель (англ. Simulation model)

API – інтерфейс прикладних програм (англ.,Application programming interface)

DA – область даних (англ.,Data area)

DPS – системи розподіленої обробки (англ.,Distributed Processing System)

ISP – інтернет-провайдер (англ.,Internet Service Provider)

MAS – многоагентная система (англ.,Multi Agent System)

OS – операційна система (англ.,Operating System)

PE – показники продуктивності (англ.,Performance Indicators)

QoS – якість обслуговування (англ.,Guality of Service)

ВСТУП

ІТ-системи стають все більш поширеними та допомагають підтримувати більшість аспектів повсякденного життя. Інтернет допоміг прискорити швидкість інтеграції ІТ у соціальні системи. Люди покладаються на ІТ-системи для вирішення більшості своїх основних людських і соціальних проблем, таких як здоров'я, освіта, розваги, доступ до послуг зв'язку, доступ до підтримки клієнтів, фінанси, безпека, конфіденційність, доступ до державних послуг і подорожі. Різноманітні проблеми окремих осіб і суспільства в цілому (загалом) можуть зіткнутися з серйозними збоями та високими витратами, якщо ІТ-системи не відповідають очікуваним вимогам якості обслуговування (QoS) щодо продуктивності, доступності, безпеки та зручності обслуговування. від них.

Наприклад, на дзвінок за номером 911 — номером екстреної служби в США — диспетчер має відповісти за кілька секунд, інакше життя людини може бути під загрозою. Коли фондовий ринок переживає періоди екстремальних злетів і падінь, велика кількість онлайн-трейдерів, як правило, стікається на сайти онлайн-торгівлі, що спричиняє потенційні проблеми через перевантажені та невідповідні системи. Неможливість своєчасної торгівлі може призвести до значних фінансових втрат. Під час криз здоров'я, таких як спалах нових захворювань, людям потрібно отримати легкий і швидкий доступ до медичних страхових компаній, щоб отримати дозвіл на госпіталізацію або пройти медичну процедуру.

Під час терористичної загрози основні інфраструктури, такі як телефонна та стільникова мережі, можуть стати мішенню терористів або, у разі атак на інші структури, можуть стати перевантаженими, оскільки їхня здатність обробляти дзвінки недостатня, що погіршує реагування таких системи. Операції збройних сил дедалі більше залежать від гнучкої інформаційної та комунікаційної інфраструктури, яка допомагає визначати

місцезнаходження, знаходити цілі та знищувати ворожі сили. Ця інфраструктура має бути належним чином спроектована та має відповідний розмір, щоб відповідати надзвичайним вимогам обміну інформацією на полі бою.

Більшості людей потрібно взаємодіяти з автоматизованими або напівавтоматичними системами підтримки клієнтів і очікувати майже негайної відповіді. На жаль, нерідкі випадки, коли деякі з них перебувають у режимі очікування на десятки хвилин, перш ніж з’єднатися з людиною, яка вирішить проблему або надасть необхідну інформацію. Ці ситуації викликають значне розчарування та є основною причиною втрати клієнтів компаніями.

Кількість людей, які підписуються на доступ до різноманітних комунікаційних послуг, таких як бездротові послуги та послуги доступу до Інтернету, зростає експоненціально. Зростання трафіку не було задоволено адекватним зростанням пропускної здатності системи. У результаті абоненти можуть почути неприємний запис «усі лінії зайняті, спробуйте зателефонувати пізніше» під час спроби здійснити дзвінок. Люди звикли очікувати 24/7, миттєвих і надзвичайно надійних послуг.

1 АКТУАЛЬНІСТЬ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Якість обслуговування в IT системах

IT-системи стосуються людей усюди, і необхідно докласти всіх зусиль, щоб забезпечити надійну та надійну роботу IT-систем, щоб вони відповідали потребам суспільства та доповнювали можливості користувачів [1].

У цьому розділі розглядаються такі атрибути QoS IT-системи: час відгуку, пропускна здатність, доступність, надійність, безпека, масштабованість і розширюваність.

Час відгуку. Час, потрібний системі для реакції на запит людини, називається часом відповіді. Прикладом є час, який потрібен для появи у вашому браузері сторінки з результатами пошуку в каталозі улюбленого вами книжкового онлайн-магазину. Час відгуку, який зазвичай вимірюється в секундах, може бути розбитий на кілька компонентів.

Існують три основні компоненти часу відповіді пошукового запиту на сайт електронної комерції: час браузера, час мережі та час сервера. Час браузера включає обробку та час введення/виведення, необхідний для надсилання пошукового запиту та відображення сторінки результатів. Компонент мережевого часу включає час, витрачений на передачу від браузера до Інтернет-провайдера (ISP) користувача, час, проведений в Інтернеті, і час, витрачений на спілкування між ISP на сайті електронної комерції та його сервером. Третій компонент включає весь час, залучений до обробки запиту на комерційному сайті, весь час введення/виведення, час внутрішньої мережі сайту електронної комерції. Будь-який з трьох компонентів включає час, витрачений на очікування використання різних ресурсів (процесорів, дисків і мереж). Це називається часом перевантаження. Час перевантаження залежить від кількості запитів, які обробляються

системою. Чим більше число запитів у системі, тим більше час перевантаження. У книзі ми навчимося обчислювати час перевантаження за допомогою моделей продуктивності. Пропускна здатність. Швидкість, з якою запити виконуються з комп'ютерної системи, називається пропускною здатністю та вимірюється в операціях за одиницю часу. Характер операції залежить від відповідної комп'ютерної системи. Розглядаючи показник пропускної здатності, необхідно переконатися, що відповідна операція чітко визначена. Наприклад, у системі онлайн-обробки транзакцій (OLTP) пропускна здатність зазвичай вимірюється в транзакціях за секунду (tps). Однак транзакції можуть суттєво відрізнятися за характером і обсягом ресурсів, які вони вимагають від системи OLTP. Отже, для того, щоб значення пропускної здатності було значущим, потрібно охарактеризувати тип транзакції, який розглядається під час звітування про пропускну здатність. У деяких випадках ця характеристика виконується з посиланням на добре встановлений галузевий еталон. Наприклад, Рада з обробки транзакцій (TPC) визначає еталон для систем OLTP, який називається TPC-C, який визначає комбінацію транзакцій, типових для системи введення замовлень. Показник продуктивності, визначений еталонним тестом, вимірює кількість замовлень, які можна повністю обробити за хвилину, і виражається в tpm-C.

Пропускна здатність є функцією навантаження, що пропонується системі, і максимальної потужності системи для обробки роботи, як показано в наступному прикладі.

Припустімо, що операція введення-виведення на диску в системі OLTP займає в середньому 10 мс. Якщо диск постійно зайнятий (тобто його використання становить 100%), то він безперервно виконуватиме операції введення-виведення зі швидкістю одна операція введення-виведення кожні 10 мс або 0,01 с. Таким чином, максимальна пропускна здатність диска становить 100 (= 1/ 0.01) операцій вводу-виводу в секунду. Але якщо швидкість, з якою запити введення-виведення надсилаються на диск, менше 100 запитів/с, то

його пропускна здатність буде дорівнювати швидкості, з якою надсилаються запити. Це призводить до виразу

Пропускна здатність = мінімум [потужність сервера, запропоноване навантаження]

Цей вислів слід пояснювати припущенням, що надходять запити не «змінюють свою думку», якщо система зайнята, як зазвичай трапляється на веб-сайтах.

Теоретично і експериментально доведено [4,5] що, демонструє майже лінійне збільшення при невеликих навантаженнях, а потім насичується при максимальному значенні, коли один із системних ресурсів досягає 100% використання. Однак у деяких випадках при високих загальних навантаженнях пропускна спроможність може фактично зменшитися в міру подальшого збільшення навантаження. Поява трешу відбувається, коли комп'ютерна система з недостатнім обсягом основної пам'яті витрачає значну кількість циклів процесора та пропускну здатність вводу-виводу на обробку помилок сторінки, а не на обробку робочого навантаження. Це може статися через те, що при високих навантаженнях надто багато процесів конкурують за фіксований обсяг основної пам'яті. Оскільки кожен процес отримує менше пам'яті для свого робочого набору, частота помилок сторінок значно зростає, а пропускна здатність зменшується. Операційна система постійно витрачає свій час на виконання додаткових накладних операцій (через збільшення навантаження, яке зменшує час, який ЦП може бути виділено для процесів. Це ще більше збільшує відставання, що призводить до спіралі зниження продуктивності, яка може певним чином паралізувати систему схоже на пробку.

Важливим моментом при оцінці комп'ютерних систем є визначення максимально ефективної пропускної здатності цієї системи та способів її досягнення.

Доступність. Уявіть, що ви заходите в книжковий інтернет-магазин і отримуєте в результаті відповідну сторінку. Ви, ймовірно, розчаруєтесь і

можете звернутися до іншого книжкового інтернет-магазину, щоб купити книгу, яку шукаєте. Наслідки недоступності системи можуть бути набагато серйознішими, ніж втрата клієнтів. Довіра та репутація компанії є життєво важливими. Як зазначено у [15], перебої в обслуговуванні можуть навіть загрожувати життю та майну.

Доступність визначається як частка часу, протягом якого система працює та доступна своїм клієнтам. Наприклад, система з доступністю 99,99% протягом тридцяти днів буде недоступна для

Рівняння

(1-0,9999) × 30 днів × 24 години/добу × 60 хв/год = 4,32 хвилини.

Для багатьох систем (наприклад, онлайн-книжковий магазин) цей рівень недоступності буде вважатися чудовим. Однак для інших систем (наприклад, систем захисту, служб 911) навіть 99,99% буде неприйнятним.

Двома основними причинами недоступності систем є збої та перевантаження. Нелагоди можуть завадити нам отримати доступ до комп’ютерної системи. Наприклад, мережеве з’єднання веб-сайту може бути перервано, і жоден користувач не зможе надіслати свої запити на інформацію. Крім того, перевантаження виникають, коли всі компоненти працюють, але система не має достатньо ресурсів для обробки великої кількості нових вхідних запитів. Ця ситуація зазвичай призводить до відхилення запитів. Наприклад, веб-сервер може відмовитися відкривати нове TCP-з'єднання, якщо досягнуто максимальної кількості з'єднань.

Нелагоди потрібно усувати швидко, щоб уникнути тривалих простоїв. Першим кроком до вирішення проблем є виявлення помилок. Потім необхідно знайти причини збоїв, щоб задіяти відповідні ресурси (наприклад, людей і матеріальні засоби), щоб повернути систему до нормального робочого стану. Таким чином, обробка відмов включає в себе виявлення відмов, діагностику відмов і відновлення від несправності.

Причини для контролю та обмеження кількості запитів, які одночасно обробляються IT-системою, полягають у гарантуванні високої якості

обслуговування прийнятих запитів і керуванням допуском. Якщо контроль допуску не використовується, час відповіді має тенденцію зростати експоненціально з навантаженням. У разі контролю допуску кількість запитів у системі обмежена, щоб час відповіді не перевищував певного порогу. Це досягається за рахунок відхилення запитів. Таким чином, у той час як прийняті запити мають прийнятний рівень обслуговування, відхилені можуть мати дуже великі затримки, щоб бути допущеними.

Надійність. Надійність системи – це ймовірність того, що вона функціонує належним чином і безперервно протягом фіксованого періоду часу [8]. Надійність і доступність тісно пов'язані поняття, але різні. Коли період часу, протягом якого обчислюється надійність, стає дуже великим, надійність прагне до доступності.

Безпека. Безпека — це комбінація трьох основних атрибутів: Конфіденційність: доступ до відповідної інформації мають лише авторизовані особи; Цілісність даних: інформація не може бути змінена неавторизованими користувачами; Доступність: відправникам повідомлення заборонено заперечувати відправлення повідомлення.

Щоб забезпечити дотримання цих властивостей, системи повинні реалізувати механізми автентифікації [5], щоб гарантувати, що кожна сторона в обміні повідомленнями впевнена, що інша дійсно є тією особою, за яку себе видає. Більшість механізмів автентифікації, які використовуються для забезпечення безпеки системи, базуються на одній або кількох формах шифрування. Деякі операції шифрування можуть бути дуже дорогими з обчислювальної точки зору. Компроміси між безпекою та продуктивністю вивчалися в [6].

Масштабованість. Система називається масштабованою, якщо її продуктивність не знижується значно зі збільшенням кількості користувачів або, еквівалентно, збільшенням навантаження на систему. Наприклад, час відгуку системи однієї системи може збільшуватися нелінійним чином із навантаженням, тоді як час відгуку другої системи демонструє набагато більш

контрольоване зростання. Розширюваність. Розширюваність — це властивість системи легко розвиватися, щоб впоратися з новими функціональними вимогами та вимогами до продуктивності. Рідко трапляється, коли нова система починає працювати. Навіть ретельний аналіз вимог не може обов'язково передбачити всі потреби користувачів системи.

Зміни в середовищі, в якому система повинна працювати (наприклад, нові закони та правила, різні бізнес-моделі), можуть вимагати, щоб система розвивалася, щоб адаптувати нові обставини. До таких показників належать час відгуку, пропускна здатність, доступність, надійність, безпека, масштабованість і розширюваність.

Заключні зауваження. IT-системи стають дедалі складнішими і містять багато тисяч або навіть мільйони взаємодіючих програмних і апаратних компонентів. Їх охоплення таке ж всеосяжне, як система управління повітряним рухом для всієї країни або система електронної комерції [14].

Розробники та аналітики систем часто не враховують вимоги до QoS під час проектування та/або аналізу IT-систем. Основною причиною цього є проста відсутність поінформованості про проблеми та доступні методи для розгляду проблем, пов'язаних з продуктивністю. У цьому розділі ми представили кілька властивостей і показників, які використовуються для оцінки якості IT-системи. До таких показників належать час відгуку, пропускна здатність, доступність, надійність, безпека, масштабованість і розширюваність. Ми також обговорили різні етапи життєвого циклу комп'ютерної системи та показали важливість вирішення проблем QoS на ранній стадії проектування, а не після розгортання системи.

1.2 Мета і задачі дослідження

Мета роботи є оцінка продуктивності сервера базі даних.

Задачі дослідження:

1) розробка аналітичної моделі серверу БД;

2) оцінка параметрів продуктивності аналітичної моделі серверу БД;

3) експериментальне визначення та аналіз вузьких місць серверу БД;

4) розробка імітаційної моделі серверу БД;

5) оцінка параметрів продуктивності імітаційної моделі серверу БД.

2 АНАЛІТИЧНІ МОДЕЛІ МЕРЕЖЕВИХ СИСТЕМ

2.1 Аналітичні моделі сервера БД

Простий приклад сервера бази даних. Розглянемо сервер бази даних, який має один центральний процесор і один диск. Транзакції бази даних надходять на сервер бази даних для виконання з певною швидкістю (наприклад, 1,5 транзакції в секунду (tps)). Під час свого виконання транзакція чергується з використанням процесора та диска, імовірно, більше одного разу. У будь-який момент часу одна транзакція може використовувати ЦП, а інша — диск, тоді як інші транзакції очікують використання ЦП або диска. Таким чином, центральний процесор і диск можна охарактеризувати як чергу з чергою очікування та пристрій, який обслуговує транзакції. На рисунку 2.1(a) показано графічне представлення, яке використовується для ілюстрації черги з одним сервером ресурсів. Транзакції надходять і t, якщо ресурс зайнятий, інакше вони починають використовувати ресурс негайно. У деяких випадках існує одна черга для кількох ресурсів (наприклад, мультипроцесор, одна черга для кількох касирів у банку). Цей тип черги представлений у F. 2.1 (b). Тут транзакція чекає в черзі, якщо всі m ресурсів зайняті. Як тільки ресурс стає доступним, він починає обслуговувати одну з транзакцій, що очікують у черзі. обслуживать одну из транзакций, ожидающих в очереди.

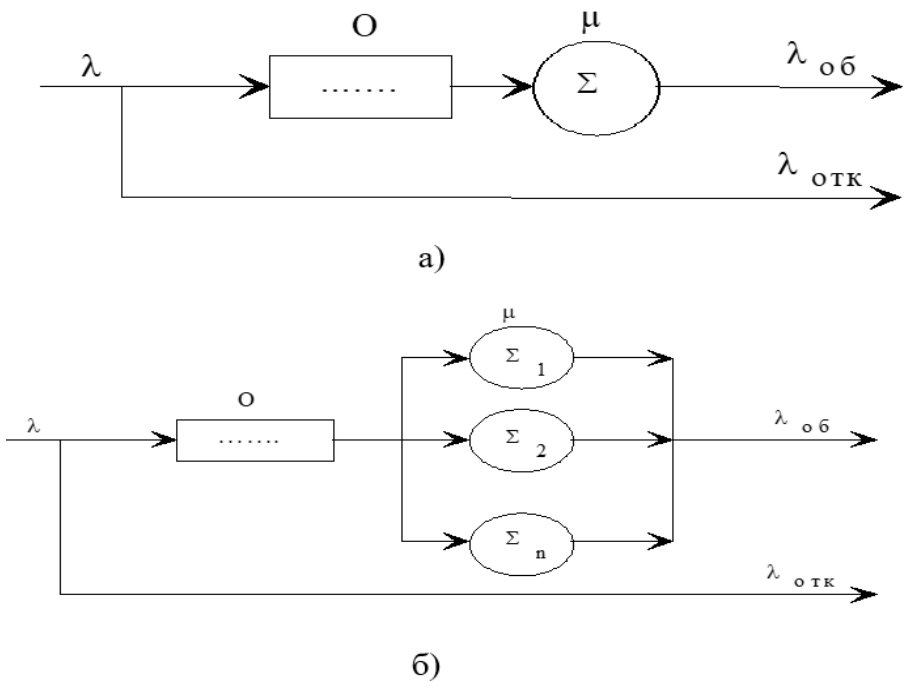


Рисунок 2.1 – СМО з відмовами і з одним сервером ресурсів (а), СМО з чергою із m серверами (б)

Щойно описана нотація може бути використана для представлення простого сервера бази даних, структуру якого наведено на рисунку 2.2. На цьому рисунку зображена мережа черг, або мережа черги (QN). Такі елементи, як транзакції бази даних, HTTP-запити та пакетні завдання, які отримують обслуговування з кожної черги в QN, зазвичай називають клієнтами. QN використовуються в цій книзі як основа для аналізу продуктивності на всіх етапах життєвого циклу системи.

Відображення існуючої системи в QN не є тривіальним. Зазвичай моделі будуються з певною метою. Наприклад, можна дізнатися, як час відповіді транзакцій бази даних змінюється залежно від швидкості, з якою транзакції надсилаються на сервер бази даних. Або можна дізнатися час відповіді, якщо сервер буде оновлено. Хороші моделі абстрагують деякі складності реальних систем, зберігаючи те, що необхідно для досягнення цілей моделі. Наприклад, QN для прикладу сервера бази даних абстрагує складність обертового магнітного диска (наприклад, дисковий контролер, дисковий кеш, обертовий

диск, плече) в один ресурс, що характеризується одним параметром (тобто середня кількість I /Os, що диск може обслуговувати за секунду). Однак, якби хтось був зацікавлений у вивченні впливу різних архітектур дисків на продуктивність, конкретні особливості архітектури вводу-виводу, які цікавлять, повинні бути явно враховані в моделі.

Щоб використовувати модель QN сервера бази даних для прогнозування продуктивності для заданої швидкості надходження транзакцій (наприклад, 1,5(tps), потрібно знати, скільки часу типова транзакція витрачає на використання ЦП і диска (тобто загальний час обслуговування). вимагається транзакцією на ЦП і диску). Потім модель використовується для обчислення часу очікування транзакції в ЦП і на диску. Загальний середній час обслуговування транзакції на ресурсі називається попитом на обслуговування. Це важливе поняття, яке буде багато разів повертатися в цій книзі. Час очікування залежить від вимог служби та навантаження системи (тобто середньої кількості одночасних транзакцій у системі).

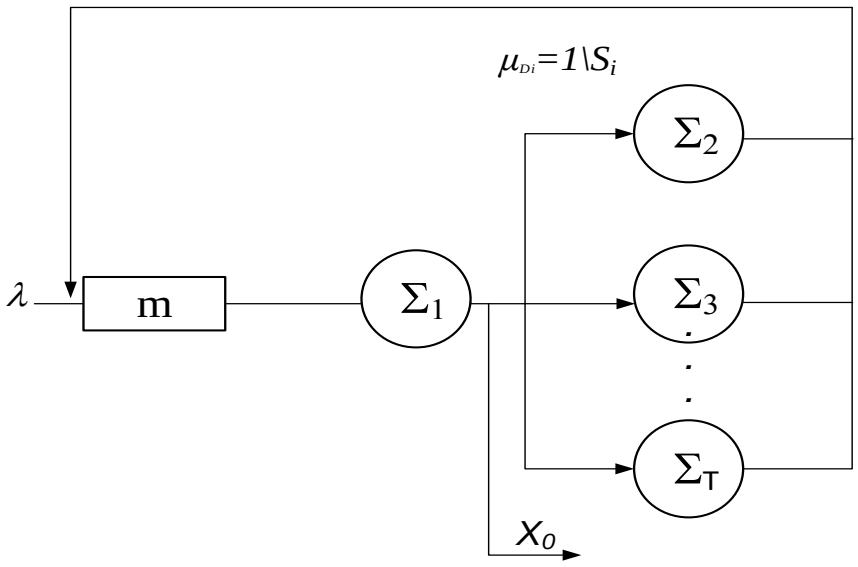


Рисунок 2.2 – Мережа систем масового обслуговування для простого сервера бази даних

Приклад сервера бази даних: Дисципліни черги. Припустімо, що сервер бази даних обслуговує і запити, і транзакції оновлення, і що час відповіді SLA на запити важливіший, ніж транзакції оновлення. У цьому випадку запитам слід надати перевагу, наприклад високий пріоритет у ЦП, щоб вони виконувалися швидше, ніж оновлення. У QN можна розглянути багато різних дисциплін масового обслуговування:

Першим прийшов перший обслужений (FCFS). Покупці обслуговуються в порядку прибуття в черзі, як у касі супермаркету.

Пріоритетна черга. Коли сервер стає доступним, наступним обслуговується клієнт із найвищим пріоритетом. Якщо є більше одного клієнта з однаковим пріоритетом, FCFS використовується для розірвання зв'язку в межах кожного класу пріоритету. Існує багато можливих варіацій черги на основі пріоритетів: статичні пріоритети проти динамічних пріоритетів, випереджувальні проти невипереджувальних, випереджувальне відновлення проти випереджувального перезапуску. У випадку статичного пріоритету пріоритет клієнта змінюється з часом. З превентивним пріоритетом прибулі клієнти з вищим пріоритетом можуть негайно захопити ресурс, який використовується клієнтом з нижчим пріоритетом. Клієнти з попередженням можуть продовжити з того місця, де вони зупинилися (випадок попереджувального відновлення), або їм, можливо, доведеться перезапустити(попереджувальний перезапуск). В роботах [10,11] результати для окремих черг із пріоритетами обговорюються як результати наближення для QN черги, які використовують дисципліни масового обслуговування на основі пріоритетів.

Кругова система (RR). У цьому випадку кожна транзакція обслуговується протягом короткого періоду часу, який називається квантовим або часовим зрізом, після чого ресурс перемикається на наступну транзакцію циклічним способом. Отже, якщо в черзі є n транзакцій, ресурс розподіляє відрізок часу для транзакцій у такому порядку: 1, 2, ⋯, n, 1, 2, ⋯. Цей тип

дисципліни планування зазвичай використовується операційними системами для планування ЦП для набору готових процесів.

2.2 Аналітичні методи оцінки продуктивності сервера БД

Основні співвідношення операційного аналізу, які дозволяють оцінити продуктивність моделі сервера БД наведені в [1].

Таблиця 2.1 – Основні співвідношення операційного аналізу

Коефіцієнта використання вузла i	U_i	$= X_i S_i$
Закон вихідного потоку	X_0	$= \sum_{i=1}^{K} X_i q_{i0}$
Закон запиту на обслуговування абр середній час обслуговування запиту	D_i	$= \frac{U_i}{X_0} = Vi \times Si$
Закон примусового потоку	X_i	$= V_i X_0$
Рівняння балансу потоків	X_j	$= \sum_{i=0}^{K} X_i q_{ij}, \ j = \overline{0,K}$
Розімкнена система, час відповіді	R	$= \sum_{i=1}^{K} V_i R_i = \overline{N} / X_0$
Розімкнена система, середнє число заявок у системі	R	$= \frac{\overline{N}}{X_0} = \sum_{i=1}^{K} V_i R_i$
Формула інтерактивного часу	R'	$= M/X_0 - Z$

Для підтримки сервера бази даних використовується комп'ютерна система, що складається з одного процесора та трьох дисків. Припустимо, що усі запити однорідні, тобто мають однакові ресурсні запити (однорідні), а також навантаження на сервер постійне. Таким чином, комп'ютерну систему можна моделювати замкнутою одноканальною СМО, як показано на фігурі.

Процесор представлений СМО з номером 1, диски мають номери від 2 до 4. Вимірювання, проведені протягом години, показали що,

1) число обслужених запитів дорівнює $C_0 = 13680$;

2) число відвідувань на запис та читання за секунду для кожного диска та відповідний коефіцієнт використання показано в таблиці 2.2.

Таблиця 2.2 – Результати вимірювального експерименту

Диск	Число відвідувань на читання в сек.	Число відвідувань на запис в сек.	Коэф. використання
1	24	8	0.30
2	28	8	0.41
3	40	10	0.54

Визначити:

1) продуктивність сервера;

2) середній час обслуговування запиту для процесора та трьох дисків сервера бази даних;

3) середню кількість відвідувань кожного диска на один запит БД;

Відомо що, коефіцієнт використання процесора дорівнює 35%;

Продуктивність кожного диска, позначена як X_i (i = 2, 3, 4), дорівнює загальному числу I/Os (запис/читання) в секунду, тобто сумі числа відвідувань на запис та читання за секунду. Використовуючи Utilization Law, ми можемо обчислити середній час обслуговування одного відвідування S_i як U_i/X_i.

Таким чином, $S_2 = U_2/X_2 = 0.30/32 = 0.0094$ сек.,

$S_3 = U_3/X_3 = 0.41/36 = 0.0114$ сек., and

$S_4 = U_4/X_4 = 0.54/50 = 0.0108$ сек..

Продуктивність сервера, X_0, дорівнює

$X_0 = C_0/T = 13680$ запитів/3600 секунд = 3.8 запит/сек.

Продуктивність сервера, X_0, дорівнює $X_0 = C_0/T = 13680$ запитів / 3600 секунд=3.8запит/сек. Використовуючи вираз запиту на обслуговування і значення коефіцієнтів використання трьох дисків, наведені у таблиці, отримаємо:

$D_{CPU} = 0.35/3.8 = 0.092$ сек/заявка,

$D_{disk1} = 0.30/3.8 = 0.079$ сек/заявка,

$D_{disk2} = 0.41/3.8 = 0.108$ и

$D_{disk3} = 0.54/3.8 = 0.142$ сек/заявка

Для порівняння в завданні 5.2

$S_2 = U_2/X_2 = 0.30/32 = 0.0094$ сек.,

$S_3 = U_3/X_3 = 0.41/36 = 0.0114$ сек., and

$S_4 = U_4/X_4 = 0.54/50 = 0.0108$ сек.

Величина V_i для кожного диска і відповідно до Forced Flow Law може бути отримана як X_i/X_0. Продуктивність сервера БД дорівнює 3.8 запиту сек. а продуктивність кожного диска, виражена в операціях введення-виводу сек. наведено у таблиці.

Отже,

$V_1 = X_1/X_0 = 32/3.8 = 8.4$ число відвідувань диска 1 однією запит БД . Аналогічно,

$V_2 = X_2/X_0 = 36/3.8 = 9.5$ и

$V_3 = X_3/X_0 = 50/3.8 = 13.2$.

2.3 Експериментальне визначення та аналіз вузьких місць серверу БД

Обчислювальний експеримент щодо залежності коефіцієнта використання від навантаження серверу

Моделі. Щоб розробити ефективний підхід до виявлення вузьких місць, необхідно мати точну та адекватну модель вузьких місць. Кілька основних визначень узагальнено в літературі [4], наприклад: у потоці продукту

виникають точки перевантаження; ресурс, потужність якого менша за запити, що до нього надходять; процес, який обмежує пропускну здатність і так далі. Вузьким місцем зазвичай визнають ресурс, який сильно обмежує продуктивність системи.

Однак для складних систем топологія з одним вузьким місцем може більше не бути репрезентативною, і потік запитів може проходити через кілька підсистем із незначними втратами запитів. Через взаємозалежність компонентів складної системи два або більше компонентів системи одночасно можуть стати насиченими і водночас створити вузькі місця в системі. а практиці перевантаження в мережах із декількома вузькими місцями може призвести до непрогнозованих системних проблем. Ось чому проблеми мережі, включаючи маршрутизацію, планування пропускної спроможності та керування потоками з кількома вузькими місцями, залишаються важливими питаннями, які потребують подальших досліджень.

Методи виявлення вузьких місць. Щоб виявити вузькі місця, спочатку необхідно визначити умови, за яких виникають вузькі місця. Щоб виявити вузькі місця, спочатку необхідно визначити умови, за яких виникають вузькі місця. Метод виявлення аналізує дані вимірювання або моделювання, щоб визначити, хто відповідає цим умовам. Різноманітність систем, стрімке зростання складності мережі та стислі часові обмеження ускладнюють точне визначення вузьких місць. Класифікація методів виявлення вузьких місць наведена в [4].

$$U_i = X_i S_i$$

Для обчислювального експерименту візьмемо таку концепцію. Вузьким місцем є вузол, значення коефіцієнту використання якого, перевищує 0,62. Очевидно, що при збільшенні навантаження

на 30% диск 4 є вузьким місцем, оскільки $U_4 = 0,71$;

ще на 30% диски 3 і 4 є вузькими місцями, оскільки $U_3 = 0,78$, $U_4 = 0,86$.

Таблиця 2.3 - Залежність коефіцієнта використання від навантаження серверу

λ	S_2	S_3	S_4	X_2	X_2	X_3	U_2	U_3	U_4
13680	0.0094	0.0114	0.0108	32	36	50	0.30	0.41	0.54
17784	0.0094	0.0114	0.0108	42	47	65	0.39	0.54	0.71
21888	0.0094	0.0114	0.0108	62	69	80	0.64	0.78	0.86

2.4 Замкнута модель оцінки продуктивності сервера БД

Структурна схема сервера БД, яка є замкнутою мережею, представлена на малюнку. Практично вона є клієнт-серверна система.

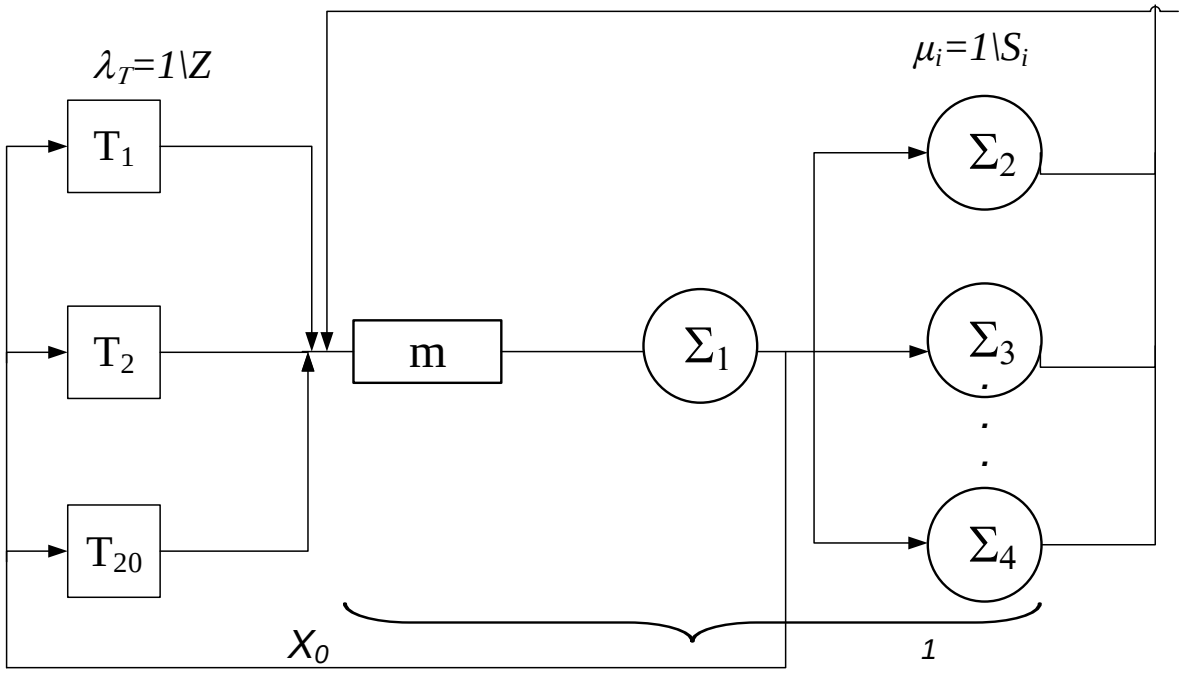


Рисунок 2.3 – Структурна схема сервера БД

Обслуговування запитів виконується одним сервером і n дисками. Вони розглядаються без деталізації внутрішньої структури як канали з тривалістю обслуговування, розподіленої за експоненціальним законом із середнім

значенням, рівним $\bar{t}_{об} = \frac{1}{\mu}$. Операційна система реалізує безпріоритетні дисципліни очікування й обслуговування в порядку надходження запитів, причому всі ресурси каналу обслуговування цілком монополізуються призначеною на обслуговування заявкою до кінця її обслуговування. Заявка, що застала центральний процесор зайнятим, займає місце в черзі, число місць в якої $m = M - n$; заявки вважаються терплячими. Формування нового запиту користувач починає лише після одержання відповіді на попередній запит, причому час, необхідний користувачу для формування чергового запиту, вважатиме розподіленим експоненціально із середнім значенням, рівним $\bar{t}_з$, що дорівнює розглядати користувача як джерело найпростішого потоку заявок з інтенсивністю $\lambda = 1/\bar{t}_з$.

Специфікація системи. Є 25 терміналів. Середній час обмірковування дорівнює 18 сек. Кожна заявка терміналу (інтерактивна заявка) генерує в середньому 20 запитів до диска. Коефіцієнт використання диска дорівнює 0,5 (50%). Середній час обслуговування запиту диском дорівнює 25 мілісекунд.

Дано: 1. U_i = 0,5 (50%); 2. Vi = Xi / X0 = 20 запит; 3. Si = 1/ μT = 25ms; 4. M = 25; 5. $t_з$= 1/ λ_T = 18c 6. m = ∞

Завдання:

1) побудувати GPSS – модель у припущенні Марківської моделі системи, тобто. що у системі протікають найпростіші потоки;

2) експериментально оцінити показники ефективності системи: продуктивність системи X_0; Середній час відповіді системи R_i.

Результати, отримані за допомогою ОА:

1) $X_0 = X_i / V_i = U_i / S_i V_i$ = 1запит/сек;

2) R_i = 25/1 – 18 = 7c.

3 СТРУКТУРА ІМІТАЦІЙНИХ СИСТЕМ

3.1 Імітаційна модель одноканальної СМО з обмеженою чергою

Для демонстрації розглянутих принципів моделювання побудуємо імітаційну модель одноканальної СМО з необмеженою чергою (рисунок 2.1а), специфікація якої був докладно розглянутий вище [1]. Припустимо, що у потоці заявок інтервал часу має рівномірний розподіл , а час обслуговування має нормальний розподіл N(m, σ). У процесі моделювання цікавить оцінка наступних показників ефективності: середнього часу очікування заявки у черзі.

Для проведення статистичного моделювання такої СМО необхідні: підпрограми імітації випадкових чисел, розподілених за рівномірним і нормальним законом, а також алгоритм її поведінки. При побудові схеми алгоритму будуть використані такі позначення:

t_0 – початок моделювання;

T-момент закінчення моделювання;

n – поточна кількість обслужених заявок;

t_j`- момент часу надходження j-ї заявки;

t_j`` - момент часу закінчення обслуговування j-ї заявки;

t_m - таймер модельного часу або поточний час;

Δt – інтервал часу між заявками;

$t_{обсл.j}$ – тривалість обслуговування j-ї заявки;

z - прапор зайнятості приладу: z = 0 - прилад вільний, z = 1 - прилад зайнятий;

r – поточне значення черги;

$t_{очj}$ – час очікування j-ї заявки у черзі;

M = {t_j`, t_j``} - масив упорядкованих моментів часу основних подій або список основних подій. Цей масив можна як другий стовпець з таблиці 7.3 попереднього розділу.

R = {t_j`} - масив упорядкованих часів надходження заявок, що знаходяться в черзі на момент часу tm;

При розробці алгоритму прийнято такі обмеження:

1) j-я заявка, для якої $t_j` > T$ не розглядається

2) j-я заявка, для якої $t_j`` < T$ вважається обслуженою

3) j-я заявка, для якої $t_j` < T$, а $t_j`` > T$ вважається такою, що отримала відмову

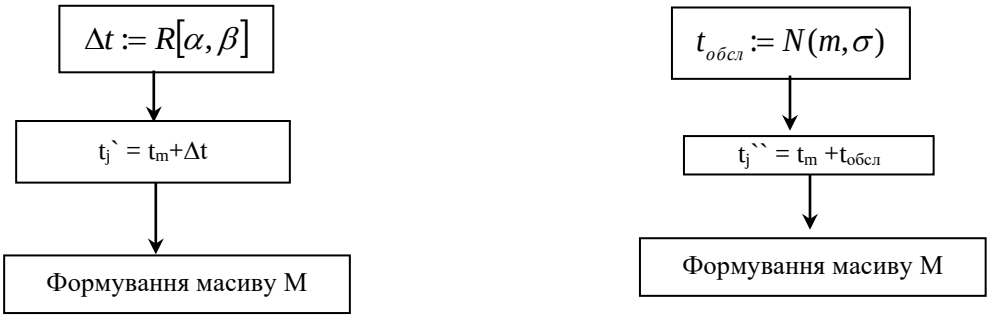
Необхідні статистичні характеристики - СМО обчислимо за формулою :

$$\bar{t}_{ож} = \frac{\sum_j t_{ожj}}{n}$$

Розглянемо зміст процедур формування основних подій:

$t_j`$(Пр.1) и $t_j``$ (Пр. 2).

Процедура 1 - формування $t_j`$ Процедура 2 - формування е $t_j``$



а) б)

Рисунок 3.1 – Процедуры формування основных подый

а) - $t_j`$ и б) - $t_j``$

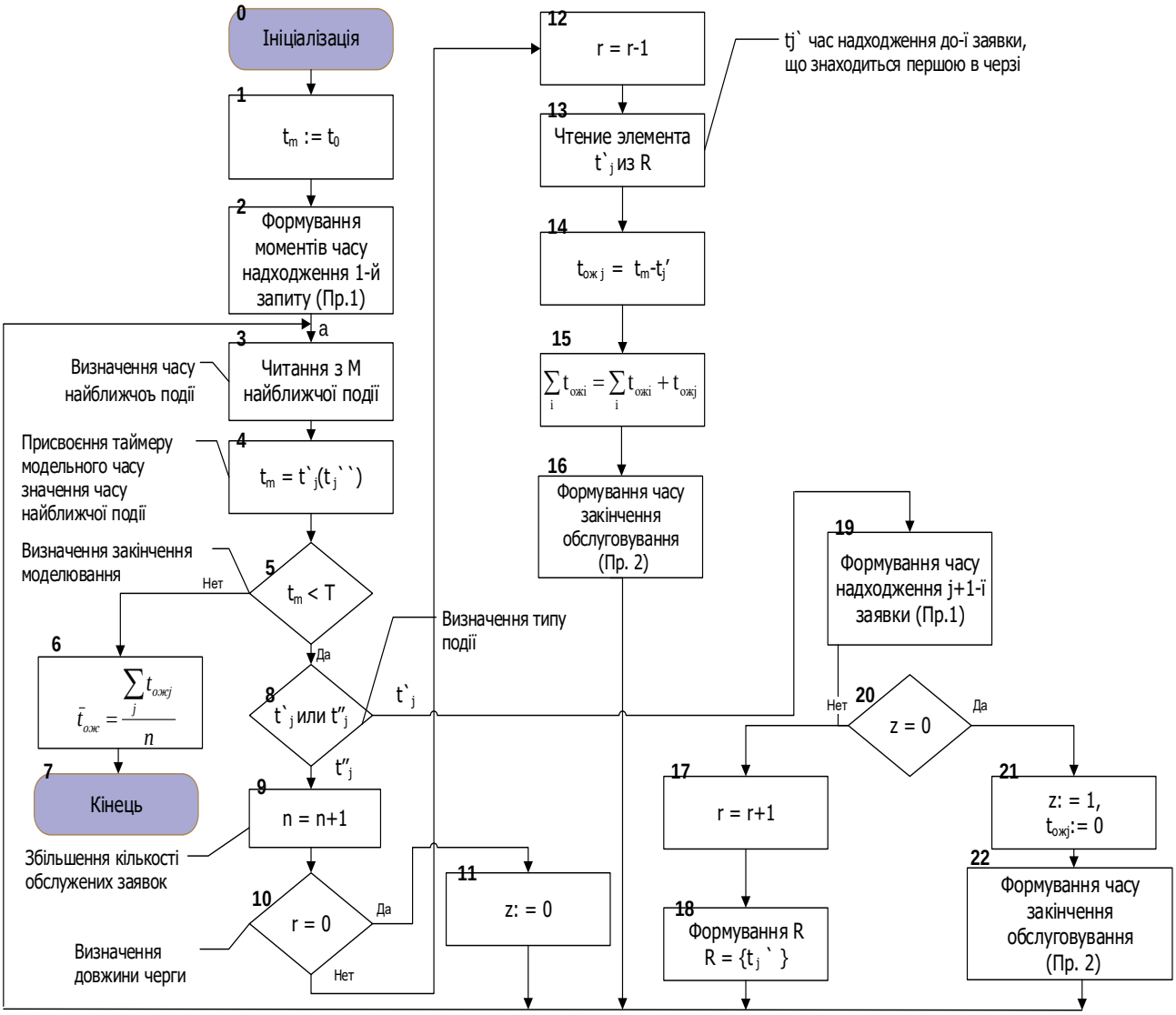


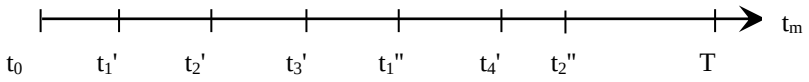
Рисунок 3.2 - Опис імітаційної моделі одноканальної СМО з необмеженою чергою: а) - геометрична інтерпретація списку подій; б) – стан списку подій М і R у моменти часу основних подій

Розглянемо в динаміці імітаційну модель, представлену на малюнку 3.2. Цей алгоритм будується виходячи з таблиці 3.1, а отже базується на подієво-орієнтованої концепції. Для його пояснення скористаємося малюнком 3.3. На цьому малюнку зображена тимчасова вісь t_m, на якій відзначатимемо всі основні події (ці події наведені в хронологічній послідовності в другій колонці таблиці 3.1), які відбуваються в моменти часу t0 = 0 і t_j', $t_j``$ (j = 1 ,2, ...).

Таблиця 3.1 - Подієво-орієнтований опис імітації роботи банківського касира

№ події	Час події, хв..	Номер клієнта	Тип події Список подій	Довжина черги	Число клієнтів	Стан касира	Час простою касира, хв.
	0,0	-	Начало	0	0	Вільний	3,2
1	3,2	1	Приход **(t_1`)**	0	1	Зайнятий	-
2	7,0	1	Уход **(t_1``)**	0	0	Вільний	3,9
3	10,9	2	Приход **(t_2`)**	0	1	Зайнятий	
4	13,2	3	Приход **(t_3`)**	1	2	Зайнятий	
5	14,4	2	Уход **(t_2``)**	0	1	Зайнятий	
6	14,8	4	Приход **(t_4`)**	1	2	Зайнятий	
7	17,7	5	Приход **(t_5`)**	2	3	Зайнятий	
8	18,6	3	Уход **(t_3``)**	1	2	Зайнятий	
9	19,8	6	Приход **(t_6`)**	2	3	Зайнятий	
10	21,5	7	Приход **(t_7`)**	3	4	Зайнятий	
11	21,7	4	Уход **(t_4``)**	2	3	Зайнятий	
12	24,1	5	Уход **(t_5``)**	1	2	Зайнятий	
13	26,3	8	Приход **(t_8`)**	2	3	Зайнятий	
14	28,4	6	Уход **(t_6``)**	1	2	Зайнятий	
15	31,1	7	Уход **(t_7``)**	0	1	Зайнятий	
16	32,1	9	Приход **(t_9`)**	1	2	Зайнятий	
17	33,2	8	Уход **(t_8``)**	0	1	Зайнятий	
18	35,7	9	Уход **(t_9``)**	0	0	Вільний	0,9
19	36,6	10	Приход **(t_{10}`)**	0	1	Зайнятий	
20	40,0	10	Уход **(t_{10}`)**	0	0	Вільний	

Лістинг 3.1 - Опис імітаційної моделі одноканальної СМО з необмеженою чергою: а) - геометрична інтерпретація списку подій; б) – стан списку подій M і R у моменти часу основних подій.



а)

Крок 1. $t_m=t_0$

$$M=\left\{t_1'\right\}$$

Крок 2. $t_m=t_1'$. Ставим на виконання 1-у заявку.

$$M=\left\{t_2',\quad t_1''\right\}$$

Крок 3. $t_m=t_2'$

$$M=\left\{t_3',\quad t_1''\right\}$$
$$R=\left\{t_2'\right\}$$ Ставим в чергу 2у- заявку.

Крок 4. $t_m=t_3'$

$$M=\left\{t_1'',\quad t_4'\right\}$$
$$R=\left\{t_2',\quad t_3'\right\}$$

Крок 5. $t_m=t_1''$ (Закінчення обслуговування 1-ї заявки. Ставимо на обслуговування 2-у заявку)

$$M=\left\{t_4',\quad t_2''\right\}$$
$$R=\left\{t_3'\right\}$$

б)

Покрокова процедура інтерпретація списку подій.

Крок 1. Ініціалізація (t_0). Моделювання починається з виклику основної програми ініціалізації (Бл.0). Змінні стани моделі встановлюються у вихідну позицію: прапор стану приладу обслуговування z = 0, число заявок у черзі r = 0, обнулюється статистичний лічильник $\sum t_{ож}$, а також масиви M і R.

Годинник модельного часу встановлюється 0, тобто. $t_m = t_0 = 0$ (Бл. 1).

За допомогою процедури Пр.1 формується час приходу першої заявки (першої події) $t_1`$, яке міститься в список подій: M = {$t_1`$} (Бл. 2).

Крок 2. Визначення наступної події та реакції системи на поточну подію. Спеціальна програма вибирає для обробки зі списку подій M подію з найменшим значенням часу (Бл. 3). Для розглянутого прикладу це $t_1`$.

Таймер модельного часу встановлюються на цю точку, тобто. $t_m = t_1`$ (Бл. 4).

У логічній вершині алгоритму «визначення закінчення моделювання» встановлюється факт, що $t_m < T$ (Бл. 5).

У логічній вершині алгоритму «визначення типу події» встановлюється факт, що перша подія - це час приходу першого клієнта $t_1`$(Бл. 8).

Відповідно до таблиці 3.1 алгоритм відображає наступну послідовність дій:

1) формування часу надходження наступного j+1-го клієнта (Пр.1) (Бл. 19). Припустимо, що другий клієнт надійде в банк у момент часу $t_2`$ (лістинг 3.1а) ;

2) визначення стану обслуговуючого приладу (Бл. 20) ;

3) стан пристрою обслуговування встановлюється значення 1 (Z = 1) (Бл. 21), що вказує а незайнятість пристрою, тоді як черга все ще порожня;

4) формування часу закінчення обслуговування першого клієнта (Пр. 2) (Бл. 22). Припустимо, перший клієнт буде обслужений у час t_1", як це відбито малюнку. Тут же показано стан масиву М;

5) повернення до точки алгоритму.

Крок 3. Визначення наступної події та реакції системи на поточну подію. Спеціальна програма переглядає список подій М і визначає за найменшим значенням, що наступною подією стане надходження ще одного клієнта в момент часу t2 `(Бл. 3).

Таймер модельного часу встановлюються на цю точку, тобто. $t_m = t_2$` (Бл. 4).

У логічній вершині алгоритму «визначення закінчення моделювання» встановлюється факт, що $t_m < T$ (Бл. 5).

У логічній вершині алгоритму «визначення типу події» встановлюється факт, що подія, що розглядається, - це час приходу другого клієнта t2 `(Бл. 8). Відповідно до рисунку 3.1 алгоритм відображає таку послідовність дій:

1) формування часу надходження наступного j+1 (третього) клієнта (Пр.1) (Бл. 19). Припустимо, що третій клієнт надійде до банку на момент часу t_3` (рисунок 3.1 а));

2) визначення стану обслуговуючого приладу (Бл. 20). Пристрій обслуговування залишається зараз у стані зайнято (відбувається обслуговування першого клієнта);

3) число клієнтів у черзі збільшується на 1, r = r + 1 0визначення стану обслуговуючого приладу (Бл. 17) ;

4) заповнення масиву R 0визначення стану обслуговуючого приладу (бл. 18) ;

5) повернення до точки а алгоритму.

Крок 4. Визначення наступної події та реакції системи на поточну подію. Спеціальна програма переглядає список подій М і визначає за найменшим значенням, що наступною подією стане надходження ще одного клієнта в момент часу t3` 0визначення стану обслуговуючого приладу (Бл. 3).

Таймер модельного часу встановлюються на цю точку, тобто. $t_m = t_3$`(Бл. 4). У логічній вершині алгоритму «визначення закінчення моделювання» встановлюється факт, що $t_m < T$ (Бл. 5).

У логічній вершині алгоритму «визначення типу події» встановлюється факт, що подія, що розглядається, - це час приходу третього клієнта t_3` (Бл. 8). Відповідно до рисунку 3.2 алгоритм відображає таку послідовність дій:

1) формування часу надходження наступного j+1 (четвертого) клієнта (Пр.1) (Бл. 19). Припустимо, що четвертий клієнт надійде до банку на момент часу t4` (див. рис. 7.6 а)). Нехай t4 `> t1";

2) визначення стану обслуговуючого приладу (бл. 20). Пристрій обслуговування залишається зайнятий (відбувається обслуговування першого клієнта);

3) число клієнтів у черзі збільшується на 1, r = 2 визначення стану обслуговуючого приладу (Бл. 17) ;

4) повернення до точки алгоритму.

Крок 5. Визначення наступної події та реакції системи на поточну подію. Спеціальна програма переглядає список подій М і визначає за найменшим значенням, що наступною подією стане час закінчення обслуговування першого клієнта t_1`` визначення стану обслуговуючого приладу (Бл. 3).

Таймер модельного часу встановлюються на цю точку, тобто. $t_m = t_1$`` визначення стану обслуговуючого приладу (Бл. 4).

У логічній вершині алгоритму «визначення закінчення моделювання» встановлюється факт, що $t_m < T$ (Бл. 5).

У логічній вершині алгоритму «визначення типу події» встановлюється факт, що подія, що розглядається, - це час закінчення обслуговування першого клієнта t_1``(Бл. 8). Відповідно до таблиці 7.4 алгоритм відображає таку послідовність дій, пов'язану з відходом заявки:

1) збільшується на 1 кількість обслужених заявок (Бл. 9) ;

2) визначення стану черги приладу (Бл. 10). Пристрій продовжує залишатися зайнятим, оскільки заявка 2 надходить обслуговування;

3) черга зменшується на одиницю (Бл. 12), а заявка 3 тепер є першою в черзі;

4) обчислення часу очікування у черзі другої заявки. В даному випадку ми скористаємося масивом R упорядкованих часів надходження заявок, які ставляться в чергу на момент часу t_m (Бл. 13) ;

5) обчислимо час очікування заявки 2 у черзі наступним чином: поточний час (t_m) мінус час надходження другої заявки (t_2`), що зберігається в масиві R (Бл. 14) ;

6) обчислення;

7) блок 15;

8) обчислюємо час очікування

$$\sum_i t_{ож} = \sum_i t_{ож} + t_{ожj} \ ;$$

9) формування часу закінчення обслуговування другого клієнта (Пр. 2) (Бл. 16). Припустимо, що друга заявка клієнт буде обслужений в момент часу t_2, як це відображає лістинг 3.1а. Тут же показано стан масиву М;

10) повернення до точки алгоритму.

Виконуючи подібним чином наступні кроки аналізованого алгоритму, прийдемо до ситуації, коли в логічній вершині алгоритму «визначення закінчення моделювання» (Бл. 5) буде встановлено той факт, що $t_m > T$. Це призведе до переходу до операторної вершини (Бл 6), де буде обчислено середній час очікування заявки в черзі:

$$\bar{t}_{ож} = \frac{\sum_j t_{ожj}}{n} \ .$$

Після цього роботу алгоритму буде завершено.

Відповідно, імітаційне моделювання - це подання динаміки імітаційної моделі за допомогою її просування від одного стану до іншого за заданими операційними правилами.

3.2 Проблеми формалізації багатокомпонентних імітаційних моделей

Імітаційне моделювання нині є найефективнішим методом оцінки показників процесу функціонування складних систем. Нагадаємо, що імітаційна модель є логіко-математичним описом об'єкта дослідження, який відтворює як структуру досліджуваної системи, залежності, що існують між її параметрами, так і процеси функціонування системи в часі. Найчастіше імітаційна модель виражається як алгоритму (рисунок 7.5). В імітаційній (алгоритмічній) моделі або моделювальному алгоритмі внутрішні операції і внутрішня структура можуть досить сильно збігатися з тими, які існують в реальній системі. Для порівняння, при аналітичному моделюванні оцінки показників ефективності моделюється системи обчислюються безпосередньо з математичної моделі (1.3) за допомогою різних чисельних методів.

Основна увага в даному розділі буде приділена основним проблемам, які виникають при розробці імітаційних моделей складних багатокомпонентних систем. Для їх опису доцільно розглянути базові поняття, що конкретизують динаміку моделей досліджуваних систем. Як згадувалося неодноразово раніше, ми вважаємо, що систему можна описати у межах понять " стан " і " перехід зі стану на стан " під впливом відповідних подій.

Стан системи визначається сукупністю змінних, необхідних для опису системи на певний момент часу. З формальної точки зору, під станом системи будемо розуміти комбінацію значень параметрів, (рисунок 1.2). Тоді стану системи можна подати у вигляді вектора:

$$\bar{S}' = (S_1', S_2', ..., S_k'),\ \bar{S}'' = (S_1'', S_2'', ..., S_k'')\ ,$$

і т.д., де $S_i' = S_i(t'), S_i'' = S_i(t'')$ - значення деякого узагальненого і-го параметра в моменти часу, відповідно, t' і t''.

Сукупність всіх можливих значень станів $\{S_i'\}$ називається простором станів об'єкта моделювання. Якщо розглядати процес функціонування системи як послідовну схему станів, то вони можуть бути інтерпретовані як координати точки в k-мірному фазовому просторі. Причому кожної реалізації процесу відповідатиме деяка фазова траєкторія.

Відповідно, імітаційне моделювання - це подання динаміки імітаційної моделі за допомогою її просування від одного стану до іншого за заданими операційними правилами. Як ми переконалися із попереднього розділу, ці правила мають алгоритмічну основу.

Для опису динаміки імітаційної моделі необхідно скористатися специфікаціями, які пов'язують характеристики станів системи $S_i(t)$ з її параметрами та часом, а також початковими умовами S_i^0 початкового моменту часу t_0.

Згадані операційні правила перетворюються на такий вид, щоб, наскільки можна, зробити зручним визначення $S_i(t_0 + \Delta t)$. У моделі виділяється певний ідентифікатор для фіксації поточного значення t_m (модельний час). У початковий момент модельний час дорівнює t_0. До t_0 додається величина Δt та обчислюється момент часу $t_1 = t_0 + \Delta t$ (на осі t_m) зміни стану системи, а також самі стани $S_i(t_1)$; потім знову визначається час $t_2 = t_1 + \Delta t$ (на осі t_m) зміни станів тощо.

Моменти часу t_0, t_1, t_2 ... на осі t_m називаються особливими. Зміна станів системи, що у ці моменти часу, повністю визначає динаміку імітаційної моделі. Тому при комп'ютерній реалізації достатньо відтворювати лише ці зміни. Необхідно відзначити, що зміни в особливі моменти часу можуть бути випадковими. Тому структура імітаційної моделі повинна забезпечувати обчислення розподілу ймовірностей і-го параметра $S_i(t)$, $i = \overline{1,n}$.

Розрізняють два типи імітаційних моделей, що пов'язано з такими обставинами.

У моделях одного типу переходи системи з одного стану до іншого, ініціюється реальними подіями, що відбуваються в системі. Тоді послідовність зазначених подій становить істоту функціонування системи. У цьому випадку динаміка моделі фактично повторює динаміку системи, тобто перехід від однієї події до іншої. Кажуть, що відповідна алгоритмічна модель має подійний характер.

У моделях іншого типу події вводяться штучно внаслідок необхідності представити деякий безперервний процес дискретної ЕОМ. Це притаманно випадку, коли імітаційна модель є алгоритмічну реалізацію аналітичної моделі з її рішення на ЕОМ. Прикладом є будь-який чисельний метод інтегрування диференціальних рівнянь. У таких методах моменти настання подій визначаються кроком інтегрування. Крок інтегрування або призначається постійним, або, якщо це необхідно, змінюється відповідно до спеціально встановленого механізму. Тому методи цього називають покроковими. При їх використанні динаміка моделі є дискретним наближенням реальних безперервних процесів.

Зупинимося однією важливою особливості зазначеного алгоритмічного уявлення. Внаслідок послідовного характеру обробки інформації в обчислювальній однопроцесорній машині паралельні процеси, що відбуваються в моделі, перетворюються за допомогою обговорюваного алгоритму на послідовні. Такий спосіб подання зветься квазіпаралельного процесу. Протиріччя між паралельністю модельних процесів і послідовним характером квазіпаралельного процесу є причиною їх неповної відповідності.

Узагальним прийоми побудови імітаційної моделі однокомпонентної системи у разі багатокомпонентної системи. Розглянемо два приклади.

Приклад 1. Розглянемо структурну модель багатокомпонентної системи, яка є мережею СМО (рисунок 3.3).

Якщо припустити, що події у вхідних потоках з інтенсивностями λ_1 і λ_2 з'являтимуться одночасно, то і прилади 1 і 2, що обслуговують, будуть активуватися також одночасно, тобто. всередині цих приладів протікатимуть паралельно відповідні внутрішні процеси. Обслуговуючий прилад Σ_3 активуватиметься, як тільки на виходах Σ_1 або Σ_2 з'являться будь-які події. Когда в Σ_3 протікатиме внутрішній процес, може виявитися, що Σ_1 і (або) Σ_2 протікатимуть відповідні внутрішні процеси, викликані подіями вхідних потоків. Таким чином, внутрішні процеси у всіх трьох приладах можуть проходити одночасно (тобто паралельно). Для ілюстрації такого паралелізму скористаємось додатково малюнками 3.4 та 3.5. На малюнку 3.3 тимчасова вісь t_i – являє собою локальний час для і-ої компоненти, а тимчасова вісь t_m – є глобальним модельним часом системи. За допомогою t_i можна будувати ІМ для і-ої компоненти незалежно від інших ІМ компонентів.

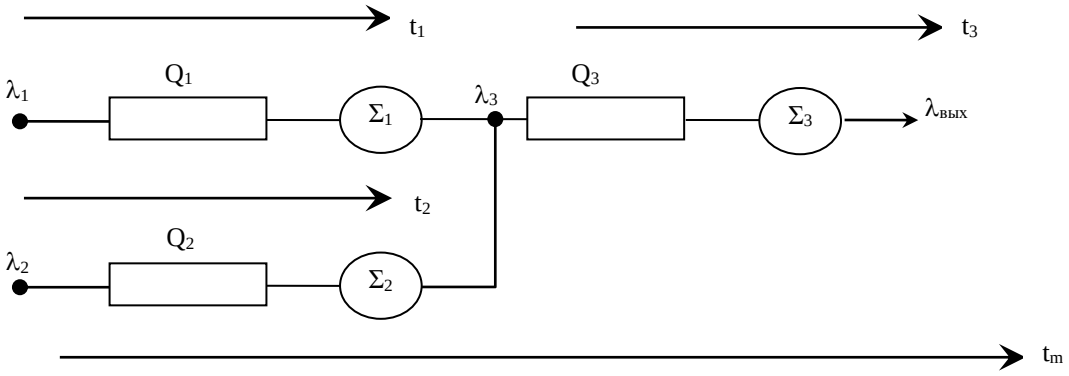


Рисунок 3.3 – Багатокомпонентна СМО

При побудові ІМ всієї системи необхідно спроектувати всі локальні осі t_{mi} на вісь глобального модельного часу t_m.

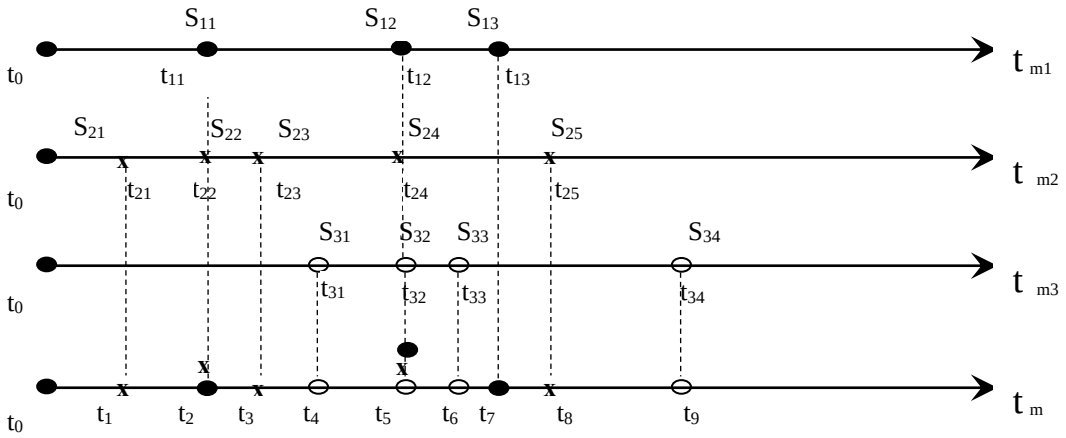


Рисунок 3.4 – Локальні та глобальні таймери модельного часу

На рисунку 3.4 моменти часу основних подій, що виникають у трьох компонентах, позначені відповідно ●, **x**, o. З цього малюнка видно, що в момент глобального часу t_2 відбуваються одночасні події в моменти часу t_{11} і t_{22}, відповідно в першій і другій компонентах, а в момент глобального часу t_5 відбуваються одночасні події в моменти часу t_{12}, t_{24} і t_{32} відповідно в першій, другий та третій компоненти.

Приклад 2. Як зазначалося вище, моделювання елементарних паралельно виконуваних дій в електронній схемі можна здійснити шляхом застосування операторів призначення VHDL сигналів. Однак елементарних призначень явно недостатньо для того, щоб описувати паралельні процеси, що протікають у таких складних електронних системах, як ІС. Очевидно, що для моделювання складних пристроїв необхідний механізм, що дозволяє будувати складніші блоки та імітувати їхнє паралельне виконання. Ілюстрацією цієї ситуації може бути схема, наведена на рисунку 3.5.

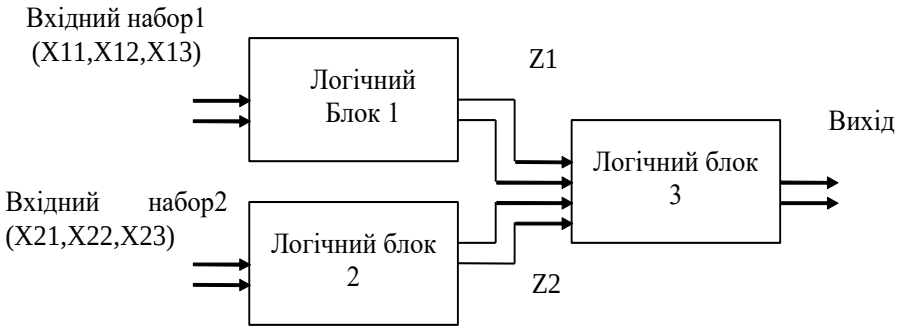


Рисунок 3.5 - Структура устройства с логическими блоками

Якщо припустити, що вхідний набір сигналів 1 і 2 активуватимуться одночасно, то логічні блоки 1 і 2 активуватимуться також одночасно. Логічний блок 3 буде активуватися, як тільки станеться зміна значень сигналів Z1 або Z2. Коли ці сигнали проходять через блок 3 через блоки 1 і 2 можуть проходити нові змінені значення вхідних сигналів. Таким чином, потік сигналів може проходити через блоки одночасно (тобто паралельно). Така одночасність реалізується з допомогою механізму процесів.

На однопроцесорній машині зазвичай немає можливості дійсно паралельно виконувати всі процеси, описані у програмній моделі. Реально програма моделювання всі процеси виконує послідовно. Ефект паралельності досягається лише за рахунок використання глобальних сигналів та процедури формування модельного часу. Це означає, що кожен процес де-небудь у своєму виконуваному розділі містить оператори призначення глобальних сигналів, наприклад, Z1 або Z2 (тобто сигналів, описаних у зовнішньому по відношенню до процесу блоці). Змінюючи значення цих глобальних сигналів, процес передає результат свого виконання іншим процесам системи, наприклад, в блок 3. Така передача здійснюється з використанням драйверів сигналів, які відіграють роль таймерів модельного часу для сигналів.

Розглянемо VHDL – модель пристрою, що розглядається на рисунку 3.5.

Лістинг 3.1– VHDL – модель пристрою з логічними блоками

```
L_B_1 : process (X11, X12, X13)
                    Variable YINT :  Bit;
                  begin
                     YINT := X11  and  X12;
                      Z1 <= YINT or X13 after 35 ns;
                  end process L_B_1;
L_B_2 : process (X21, X22, X23)
                    Variable YINT23 :  Bit;
                  begin
                     YINT23 := X21  or  X22;
                      Z2 <= YINT23 or X23 and X22 after 30 ns;
                  end process L_B_2;
L_B_3 : process (Z1, Z2)
                     Begin
                     тіло процесса
                      end process L_B_3;
```

Процес L_B_1 активізується, якщо змінюється хоча б один із сигналів вхідного набору 1.

Процес L_B_2 активізується, якщо змінюється хоча б один із сигналів вхідного набору 2.

У результаті виконання L_B_1 змінює глобальний сигнал Z1. При виконанні оператора призначення сигналу Z1 відводиться драйвер, відповідно до якого обчислене значення Z1 встановиться через 35 наносекунд.

У результаті виконання L_B_2 змінює глобальний сигнал Z2. При виконанні оператора призначення сигналу Z2 відводиться драйвер, відповідно до якого обчислене значення Z2 встановиться через 30 наносекунд.

Зміна Z1 та Z2 активізуватиме процес L_B_3.

3.3 Висновки:

1) одним з ключових елементів динаміки моделювання є взаємодія між годинниками модельного часу t_m і списком подій М. Годинник перекладається на час наступної події, яка визначається за допомогою сканування списку

подій в кінці обробки кожної події та знаходження в ньому події з найменшим часом . Таким чином, здійснюється моделювання у часі;

2) поки обробляється чергова подія, модельний час не змінюється;

3) у деяких імітаційних моделях може виявитися, що два (і більше) елементи у списку подій пов'язані з найменшим часом. Йдеться про одночасні події, які мають відбуватися в один і той же модельний час, наприклад події закінчення обслуговування заявки та надходження нової. У таких ситуаціях необхідно використати вирішальне правило. Вирішальне правило може вплинути на результати моделювання, тому його слід вибирати відповідно до того, як система моделюється. Розглянемо два приклади вирішального правила;

4) у першому випадку, коли кілька елементів у списку подій М пов'язані з найменшим часом, без зміни модельного часу tт послідовно обробляються всі такі події. Після цього програма, яка переглядає список подій та визначає наступне найменше значення, оновлює годинник відповідно до цього значення;

5) у другому випадку вирішальне правило використовує спеціальний механізм (алгоритм), який забезпечує нульову ймовірність появи декількох елементів у списку подій М пов'язаних з найменшим часом;

6) наведений вище приклад демонструє які компоненти імітаційної системи та структури даних задіяні у дискретно-подійному моделюванні з погляду планування подій. У ньому містяться всі найважливіші ідеї, необхідні здійснення моделювання складних систем такого типу.

4 ЕКСПЕРИМЕНТАЛЬНА ОЦІНКА ПРОДУКТИВНОСТІ СЕРВЕРА БД МЕТОДОМ ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ

4.1 Основні характеристики і специфікація імітаційних моделей

Цей розділ дає вступ до проблеми, для вирішення якої призначена наша мова допомогти розв’язати та представити різні конструкції, які підтримує мова.

Більшість рішень великих компаній включають аналіз витрат і вигод на основі оцінок різні параметри [4], наприклад, наскільки клієнти готові купувати певний продукт. Але оцінка цих параметрів може вимагати дуже складних міркувань і значний досвід роботи з конкретною областю, щоб бути коректним [5].

Наприклад, постачальник транспортних послуг може захотіти з’ясувати, чи а зниження ціни на квитки збільшило б або зменшило його дохід. Експериментує безпосередня зміна цін на квитки може бути дорогою важко, оскільки це може завдати шкоди репутації постачальника послуг. Замість цього можна спробувати змоделювати поведінку клієнтів. Сервіс постачальник може припустити, що сім’ї з низьким рівнем доходу подорожують автобусом, якщо це так дешевше, ніж подорожувати автомобілем, тоді як сім'ї з високим рівнем доходу будуть їздити самостійно автобус якщо так зручніше не дивлячись ні на що. Тоді постачальник послуг зможе знайти кількість малозабезпечених і багатозабезпечених сімей і вивчити ціни на бензин і середню тривалість подорожі для розрахунку оцінки.

Запровадження моделі зміщує тягар визначення параметрів для само рішення до визначення параметрів моделі, які в деяких ситуаціях може бути набагато простіше і дешевше. Це також створює проблему забезпечення цього однак модель дійсна сама по собі.

Отже, як можна вказати модель? Мета цього проекту – допомогти цьому. Один підхід полягає в тому, щоб виразити зв’язки між об’єктами в математичних термінах функції. Наприклад, постачальник транспортних послуг може припустити від якого частка сімей з низьким доходом, які подорожують автобусом, залежить лінійно ціна квитка між двома кінцевими пунктами, кожен, хто подорожує автобусом, і кожен подорож на машині.

Математичний опис поки що досить хороший, але специфікація а Модель є лише частиною процесу. Інша частина полягає в отриманні даних із модель. Якщо обсяг даних для отримання великий, це отримання краще виконувати автоматично що вимагає, щоб модель була специфікована строго формальною мовою. Потім модель можна запустити через симулятор, який обробляє опис і виводить необхідні дані.

Існують принципово різні варіанти побудови моделі. Один може виберіть підхід безперервного часу або модель часу з дискретними подіями, щоб кожна стан обчислюється з попереднього стану. А при моделюванні багатьох подібних об’єктів, можна або фактично побудувати всі ці об’єкти, або імітувати їх окремо поведінку, або створити лише один і дозволити йому демонструвати середню поведінку. Найкращий вибір неочевидний і може залежати від ситуації, але ми вибрали підтримка дискретного моделювання кількох об'єктів.

Важливою перевагою формулювання моделей як дискретних подій є те, що вони дає можливість безпосередньо висловити залежність від минулих подій. Наприклад, с модель транспорту, якщо хтось із родини позичив машину на пару днів, автомобіль недоступний, що впливає на процес прийняття рішення. З дискретним ми можемо легко відслідковувати такі ситуації, тоді як безперервна модель доведеться потурбуватися про те, як вони впливають на середню поведінку випадку. Також дозволяє використовувати стільки об’єктів, скільки вже пройшли моделювання, а не лише середні об’єкти один фокус на моделюванні конкретного об'єкта інтересу, розщепленні індивідуальних відмінностей у різні випадки, замість того, щоб думати про все

одразу, щоб охопити середня поведінка. Більш складний аналіз даних, отриманих за допомогою модель також може стосуватися не лише середніх значень, і в цьому випадку ми повинні мати дані з окремих об'єктів.

4.2 Специфікація клієнт-серверної системи на базі сервера БД

Система «клієнт-сервер» складається з 20 терміналів, підключених до локальної мережі, і сервера, який обслуговує загальну базу даних для обробки запитів клієнта.. Середній час обмірковування дорівнює 18 сек. Кожна заявка від терміналу генерує в середньому 20 запитів на диск. Коефіцієнт використання диска дорівнює 0.5(50%). Середній час обслуговування запиту диском дорівнює 20 мілісекунд.

Вихідні дані:

1) коефіцієнт використання диска U = 0.5(50%);

2) коефіцієнт відвідування диска запитами

$$V_i^{'} = X_i / X_0 = 20 \text{ запитів;}$$

3) середній час обслуговування запиту на диск $S_i = 1/\mu_D$ =20ms;

4) число терміналів дорівнює M = 20;

5) середній час обмірковування терміналом дорівнює $Z=1/\lambda_T$ =18с.

Завдання:

1) побудуємо структурну схему розробленої системи «клієнт-сервер» ;

2) оцінити середнє число клієнтів $\bar{r}$ і середній час обслуговування клієнта $\bar{t}_{\text{syst}}$ в залежності від навантаження ρ з використанням аналітичного і імітаційного моделювання. Використати імітаційну систему GPSS.

4.3 Експериментальна оцінка продуктивності сервера БД

Побудуємо структурну схему розробленої системи «клієнт-сервер», використовуючи один сервер для бази даних (рис. 4.1).

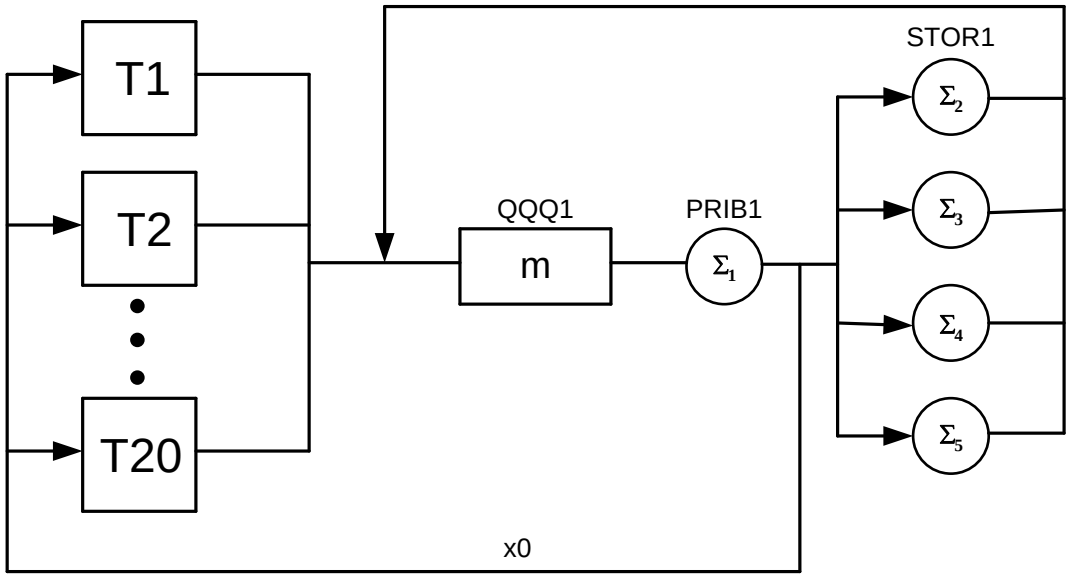


Рисунок 4.1 - Структурна схема системи «клієнт-сервер»

Таблиця 4.1 – Результат $\bar{r}(\rho)$

ρ / $\bar{r}$	0,2	0,4	0,66	0,8
Аналітична модель	0,05	0,267	1,33	3,2
Імітаційна модель (GPSS)	0,052	0,246	1,33	3,4

Таблиця 4.2 – Результат $\bar{t}_{syst}(\rho)$

ρ / $\bar{t}_{сист}$	0,2	0,4	0,66	0,8
Аналітична модель	6,25	4,76	60	83
Імітаційна модель (GPSS)	6,34	4,629	60	84,5

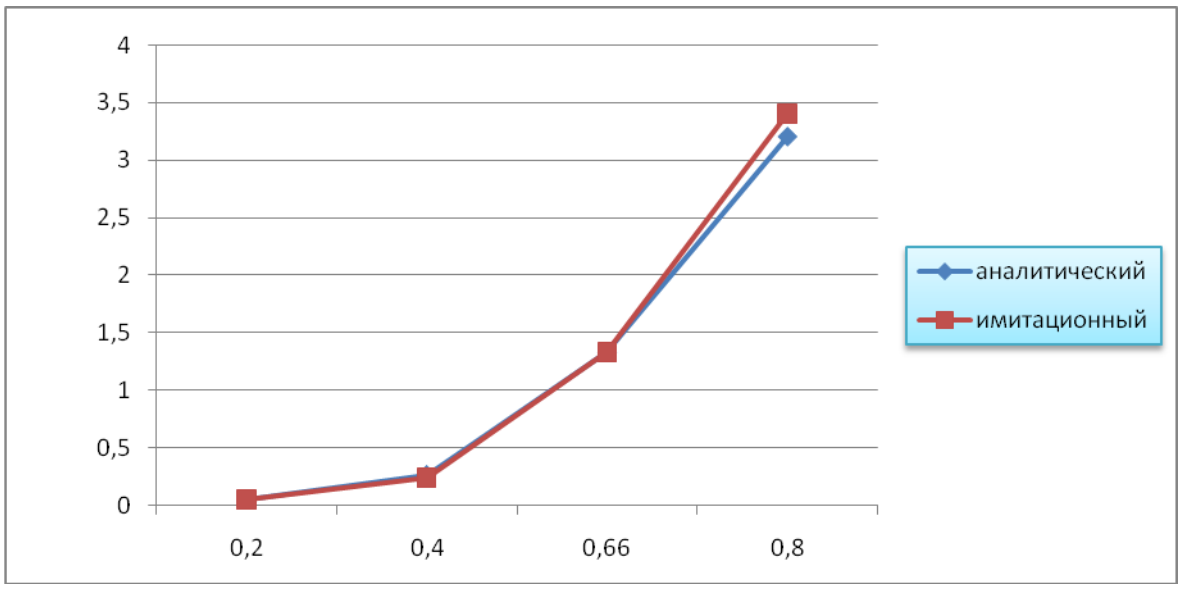


Рисунок 4.2 – Порівняльні графіки

Лістинг програми та звіт імітаційної системи GPSS наведено у додатку Б.

ВИСНОВКИ

Оперативна теорія мережі масового обслуговування базується на передумові можливості тестування. Усі основні величини продуктивності (Таблиця II) – використання, коефіцієнти завершення, середні розміри черги, середній час відповіді, розподіл навантаження – визначаються так, як вони були б на практиці з даних, отриманих за кінцевий період. Аналітик може перевірити, чи справедливі основні припущення – баланс потоку, однокрокова поведінка та однорідність – у будь-якому періоді спостереження.

Операційні закони (табл. I і III) є тотожними серед операційних величин. Це перевірка узгодженості – невиконання операційного закону виявляє помилку в даних. Вони спрощують збір даних, показуючи альтернативи для обчислення величин продуктивності.

Баланс потоку завдань передбачає, що пропускна здатність скрізь у системі визначається пропускною здатністю в будь-якій точці системи. Оскільки збільшення навантаження призведе до насичення деяких пристроїв, це припущення дозволяє визначити асимптоти пропускної здатності та часу відгуку; Єдиними даними, необхідними для такого «аналізу вузьких місць», є коефіцієнти відвідувань і вихідні показники насичення на пристроях.

Аналіз потоку завдань не враховує ефекти черги в системі при проміжних навантаженнях, які необхідно вивчати з точки зору простору станів системи. Кожен стан $n = (n_l, \ldots, n_x)$ представляє можливий розподіл завдань між пристроями, а $p(n)$ представляє частку часу, протягом якого стан n зайнятий. Мета полягає в тому, щоб виразити $p(n)$ безпосередньо через робочі параметри системи.

За додаткових припущень щодо однокрокової поведінки та однорідності ми можемо знайти рівняння балансу, що пов’язують $p(n)$ із коефіцієнтами операційного відвідування та функціями часу обслуговування. Це, мабуть, найслабкіші припущення, що призводять до розв’язку у формі продукту для $p(n)$. Використовуючи форму продукту рішення, ми можемо розробити

ефективні методи для обчислення величин продуктивності без необхідності явного обчислення p(n). Дійсно, надзвичайна швидкість, з якою величини продуктивності можуть бути обчислені за допомогою формул мережі масового обслуговування, є важливою причиною такого широкого використання цієї технології.

Більшість помилок із цими результатами виникають через припущення щодо однорідності. Однорідність стверджує, що між пристроєм і рештою системи немає взаємодії, за винятком залежності від довжини локальної черги. У реальній системі функція обслуговування є залежною від шаблону, за яким решта системи надсилає запити до пристрою, і цей шаблон може залежати від форми розподілу розміру запиту цього пристрою.

На практиці помилки з цих припущень не є серйозними. Навіть якщо для подальшого спрощення аналізу використовується додаткове припущення про однорідний час обслуговування, ці моделі оцінюють використання, пропускну спроможність і час відповіді системи, як правило, з точністю до 10%, а середню довжину черги та час відповіді пристрою зазвичай з точністю до 30%. Удосконалення моделі пристроїв, щоб зробити явним ефект розподілу розміру запиту, підвищує точність, особливо при прогнозуванні розподілу довжини черги. Дуже мало відомо про розподіл часу відгуку для цих систем.

Щоб використовувати ці результати для прогнозування ефективності, аналітик повинен оцінити значення параметрів для прогнозованого періоду; потім використовуйте ці оцінки в рівняннях для розрахунку оціночних показників ефективності в прогнозному періоді. Ми не запропонували остаточного вирішення проблеми оцінки параметрів. Ми також не можемо: це в царині індуктивної математики, тоді як операційний аналіз є розділом дедуктивної математики. У прикладах ми проілюстрували види припущень про інваріантність, які аналітики використовують для оцінки параметрів.

Стохастична теорія масового обслуговування робить деяких аналітиків зручнішими при оцінці параметрів, оскільки теорія розповідає, як вивести довірчі інтервали, щоб обмежити невизначеність в оцінках, отриманих на

основі даних, отриманих за кінцевий базовий період. Однак стохастична модель використовує приховане індуктивне припущення: значення стохастичних параметрів у період прогнозування є відомими функціями відповідних значень у базовому періоді. Насправді немає способу дізнатися це напевно. Таким чином, стохастичний аналітик стикається з точно такою ж невизначеністю, як і операційний аналітик; обидва повинні оцінити невідомі значення для прогнозованого періоду на основі значень, що спостерігаються в базовому періоді. Робота з невизначеністю в оцінці є дуже важливою проблемою, але вона виходить за межі дедуктивної математичної системи, в якій виводяться співвідношення між змінними. (Для повного обговорення цих моментів.

Маючи слабку основу, операційна теорія мережі масового обслуговування застосовується до ширшого класу комп’ютерних систем, ніж марковська теорія мережі масового обслуговування. Навпаки, теорія Маркова включає припущення, яких немає в операційній структурі цієї статті. Марківська теорія мережі масового обслуговування, наприклад, дозволяє отримати диференціальні рівняння, що пов’язують залежну від часу ймовірність

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Технології моделювання систем. Навчальний посібник. Міністерство освіти і науки України Друк. Харків: СМІТ, 2005. – 175с. ISBN № 966-8530-26-6
2. Chen H., A. Mandelbaum, “Stochastic discrete flow networks: diffusion approximations and bottlenecks,”, Graduate School of Business, Stanford University, 1988.
3. Bukchin J., “A comparative study of performance measures for throughput of a mixed model assembly line in a JIT environment”, International Journal of Production Research, Vol.36, No.10, pp 2669-2685, 1998.
4. Bolch G., S. Greiner, et. al., “Queueing networks and Markov chains: modeling and performance evaluation with computer science applications”, John Wiley and Sons, 1998.
5. Casale G., G. Serazzi, “Estimating Bottlenecks of Very Large Models”, Performance Evaluation Stories and Perspectives-G.Kotsis Editor, Austrian Computing Society, pp 89-104, 2003.
6. Casale G., G. Serazzi, “Bottlenecks Identification in Multiclass Queueing Networks using Convex Polytopes”, In Proc. IEEE/ACM MASCOTS 2004, IEEE Comp. Soc., pp 223–230, 2004.
7. Chiang S. Y., C. T. Kuo, et. al. “DT-bottlenecks in serial production lines:theory and application”, IEEE Transctions on Robotics and Automation, Vol.16, Issue 5 , pp 567-580, 2000.
8. Gunter Bolch, StefanGreiner, Hermann de Meer, Kishor S. Trivedi. Queueing Networks and Markov Chains. Second edition. Published by Jonh Wiley & Sons, Inc., 2006.
9. Клейнрок Л. Теория массового обслуживания /Пер. с англ. И. И. Глушко. – М.: Машиностроение, 1979. – 432 с.

10. Daniel A. Menascé, Virgilio A.F. Almeida, Lawrence W. Dowdy. Performance by Design: Computer Capacity Planning by Example. Prentice Hall PTR, ISBN : 0-13-090673-5, 2004, - 552 c.
11. Peter J. Denning, Jeffrey P. Buzen, The Operational Analysis of Queuing Network Models, Computing Surveys, Vol. 10, N 3, September 1978, pp.225-261.
12. Томашевський В.М. Моделювання систем. –К.: Видавнича група BHV, 2005. – 352с.: іл.
13. Молчанов А.А, Моделирование и проектирование сложных систем. – К. Высшая шк.., 1988. – 359 с.
14. Хинчин А.Я. Математические методы теории массового обслужи-вания // Труды математического института им. В.А. Стеклова. – М.: Изд. АН СССР. – Т. 48, 1955. – 122 с.
15. Бусленко Н.П. Моделирование сложных систем. – М.: Наука, 1978.-400с.
16. Banks J., Discrete-event system simulation. Prentice-Hall Inc., 1995.- 548 p.
17. Nain Ph., Basic Elements of Queuing Theory. Application to the Modeling of Computer Systems. Lecture Notes. INRIA. 1998.- 110 p.