

Лекція 1. Поняття і види моделювання

План

1. Поняття моделі.
2. Класифікація видів моделювання.
3. Наочне, символічне і реальне моделювання.
4. Поняття математичного моделювання.

1. Поняття моделі

У найзагальнішому формулюванні *модель* — це об'єкт, подібний у деяких відношеннях на прототип, який використовується для опису і (або) пояснення і (або) прогнозування поведінки прототипу.

Найважливішою властивістю моделі є те, що вона дає спрощений образ, який відображає не всі властивості об'єкта, а тільки ті, які є істотними для дослідника.

Модель і реальна система знаходяться між собою у відношенні гомоморфізму (однозначного відображення) $f : A \rightarrow B$, де A — модель, а B — реальна система. Оскільки модель є простішою за реальну систему, відповідність f не є взаємно-однозначною.

Із давніх-давен під час вивчення складних процесів чи явищ, конструювання нових споруд люди використовували моделі. Добра модель більш доступна для дослідження, ніж реальна система, вивчення моделі дає нові знання про об'єкт-оригінал.

Моделювання — це спосіб дослідження будь-яких об'єктів, процесів чи явищ шляхом побудови та аналізу їхніх моделей.

Типовими цілями моделювання можуть бути пошук оптимальних чи близьких до оптимальних розв'язків, оцінка ефективності роботи системи, визначення властивостей системи, прогнозування її розвитку.

Можна виділити три основні області використання моделей: навчання, наукові дослідження, керування. При навчанні моделі сприяють наочності відображення різних об'єктів і полегшенню передачі знань про них. У наукових дослідженнях моделі дають нові знання про досліджуваний об'єкт, процес чи явище. При керуванні моделі використовуються для прийняття рішень. Такі моделі мають забезпечувати як опис, так і прогнозування поведінки систем.

2. Класифікація видів моделювання

Єдина класифікація видів моделювання неможлива через багатозначність поняття моделі в науці, техніці, суспільстві. Ми розглянемо чотири способи класифікації.

1. Моделювання можна ділити на детерміноване і ймовірнісне. *Детерміноване* моделювання відображає процеси, у яких припускається відсутність випадкових впливів. *Ймовірнісне (стохастичне)* моделювання враховує ймовірнісні процеси і події.

2. Залежно від того, яким чином відтворюються в часі стани моделі, розрізняють *неперервне, дискретне* і *змішане (дискретно-неперервне)* моделювання. Неперервні моделі логічно використовувати для моделювання неперервних систем, а дискретні — для дискретних. Проте часто бувають випадки, коли дискретні системи успішно моделюють з допомогою неперервних моделей, а неперервні — з допомогою дискретних.

Об'єднуючи попередні класифікації, отримуємо неперервно-детерміноване, дискретно-детерміноване, неперервно-ймовірнісне і дискретно-ймовірнісне моделювання.

3. Моделювання буває статичним і динамічним. *Статичне* моделювання описує стан об'єкта у фіксований момент часу, а *динамічне* — зміну станів об'єкта в часі.

4. Моделювання ділять також на *уявне* і *реальне*. Уявне моделювання в свою чергу ділять на наочне, символічне і математичне, а реальне — на натурне і макетне¹.

¹Макетне моделювання також називають фізичним.

Уявне моделювання (його називають також *абстрактним*) є єдиним способом моделювання систем, які не можуть бути вивчені безпосередньо. Зокрема, нереально поставити експеримент по прямому дослідженню зірок чи планет. У цивілізованих країнах неприпустимо експериментувати з економікою чи здоров'ям людей. Часто безпосередні експерименти неможливі у зв'язку зі складністю, великими матеріальними затратами, унікальністю системи, тривалістю експерименту тощо. У таких випадках уявним моделям немає альтернативи.

3. Наочне, символічне і реальне моделювання

При *наочному* моделюванні на базі уявлень людини про реальні системи створюються наочні моделі, які відображають явища і процеси, що відбуваються в системі. Прикладами таких моделей є навчальні плакати, рисунки, графіки, схеми, діаграми, анімація. З допомогою спеціальних комп'ютерних програм можна створювати тривимірні моделі. До наочного моделювання також відносять створення макетів, які відображають окремі сторони об'єкта. Детальніші макети відносять до макетного моделювання.

Символічне моделювання є штучним процесом створення логічного об'єкта, що замінює реальний і виражає його основні властивості з допомогою певної системи слів чи знаків. Символічне моделювання можна поділити на мовне і знакове.

В основі *мовного* моделювання лежить деякий тезаурус. Під *тезаурусом* розуміють словник, який відображає зв'язки між словами чи іншими елементами даної мови, причому кожному слову у тезаурусі відповідає єдине поняття.

Якщо ввести умовні позначення окремих понять, тобто знаки, а також певні операції між цими знаками, то можна реалізувати *знакове* моделювання і з допомогою цих знаків складати ланцюжки слів і речень.

Поняття *математичного* моделювання розкривається в наступному пункті.

При *реальному* моделюванні досліджується реальний об'єкт, його частина або подібний до нього об'єкт. Реальне моделювання ділять на натурне і макетне.

Натурним моделюванням називають проведення дослідження на реальному об'єкті з наступною обробкою результатів. Методами натурального моделювання є науковий експеримент, комплексні випробування і виробничий експеримент.

Науковий експеримент характеризується використанням різноманітних засобів обробки інформації і можливістю втручання людини в процес проведення експерименту. В ході експерименту здійснюються збурювальні впливи на процес функціонування об'єкта, при цьому можуть виявитись окремі критичні ситуації і визначитись межі стійкості процесу. *Комплексні випробування* здійснюють шляхом багаторазових повторень випробувань об'єктів. У цьому випадку моделювання зводиться до обробки й узагальнення інформації про помічені закономірності. *Виробничий експеримент* полягає в узагальненні досвіду, накопиченого під час виробничого процесу.

Іншим видом реального моделювання є *макетне* моделювання, яке відрізняється від натурального тим, що моделювання здійснюється на об'єктах, які зберігають природу явищ і є подібними до реальних систем. Часто такими об'єктами є макети, які відтворюють модельовану систему в певному масштабі. Найпростіші макетні моделі є зменшеними копіями реальних систем. Наприклад, можна створити макет гідротехнічних споруд і досліджувати його в умовах, близьких до реальних. Інші приклади макетного моделювання — дослідження крила літака або лісу, утвореного зі штучних дерев, в аеродинамічній трубі, дослідження моделі корабля в басейні.

Макетне моделювання може відбуватись у реальному і модельному часі. Наприклад, якщо дослідження поведінки об'єкта в аеродинамічній трубі здійснюють у реальному часі, то моделювання поведінки гірських систем можливе лише в модельному часі.

4. Поняття математичного моделювання

Математичне моделювання — це процес створення для заданого реального об'єкта деякої математичної моделі. Нею може бути як система рівнянь, так і комп'ютерна програма. Кожна математична модель описує реальний об'єкт з деякою мірою наближення. Дослідження моделі дає можливість встановити характеристики реального об'єкта. Математичне моделювання є одним з основних способів моделювання систем.

Математичне моделювання ділять на аналітичне, імітаційне і комбіноване.

При *аналітичному* моделюванні глобальні рівняння системи, які описують її закон функціонування, записуються у вигляді деяких аналітичних співвідношень (алгебричних, диференціальних, інтегральних, інтегро-диференціальних та інших рівнянь чи їхніх систем) і деяких додаткових умов (початкових, крайових, багатоточкових умов чи деяких обмежень). У цьому випадку моделюється переважно функціональний аспект системи. Найчастіше зустрічаються диференціальні моделі, в яких реальні об'єкти описуються з допомогою звичайних диференціальних рівнянь або рівнянь з частинними похідними.

Аналітична модель досліджується кількома методами: аналітичними, числовими або якісними.

Аналітичні методи дозволяють отримати в загальному вигляді залежності, що пов'язують шукані характеристики з початковими умовами, параметрами і змінними станами моделі.

Числові методи використовують тоді, коли не вдається знайти розв'язок рівняння чи системи рівнянь аналітичними методами. У цьому випадку знаходять наближений розв'язок у числовій формі.

Якісні методи також використовуються у тому випадку, коли, не маючи розв'язку в аналітичному вигляді, можна знайти деякі властивості розв'язку (наприклад, оцінити стійкість чи продовжуваність розв'язку). Якісні методи часто використовують разом з числовими.

Якщо задачу можливо розв'язати і аналітично і чисельно, то віддавати перевагу треба аналітичному методу. Деколи аналітичний розв'язок дозволяє отримати висновки якісного характеру, які неможливо зробити на основі числового розв'язку.

При *імітаційному* моделюванні математичною моделлю є, як правило, комп'ютерна програма, яка відтворює поведінку системи в часі, причому імітуються елементарні явища, які складають процес, зі збереженням їхньої логічної структури і послідовності протікання. Це дозволяє по вхідним даним отримати інформацію про стани системи в певні моменти часу та оцінити характеристики системи. У наш час імітаційне моделювання є дуже поширеним у зв'язку з активним розвитком комп'ютерної техніки.

Основною перевагою імітаційного моделювання порівняно з аналітичним є можливість розв'язування більш складних задач, для яких не вдається побудувати систему рівнянь, що їх описують, або не вдається її розв'язати. Імітаційні моделі дозволяють досить просто враховувати такі фактори, як наявність дискретних і неперервних елементів, нелінійні характеристики елементів системи, випадкові впливи та інше, що часто створює труднощі при аналітичних дослідженнях.

В рамках імітаційного моделювання виділяють метод Монте-Карло, метод мереж Петрі, метод клітинних автоматів, метод нейронних мереж та інші.

Метод Монте-Карло використовується для моделювання систем з випадковими впливами, ймовірнісні характеристики яких відомі, і полягає в багаторазовому відтворенні моделі з наступною обробкою інформації методами математичної статистики. Метод мереж Петрі використовують для імітаційного та аналітичного моделювання дискретних технологічних процесів. Клітинні автомати дозволяють моделювати різноманітні процеси у дискретному просторі і часі з випадковими впливами чи без них.

Метод нейронних мереж полягає в тому, що створену нейронну мережу по тестовим прикладам навчають діяти так, як діє реальна система. Процес цей досить трудомісткий. Навчена нейронна мережа буде працювати так, як система, імітуючи її. До речі, нейронна мережа не містить фізичних нейронів. Чи при її програмній реалізації, чи при апаратній реалізації нейрони моделюються кількома елементарними об'єктами. З'єднання нейронів задається певним чином при створенні мережі.

Імітаційна модель може бути написана на універсальних мовах програмування, таких як Delphi,

C++, але краще використовувати спеціальні мови і середовища моделювання, як-от: GPSS, AnyLogic, Arena, AweSim, PowerSim, SIMSCRIPT, Extend тощо. У шостому і сьомому розділах цього посібника розглядається мова імітаційного моделювання GPSS World.

Комбіноване (аналітико-імітаційне) моделювання дозволяє об'єднати переваги аналітичного й імітаційного моделювання. Спочатку здійснюється розбиття процесу функціонування об'єкта на складові підпроцеси. Там, де це можливо, використовуються аналітичні моделі, а для інших підпроцесів будуються імітаційні моделі.

Лекція 2. Принципи і етапи математичного моделювання

План

1. Принципи побудови математичних моделей.
2. Етапи математичного моделювання.

1. Принципи побудови математичних моделей

Математичне моделювання є скоріше мистецтвом, ніж чіткою і завершеною теорією. Тут дуже великою є роль досвіду, інтуїції та інших інтелектуальних якостей людини. Тому неможливо скласти формалізовану інструкцію, яка б визначила, як повинна будуватись модель тієї чи іншої системи. Тим не менше відсутність точних правил не заважає досвідченим спеціалістам будувати вдалі моделі. Розглянемо основні *принципи побудови моделей*.

1. *Достатність інформації*. Якщо інформації надто мало, то модель побудувати не вдасться, якщо все відомо, то немає потреби в моделюванні. Існує критичний рівень достатності інформації для моделювання.

2. *Адекватність*. Цей принцип передбачає відповідність моделі меті дослідження і реальній системі.

3. *Відповідність моделі розв'язуваний задачі*. Модель має будуватись для розв'язування певного класу задач чи конкретної задачі дослідження системи. Спроби створення універсальної моделі, спрямованої на розв'язування великої кількості різноманітних задач, призводять до такого ускладнення, що модель виявляється практично непридатною. Досвід показує, що при розв'язуванні кожної конкретної задачі треба мати свою модель, яка відображатиме ті аспекти системи, які є найважливішими в цій задачі. Цей принцип тісно пов'язаний з принципом адекватності.

4. *Спрощення при збереженні істотних властивостей*. Модель має бути в деяких відношеннях простішою за прототип — у цьому сенс моделювання. Чим складнішою є розглядувана система, тим по можливості простішим має бути її опис, який би зумисно підкреслював типові (суттєві в цій задачі) й ігнорував менш суттєві властивості. Цей принцип можна ще назвати абстрагуванням від другорядних деталей.

5. *Відповідність між потрібною точністю результатів моделювання і складністю моделі*. Моделі по своїй природі є наближеними. Для підвищення точності моделі її потрібно деталізувати, що ускладнює модель. Тому потрібно шукати певний компроміс між точністю і складністю моделі, що часто досягається методом спроб і помилок.

6. *Баланс похибок різних видів* (моделювання, вхідних даних і заокруглень). Зокрема, точність моделі не може бути більшою за точність вхідних даних.

7. *Блочна будова*. При виконанні цього принципу полегшується розробка складних моделей і з'являється можливість використання накопиченого досвіду і готових блоків з мінімальними зв'язками між ними.

Практичні рекомендації по *спрощенню моделей*.

1. Зменшення кількості змінних у моделі, тобто деякі змінні параметри розглядаються як сталі.

2. Зміна природи змінних параметрів: дискретні параметри замінують неперервними або навпаки. Наприклад, кількість живих організмів у популяції можна вважати неперервною функцією часу (§ 6) або час вважати дискретним (§§ 2, 3).

3. Зміна функціональної залежності між змінними: нелінійна залежність часто замінюється лінійною. Наприклад, замість нелінійного диференціального рівняння $x'' + k \sin x = 0$, яке описує коливання математичного маятника, розглядають лінійне рівняння $x'' + kx = 0$, яке наближено описує малі коливання математичного маятника.

4. Зміна обмежень: додавання, вилучення чи їхня модифікація.

2. Етапи побудови математичної моделі

Етапи побудови математичної моделі:

1. Формулювання проблеми та змістова постановка задачі.
2. Створення концептуальної моделі.
3. Формалізація.
4. Перевірка адекватності моделі.
5. Дослідження моделі.
6. Аналіз результатів моделювання.
7. Оформлення результатів дослідження.

На першому етапі визначаються мета і завдання моделювання, ставиться задача. На другому етапі формується уявлення про систему: здійснюється визначення сукупності елементів, зв'язків між ними, можливих станів кожного елемента тощо.

На етапі формалізації обирається спосіб математичного моделювання і розробляється сама модель. На цьому етапі задаються диференціальні чи інші рівняння, які описують процес, або складається комп'ютерна програма, створюється мережа Петрі чи використовуються інші способи для моделювання системи. Методи математичного моделювання розглядаються у наступних розділах цього навчального посібника.

На етапі перевірки адекватності моделі здійснюється порівняння отриманої моделі з реальною системою. Адекватність перевіряється порівнянням результатів моделювання з результатами експерименту над реальною системою при однакових умовах. Модель завжди лише приблизно відповідає реальній системі, абсолютно адекватних моделей не існує. Важливою є відповідність моделі конкретній меті дослідження. Для оцінки адекватності бажано залучати фахівців, які не брали участь у розробці моделі. За результатами перевірки адекватності приймається рішення про необхідність коригування або оптимізації моделі.

Під час дослідження моделі шукають і досліджують аналітичний чи числовий розв'язок задачі при аналітичному моделюванні або планують і проводять комп'ютерний експеримент з моделлю для досягнення поставлених цілей. Далі аналізують і оцінюють результати моделювання. Наводять результати у вигляді аналітичних функцій, графіків, таблиць. За отриманими результатами формують висновки і приймають рішення.

Лекція 3. Складання диференціальних моделей

План

1. Схема складання диференціальних моделей.
2. Вентиляція приміщення.
3. Потік частинок у трубці.

1. Схема складання диференціальних моделей

Диференціальними називають моделі, які задаються з допомогою диференціальних рівнянь. Цей спосіб моделювання відносять до неперервного моделювання, хоча в деяких випадках його використовують для моделювання дискретних систем. Найпоширеніший метод побудови диференціальних рівнянь полягає у використанні фундаментальних законів природи у конкретній ситуації. Наведемо декілька таких законів:

- другий закон Ньютона ($F = ma$, де m — маса тіла, a — прискорення руху, F — сума сил, що діють на тіло) і похідні від нього закони збереження енергії, кількості руху, імпульсу тощо;
- закон всесвітнього тяжіння ($F = k\frac{m_1m_2}{r^2}$, де m_1, m_2 — маси двох тіл, r — відстань між ними, k — коефіцієнт пропорційності);
- закон розчинення речовини (швидкість розчинення пропорційна наявній кількості нерозчиненої речовини та різниці концентрацій насиченого розчину і розчину у певний момент часу);
- закон Кірхгофа (алгебрична сума сил струмів, які протікають у певній точці електричного кола, дорівнює нулю);
- закон Фур'є ($q = -\lambda(T)\frac{dT}{dx}$, де q — питомий потік теплоти, $\lambda(T)$ — коефіцієнт теплопровідності середовища, $\frac{dT}{dx}$ — швидкість зміни температури T) і аналогічний закон Нернста про дифузію речовини;
- закон Ньютона про охолодження тіла (швидкість охолодження тіла прямо пропорційна різниці температур тіла та оточуючого середовища);
- закон Гука (сила пружності пружини пропорційна її видовженню) тощо.

Можна рекомендувати дотримуватись такої послідовності дій при складанні і дослідженні диференціальних моделей:

- 1) встановити величини, які змінюються у заданому явищі чи процесі, і виявити закони (формули) відповідної науки, які ці величини пов'язують;
- 2) вибрати незалежні змінні і функції цих змінних, які потрібно знайти;
- 3) виходячи з відомих даних, визначити початкові, крайові або інші умови, які накладаються на шукані функції;
- 4) виразити усі величини через незалежні змінні, шукані функції та їхні похідні;
- 5) виходячи з закону, який описує задане явище, скласти диференціальне рівняння чи систему таких рівнянь;
- 6) зінтегрувати одержані диференціальні рівняння;
- 7) якщо задані початкові чи інші умови, знайти частинний розв'язок;
- 8) провести дослідження одержаного розв'язку.

Питання про відповідність диференціальної моделі й реального явища вивчається на основі аналізу результатів досліду та їхнього порівняння з поведінкою розв'язку одержаного диференціального рівняння.

Якщо диференціальне рівняння можна розв'язати і аналітично і чисельно, то віддавати перевагу треба аналітичному методу, бо аналітичний розв'язок дозволяє зробити висновки якісного характеру, які неможливо зробити на основі числового розв'язку. Наприклад, розв'язавши крайову задачу для рівняння коливальності струни $\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2}$ методом відокремлення змінних, можна зробити висновок про накладання хвиль і вплив окремих гармонік на формування розв'язку. Тут прослідковується тісний зв'язок з тембром звучання струни, наявністю крім основної допоміжних частот, які забезпечують відмінність звучання різних типів музичних інструментів на тих самих нотах. Числовий розв'язок

звичайно можна відобразити у вигляді анімації на площині чи поверхні у просторі, але він не дає змоги зробити згадані висновки, хоча це той самий розв'язок згідно з теоремою існування і єдиності.

2. Вентиляція приміщення

У лекційній аудиторії кубатурою 200 м^3 повітря після лекції містить $0,1 \%$ вуглекислоти. Вентилятор подає свіже повітря, що містить $0,04 \%$ вуглекислого газу, в кількості $a \text{ м}^3/\text{хв}$. Припустивши, що змішування чистого повітря з забрудненим відбувається миттєво, обчислити, якою має бути величина a , щоб після 10 хвилин перерви вміст вуглекислого газу в аудиторії не перевищував $0,06 \%$.

Позначимо вміст вуглекислого газу (в $\%$) у повітрі в момент часу t через $y(t)$. Розглянемо деякий проміжок часу Δt і знайдемо зміну концентрації вуглекислого газу в аудиторії за цей проміжок, вважаючи, що процес є рівномірним. За цей час вентилятор подає $0,0004a\Delta t \text{ м}^3$ вуглекислого газу, а виходить його з приміщення назовні через шпарини $0,01y(t+\alpha)a\Delta t \text{ м}^3$. Отже, за Δt хвилин кількість вуглекислоти в повітрі змінюється на $0,0004a\Delta t - 0,01y(t+\alpha)a\Delta t \text{ м}^3$. З іншого боку, $(y(t+\Delta t) - y(t))0,01 \cdot 200$ — це приріст вуглекислоти в приміщенні. Отже,

$$(y(t+\Delta t) - y(t))0,01 \cdot 200 = (0,0004 - 0,01y(t+\alpha))a\Delta t \Rightarrow \\ 200dy = (0,04 - y)a dt.$$

Таким чином, одержали рівняння з відокремлюваними змінними. Зінтегруємо його:

$$200 \frac{dy}{0,04 - y} = a dt \Rightarrow y = 0,04 + Ce^{-\frac{at}{200}}.$$

Оскільки $y(0) = 0,1$, то $C = 0,06$ і

$$y = 0,04 + 0,06e^{-\frac{at}{200}}.$$

Внаслідок умови $y(10) = 0,06$ маємо:

$$0,06 = 0,04 + 0,06e^{-\frac{10a}{200}} \Rightarrow a = 20 \ln 3 \approx 22 \text{ м}^3/\text{хв}.$$

3. Потік частинок у трубці

Нехай у трубці сталого поперечного перерізу S рухаються деякі частинки (наприклад, пилюка в трубці пирососа або електрони в електронній трубці) з відомою швидкістю $u(t) > 0$, залежною лише від часу t і не залежною від координати (рис. 3.1). Припустимо, що густина частинок досить мала і тому частинки між собою не взаємодіють (не зіштовхуються, не притягаються і не здійснюють впливу одна на одну). Потрібно знайти густину частинок $\rho(x, t)$ у будь-який момент часу, якщо початкова густина задана відомою функцією $\rho_0(x)$.

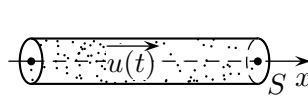


Рис. 3.1

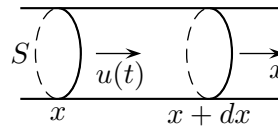


Рис. 3.2

Розглянемо елемент трубки довжиною dx від точки x до точки $x + dx$ (рис. 3.2). На протязі часу dt через лівий кінець цього елемента проходять частинки загальною масою $Su(t + \xi_1 dt)\rho(x, t + \xi_2 dt)dt$, а через правий кінець — $Su(t + \xi_1 dt)\rho(x + dx, t + \xi_3 dt)dt$, де $u(t + \xi_1 dt)$ — середнє значення швидкості, а $\rho(x, t + \xi_2 dt)$ і $\rho(x + dx, t + \xi_3 dt)$ — середні густини за час dt , $0 \leq \xi_1, \xi_2, \xi_3 \leq 1$. Отже, зміна маси частинок в елементі трубки за час dt становить

$$dm = Su(t + \xi_1 dt)(\rho(x, t + \xi_2 dt) - \rho(x + dx, t + \xi_3 dt))dt.$$

З іншого боку,

$$dm = S(\rho(x + \xi_4 dx, t + dt) - \rho(x + \xi_5 dx, t))dx, \quad 0 \leq \xi_4, \xi_5 \leq 1,$$

де $\rho(x + \xi_4 dx, t + dt)$ і $\rho(x + \xi_5 dx, t)$ — середні густини частинок в елементі трубки в кінцевий і початковий моменти часу.

Тоді згідно з законом збереження маси

$$\begin{aligned} u(t + \xi_1 dt)(\rho(x, t + \xi_2 dt) - \rho(x + dx, t + \xi_3 dt))dt = \\ = (\rho(x + \xi_4 dx, t + dt) - \rho(x + \xi_5 dx, t))dx. \end{aligned}$$

Поділивши обидві частини останнього рівняння на dx , dt і спрямувавши їх до нуля, отримуємо диференціальне рівняння з частинними похідними першого порядку:

$$u(t) \frac{\partial \rho}{\partial x} + \frac{\partial \rho}{\partial t} = 0. \quad (3.1)$$

Початкова умова для цього рівняння має вигляд:

$$\rho(x, t)|_{t=0} = \rho_0(x). \quad (3.2)$$

Зінтегруємо лінійне однорідне рівняння (3.1) методом характеристик:

$$\frac{dx}{u(t)} = \frac{dt}{1} \Rightarrow dx = u(t)dt \Rightarrow x - \int_0^t u(\tau)d\tau = C.$$

Тоді загальним розв'язком рівняння (3.1) є функція

$$\rho(x, t) = \varphi\left(x - \int_0^t u(\tau)d\tau\right),$$

де $\varphi(\xi)$ — довільна неперервно диференційовна функція. З початкової умови (3.2):

$$\rho(x, 0) = \varphi(x) = \rho_0(x).$$

Отже,

$$\rho(x, t) = \rho_0\left(x - \int_0^t u(\tau)d\tau\right)$$

— розв'язок початкової задачі (3.1), (3.2).

Припустимо, що $u(t) \equiv u_0$ (швидкість стала). Тоді густина

$$\rho(x, t) = \rho_0(x - u_0 t).$$

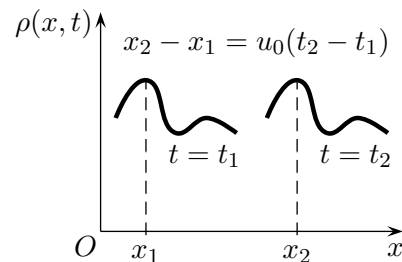


Рис. 3.3

Профіль густини без змін переноситься вздовж потоку зі сталою швидкістю (рис. 3.3). Тому має місце біжуча хвиля. У випадку, коли швидкість $u(t)$ не є сталою, профіль густини переноситься за рівні проміжки часу на різні відстані.

Лекція 4. Задачі фізики, які приводять до диференціальних моделей

План

1. Витікання рідини.
2. Прогин балок.

1. Витікання рідини

Нехай деяка посудина заповнена рідиною до рівня H . У дні посудини є отвір площею ω , через який витікає рідина. Площа поперечного перерізу посудини горизонтальною площиною є відомою функцією $S(h)$ висоти h від дна посудини. Потрібно визначити час T витікання всієї рідини з посудини та час t зниження рівня рідини від початкового H до заданого h (рис. 4.1).

Нехай висота рівня рідини в посудині у деякий момент часу t дорівнює h . Досліди показують, що зі зменшенням рівня рідини у посудині швидкість витікання рідини зменшується. Нехай $v(h)$ — швидкість витікання рідини, залежна від рівня h рідини в посудині. Об'єм рідини ΔV , що витікає з посудини за проміжок часу Δt від моменту t до моменту $t + \Delta t$, можна обчислити як об'єм циліндра з площею основи ω і висотою $v(h)\Delta t$. Таким чином,

$$\Delta V = \omega v(h)\Delta t.$$

Цей самий об'єм рідини може бути обчислений іншим способом. Внаслідок витікання рідини її рівень h змінюється на величину Δh , отже, $\Delta V = -S(h)\Delta h$ (знак мінус береться тому, що $\Delta h < 0$). Прирівнюючи обидва вирази для ΔV , маємо рівняння

$$\omega v(h)\Delta t = -S(h)\Delta h,$$

звідки

$$\frac{\Delta t}{\Delta h} = -\frac{S(h)}{\omega v(h)}.$$

Спрямовуючи Δh до нуля, отримуємо диференціальне рівняння

$$\frac{dt}{dh} = -\frac{S(h)}{\omega v(h)}.$$

Інтегруючи це рівняння і враховуючи початкову умову $t = 0$ для $h = H$, приходимо до формул

$$t = \int_h^H \frac{S(h)}{\omega v(h)} dh, \quad T = \int_0^H \frac{S(h)}{\omega v(h)} dh. \quad (4.1)$$

Якщо витікання рідини відбувається через невеликий отвір або коротку трубку, то згідно з законом Торрічеллі, $v = \mu\sqrt{2gh}$, де g — прискорення вільного падіння ($g \approx 9,8 \frac{\text{м}}{\text{с}^2}$), а μ — коефіцієнт витрат (для води $\mu = 0,62$). У цьому випадку формули (4.1) набувають вигляду

$$t = \frac{1}{\omega\mu\sqrt{2g}} \int_h^H \frac{S(h)}{\sqrt{h}} dh, \quad T = \frac{1}{\omega\mu\sqrt{2g}} \int_0^H \frac{S(h)}{\sqrt{h}} dh. \quad (4.2)$$

Приклад. Визначити час витікання всього гасу з заповненої циліндричної залізничної цистерни довжиною $L = 12$ м і діаметром $D = 2,6$ м через коротку зливну трубку з площею поперечного перерізу $\omega = 20 \text{ см}^2$ у нижній частині цистерни. Коефіцієнт витрат для гасу $\mu = 0,6$.

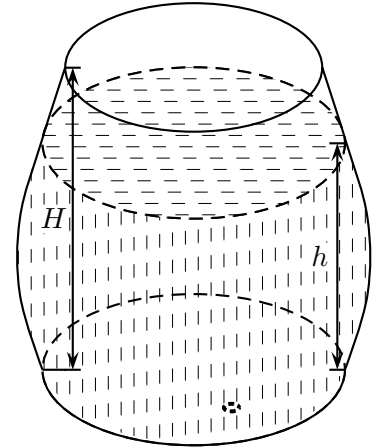


Рис. 4.1

Розв’язання. Зрозуміло, що в нормальних умовах вісь циліндра залізничної цистерни розміщена горизонтально. Тому кожен переріз цієї цистерни горизонтальною площиною є прямокутником зі сталою довжиною L і змінною шириною $2x$ (рис. 4.2). Знайдемо половину ширини x . Нехай $R = \frac{D}{2}$ — радіус циліндра. На рис. 4.3 зображено поперечний переріз цистерни вертикальною площиною, перпендикулярною до осі циліндра. За теоремою Піфагора $x = \sqrt{R^2 - (h - R)^2}$. Отже, $S(h) = 2L\sqrt{(D - h)h}$, а тому

$$T = \frac{2L}{\omega\mu\sqrt{2g}} \int_0^D \sqrt{D - h} dh = \frac{4LD\sqrt{D}}{3\omega\mu\sqrt{2g}} \approx 12626 \text{ с} \approx 3,5 \text{ год.} \blacksquare$$

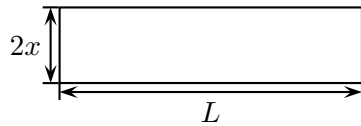


Рис. 4.2

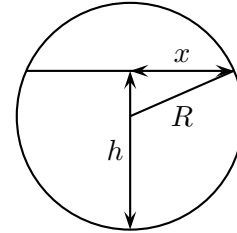


Рис. 4.3

2. Прогин балок

Розглянемо горизонтально розташовану балку сталого поперечного перерізу, зроблену з однорідного матеріалу (рис. 4.4). Її вісь симетрії позначено пунктиром. Ця балка може прогинатись під дією різних сил, що діють у вертикальній площині, а також ваги самої балки (рис. 4.5). Зрозуміло, що під дією цих сил вісь симетрії балки буде викривлятися. Викривлену вісь симетрії називають *пружною лінією* балки. Визначення форми цієї лінії відіграє велику роль у теорії пружності.



Рис. 4.4.



Рис. 4.5.

Нейтральним шаром балки називають той шар, в якому волокна балки не розтягаються і не стискаються під дією тих сил, що зумовлюють прогин балки. Вісь симетрії і пружна лінія балки проходять через нейтральний шар балки.

Існують різні типи балок в залежності від способів їхнього кріплення або опори. Балка, в якій один кінець жорстко закріплений, а інший — вільний, називається *консольною балкою* (рис. 4.6). Балка з рис. 4.7 вільно лежить на двох опорах. Бувають також інші типи балок (рис. 4.8).

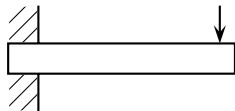


Рис. 4.6.

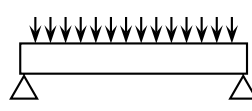


Рис. 4.7.

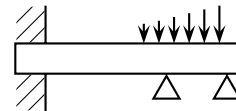


Рис. 4.8.

Існують також різні способи прикладання сил до балки. Сила може бути, наприклад, зосередженою в точці (рис. 4.6), рівномірно розподіленою (рис. 4.7), змінною (рис. 4.8).

Для горизонтальної балки вісь абсцис проведемо горизонтально через невикривлену вісь симетрії балки, а вісь ординат спрямуємо вертикально вниз так, щоб лівий кінець балки розміщувався у початку координат (рис. 4.9).

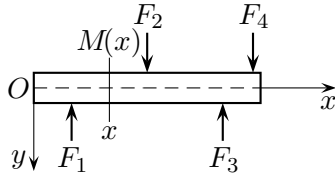


Рис. 4.9.

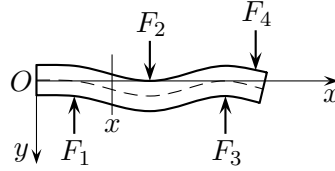


Рис. 4.10.

Під дією зовнішніх сил F_1, F_2, F_3, F_4 та інших вісь симетрії викривляється в пружну лінію. Зміщення пружної лінії від осі абсцис називають прогином балки в точці x . Таким чином, якщо відомим є рівняння пружної лінії балки, то завжди можна знайти прогин балки у кожній точці. Розглянемо тепер відшукання рівняння пружної лінії балки.

Позначимо через $M(x)$ згинальний момент у вертикальному перерізі балки з координатою x . *Згинальний момент* є алгебричною сумою моментів всіх сил, що діють на балку з одного боку (справа або зліва) від точки x . При цьому моменти сил, які діють зверху вниз, беруться зі знаком «плюс», а моменти сил, які діють знизу вверх, беруться зі знаком «мінус». *Момент сили* — це добуток сили на відстань від точки прикладання сили до точки x .

У курсі опору матеріалів доводиться, що кривизна $K(x)$ пружної лінії пов'язана зі згинальним моментом $M(x)$ співвідношенням

$$K(x) = \frac{M(x)}{E(x)J(x)},$$

де $E(x)$ — модуль пружності Юнга, який залежить від матеріалу балки, а $J(x)$ — геометричний момент інерції поперечного перерізу балки відносно *нейтральної осі* — прямої перетину нейтрального шару балки з площиною цього перерізу. Нейтральна вісь є горизонтальною прямою, що проходить через центр ваги поперечного перерізу балки (рис. 4.11). Якщо поперечний переріз балки є однаковим у різних точках, а балка складається з одного матеріалу, то величини E і J є сталими.

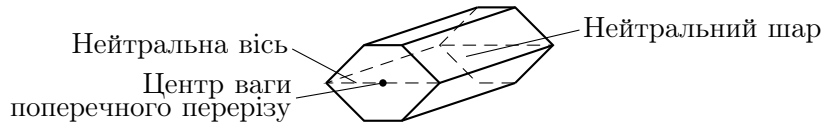


Рис. 4.11.

З математичного аналізу відомо, що кривизна $K(x)$ подається формулою

$$K(x) = \frac{y''}{(1 + y'^2)^{\frac{3}{2}}},$$

де $y(x)$ — відхилення пружної лінії від прямої в точці x . Якщо балка лише злегка прогинається, що часто буває на практиці, то кутовий коефіцієнт y' пружної лінії є дуже малим і ним можна знехтувати. Тоді отримуємо диференціальне рівняння пружної лінії балки:

$$y'' = \frac{M(x)}{EJ}. \quad (4.3)$$

З математичного аналізу відомо, що момент інерції J обчислюється за формулою

$$J = I_x = \iint_{(D)} y^2 dx dy, \quad (4.4)$$

де (D) — поперечний переріз балки, центр ваги якого знаходиться у початку координат. Момент інерції залежить лише від поперечного перерізу, тобто форми балки і не залежить від матеріалу.

Приклад. Вільний кінець консольної сталеві балки довжиною $l = 4$ м навантажений зосередженою силою $P = 20000$ Н. Знайти рівняння пружної лінії і визначити величину прогину вільного кінця

балки, якщо балка має сталий поперечний переріз, який є прямокутником з шириною 3 см і висотою 20 см. Вагою самої балки знехтувати. Модуль пружності Юнга для сталі $E = 2,1 \cdot 10^{11}$ Н/м².

Розв'язання. Спочатку знаходимо момент інерції J . Поперечний переріз з центром системи координат у центрі його ваги зображено на рис. 4.12. Користуючись формулою (4.4), маємо:

$$J = \int_{-0,1}^{0,1} dy \int_{-0,015}^{0,015} y^2 dx = \frac{4}{3} \cdot 0,1^3 \cdot 0,015 = 2 \cdot 10^{-5} \text{ м}^4.$$

На рис. 4.13 схематично зображено консольну балку з прикладеною до неї силою P . Визначаємо згинальний момент $M(x)$ для довільної точки $x \in (0, l)$. У даному випадку $M(x)$ дорівнює моменту сили P відносно точки x , взятому зі знаком «плюс», тобто

$$M(x) = P(l - x).$$

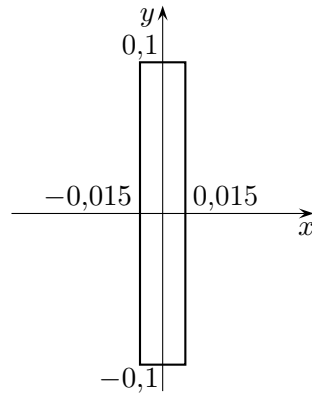


Рис. 4.12.

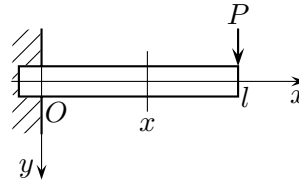


Рис. 4.13.

Отже, згідно з формулою (4.3) отримуємо диференціальне рівняння

$$y'' = \frac{P}{EJ}(l - x). \quad (4.5)$$

Умова закріплення лівого кінця консольної балки дає початкові умови

$$y(0) = 0, \quad y'(0) = 0. \quad (4.6)$$

Безпосередньо інтегруючи диференціальне рівняння (4.5), знаходимо його загальний розв'язок:

$$\begin{aligned} y' &= -\frac{P}{EJ} \frac{(l - x)^2}{2} + C_1, \\ y &= \frac{P}{EJ} \frac{(l - x)^3}{6} + C_1 x + C_2. \end{aligned}$$

Враховуючи початкові умови (4.6), отримуємо:

$$\begin{cases} \frac{P}{EJ} \frac{l^3}{6} + C_2 = 0, \\ -\frac{P}{EJ} \frac{l^2}{2} + C_1 = 0 \end{cases} \Rightarrow C_1 = \frac{Pl^2}{2EJ}, \quad C_2 = -\frac{Pl^3}{6EJ}.$$

Тому рівняння пружної лінії має вигляд

$$y = \frac{P}{2EJ} \left(lx^2 - \frac{x^3}{3} \right).$$

Величина прогину h в кінці балки отримується з останньої рівності при $x = l$:

$$h = \frac{P}{2EJ} \left(l^3 - \frac{l^3}{3} \right) = \frac{Pl^3}{3EJ} = \frac{20000 \cdot 4^3}{3 \cdot 2,1 \cdot 10^{11} \cdot 2 \cdot 10^{-5}} \approx 0,1 \text{ м.} \blacksquare$$

Лекція 5. Задачі хімії і біології, які приводять до диференціальних моделей

План

1. Закон діючих мас у хімії.
2. Модель розвитку епідемії.

1. Закон діючих мас у хімії

Закон діючих мас: швидкість хімічної реакції, яка відбувається при сталій температурі, пропорційна добутку мас хімічних речовин, які вступають у реакцію.

Нехай у реакцію вступають дві рідкі хімічні речовини A і B , внаслідок реакції утворюється нова рідка хімічна речовина C . З одного об'єму речовини A і двох об'ємів речовини B утворюється три об'єми речовини C . Реакція відбувається при сталій температурі. Спочатку було 20 л речовини A і 10 л речовини B , речовини C не було. Через 20 хвилин утворилось 6 л речовини C .

Через $x(t)$ позначимо кількість хімічної речовини C в момент часу t . Тоді отримуємо диференціальне рівняння

$$\frac{dx}{dt} = k \left(20 - \frac{x}{3} \right) \left(10 - \frac{2x}{3} \right)$$

і умови

$$x(0) = 0, \quad x(20) = 6.$$

Зінтегрувавши рівняння і врахувавши умови, отримаємо залежність кількості утвореної хімічної речовини C від часу:

$$x = \frac{60 - 60 \left(\frac{3}{2} \right)^{3t}}{1 - 4 \left(\frac{3}{2} \right)^{3t}}.$$

2. Модель розвитку епідемії

Припустимо, що деяка популяція, яка налічує N особин, ділиться на три групи. У першу з них включаються здорові особини, що є сприйнятливими до деякої конкретної хвороби. Їхню кількість у момент часу t позначаємо через $S(t)$. У другу групу включаються хворі особини, які є джерелом інфекції для особин з першої групи. Кількість таких особин у популяції в момент часу t позначаємо через $I(t)$. До третьої групи відносять особин, що є здоровими і мають імунітет до хвороби. Їхню чисельність позначаємо через $R(t)$. Таким чином,

$$S(t) + I(t) + R(t) = N.$$

Всі хворі особини виліковуються й отримують імунітет до хвороби. Іншим способом отримати імунітет неможливо. Якщо кількість хворих особин не перевищує деякого числа I^* , то хворих можна ізолювати від здорових і припинити подальший розвиток епідемії. Якщо $I(t) > I^*$, то швидкість зміни кількості сприйнятливих до хвороби особин пропорційна їхній чисельності з коефіцієнтом захворюваності $\alpha > 0$. Швидкість зміни чисельності особин з імунітетом пропорційна кількості хворих особин з коефіцієнтом одужання $\beta > 0$. Початкову кількість хворих (в момент часу $t = 0$) будемо позначати через I_0 . Крім того, припускаємо, що в початковий момент часу особин з імунітетом не було. Всі ці припущення спростують реальну ситуацію, але часто вони досить точно відображають сутність справи.

Складемо тепер диференціальну модель. Для $I(t) \leq I^*$ маємо:

$$\begin{cases} \frac{dS}{dt} = 0, \\ \frac{dI}{dt} = -\beta I, \\ \frac{dR}{dt} = \beta I. \end{cases} \quad (5.1)$$

Для $I(t) > I^*$ отримуємо:

$$\begin{cases} \frac{dS}{dt} = -\alpha S, \\ \frac{dI}{dt} = \alpha S - \beta I, \\ \frac{dR}{dt} = \beta I. \end{cases} \quad (5.2)$$

Початкові умови: $S(0) = N - I_0$, $I(0) = I_0$, $R(0) = 0$.

Випадок 1. Число $I_0 \leq I^*$. У цьому випадку маємо лише одну систему

$$\begin{cases} \frac{dS}{dt} = 0, \\ \frac{dI}{dt} = -\beta I, \end{cases}$$

що означає, що хворі ізольовані від здорових особин без імунітету до хвороби. Тоді

$$\begin{aligned} S(t) &= S(0) = N - I_0, \\ I(t) &= I_0 e^{-\beta t}, \\ R(t) &= N - S(t) - I(t) = I_0(1 - e^{-\beta t}). \end{aligned}$$

На рис. 5.1 графічно показано зміну кількості особин у кожній з трьох груп зі зростанням часу t .

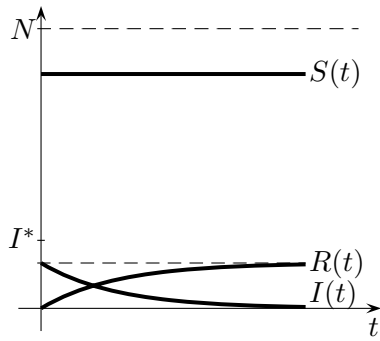


Рис. 5.1.

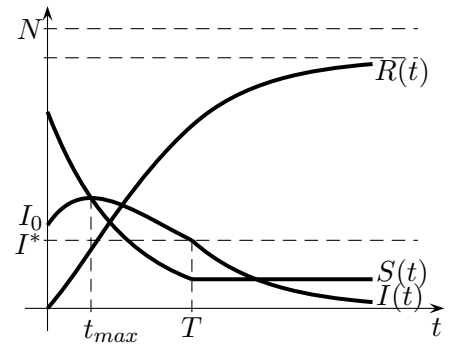


Рис. 5.2.

Випадок 2. $I_0 > I^*$, $\alpha = \beta$. Тоді має існувати інтервал $0 \leq t < T$, для всіх значень t якого справджується нерівність $I(t) > I^*$, бо за змістом задачі I як функція змінної t має бути неперервною. Отже, для всіх $t \in [0, T)$ хвороба буде поширюватись на сприйнятливих до неї особин. Таким чином, для $t \in [0, T)$

$$\begin{cases} \frac{dS}{dt} = -\alpha S, \\ \frac{dI}{dt} = \alpha S - \alpha I. \end{cases}$$

Тоді

$$\begin{aligned} S(t) &= (N - I_0)e^{-\alpha t}, \\ \frac{dI}{dt} + \alpha I &= \alpha(N - I_0)e^{-\alpha t}. \end{aligned} \quad (5.3)$$

Ми отримали лінійне диференціальне рівняння першого порядку. Застосуємо для його інтегрування метод варіації довільної сталої. Відповідне лінійне однорідне рівняння

$$\frac{dI}{dt} + \alpha I = 0$$

має загальний розв'язок $I(t) = Ce^{-\alpha t}$. Тоді розв'язок неоднорідного рівняння (5.3) шукаємо у вигляді $I(t) = C(t)e^{-\alpha t}$. Після підстановки в (5.3) маємо:

$$C'(t)e^{-\alpha t} = \alpha(N - I_0)e^{-\alpha t} \Rightarrow C(t) = \alpha(N - I_0)t + C,$$

звідки

$$I(t) = e^{-\alpha t}(\alpha(N - I_0)t + C).$$

Враховуючи початкову умову $I(0) = I_0$, отримуємо:

$$I(t) = e^{-\alpha t}(\alpha(N - I_0)t + I_0) \quad (5.4)$$

для $t \in [0, T]$.

Для того щоб знайти T , треба підставити в (5.4) замість $I(t)$ значення I^* :

$$I^* = e^{-\alpha T}(\alpha(N - I_0)T + I_0).$$

Розв'язавши останнє трансцендентне рівняння при відомих числових значеннях параметрів числовими методами або з допомогою математичного пакета, можна знайти час T , коли припиниться розповсюдження хвороби.

Знайдемо тепер таке $t_{\max}$, при якому кількість хворих особин буде максимальною. Для дослідження функції $I(t)$ на екстремум прирівняємо до нуля її похідну

$$\begin{aligned} \frac{dI}{dt} &= e^{-\alpha t}(\alpha(N - I_0) - \alpha^2(N - I_0)t - \alpha I_0) = 0 \quad \Rightarrow \\ t_{\max} &= \frac{N - 2I_0}{\alpha(N - I_0)}. \end{aligned}$$

З виразу для похідної видно, що $\frac{dI}{dt} < 0$ для $t > t_{\max}$ і $\frac{dI}{dt} > 0$ для $t < t_{\max}$. Тому в точці $t = t_{\max}$ справді досягається максимум.

Якщо тепер підставити значення $t_{\max}$ у рівність (5.4), то отримаємо:

$$I(t_{\max}) = e^{\frac{2I_0 - N}{N - I_0}}(N - I_0) = S(t_{\max}).$$

Остання рівність показує, зокрема, що у момент часу $t_{\max}$ кількість сприйнятливих до хвороби особин збігається з кількістю хворих.

Якщо $t > T$, то сприйнятливі до хвороби особини вже не захворіють і $I(t) = I^*e^{-\alpha(t-T)}$.

На рис. 5.2 графічно показано, як з плином часу t змінюватиметься кількість особин у кожній з трьох груп.

Випадок 3. $I_0 > I^*$, $\alpha \neq \beta$. Цей випадок доводиться розглядати окремо від попереднього, бо тут виходять дещо інші розв'язки диференціальних рівнянь. Нехай $0 \leq t < T$ — інтервал, для всіх значень t якого справджується нерівність $I(t) > I^*$. Для $t \in [0, T]$

$$\begin{cases} \frac{dS}{dt} = -\alpha S, \\ \frac{dI}{dt} = \alpha S - \beta I. \end{cases}$$

Тоді

$$\begin{aligned} S(t) &= (N - I_0)e^{-\alpha t}, \\ \frac{dI}{dt} + \beta I &= \alpha(N - I_0)e^{-\alpha t}. \end{aligned} \quad (5.5)$$

Ми отримали лінійне диференціальне рівняння першого порядку. Застосуємо для його інтегрування метод варіації довільної сталої. Відповідне лінійне однорідне рівняння

$$\frac{dI}{dt} + \beta I = 0$$

має загальний розв'язок $I(t) = Ce^{-\beta t}$. Тоді розв'язок неоднорідного рівняння (5.5) шукаємо у вигляді $I(t) = C(t)e^{-\beta t}$. Після підстановки в (5.5) маємо:

$$C'(t)e^{-\beta t} = \alpha(N - I_0)e^{-\alpha t} \Rightarrow C(t) = \frac{\alpha(N - I_0)}{\beta - \alpha}e^{(\beta - \alpha)t} + C,$$

звідки

$$I(t) = \frac{\alpha(N - I_0)}{\beta - \alpha}e^{-\alpha t} + Ce^{-\beta t}.$$

Враховуючи початкову умову $I(0) = I_0$, отримуємо:

$$I(t) = \frac{\alpha(N - I_0)}{\beta - \alpha}e^{-\alpha t} + \frac{\beta I_0 - \alpha N}{\beta - \alpha}e^{-\beta t} \quad (5.6)$$

для $t \in [0, T)$.

Для того щоб знайти T , треба підставити в (5.6) замість $I(t)$ значення I^* :

$$I^* = \frac{\alpha(N - I_0)}{\beta - \alpha}e^{-\alpha T} + \frac{\beta I_0 - \alpha N}{\beta - \alpha}e^{-\beta T}.$$

Розв'язавши останнє трансцендентне рівняння при відомих числових значеннях параметрів числовими методами або з допомогою математичного пакета, можна знайти час T , коли припиниться розповсюдження хвороби.

Як і у випадку 2 можна було б знайти точку максимуму $t_{\max}$ функції $I(t)$, проте для різних чисел α і β не буде виконуватись рівність $I(t_{\max}) = S(t_{\max})$.

Якщо $t > T$, то сприйнятливі до хвороби особи вже не захворіють і $I(t) = I^*e^{-\beta(t-T)}$.

На рис. 5.3 для $\alpha < \beta$ графічно показано зміну кількості осіб у кожній з трьох груп зі зростанням часу t .

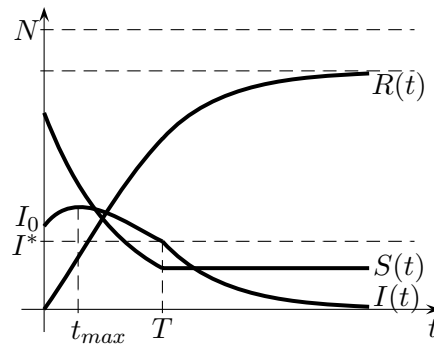


Рис. 5.3.

Лекція 6. Взаємодія в системі «хижак-жертва»

План

1. Система Лотки–Вольтерри.
2. Модифікована система Лотки–Вольтерри.

1. Система Лотки–Вольтерри

Під час першої світової війни суттєво (від 12 % до 36 %) зросли відсотки хрящових риб (акули, скати і т. п.) у промислових виловах у Середземному морі. Хрящові риби не мають промислової цінності, їхнє м'ясо люди в їжу зазвичай не вживають. Зрозуміло, що під час війни вилов риби в Середземному морі зменшився, але екологи і біологи не змогли пояснити, чому зросла кількість хрящових риб, які є хижаками. Один з біологів звернувся до відомого італійського математика Віто Вольтерри з проханням пояснити, у чому річ. Вольтерра склав математичну модель співіснування хижих і нехижих риб. Складемо її і ми.

Нехай $x(t)$ — кількість нехижих риб у момент часу t , а $y(t)$ — кількість хижих риб у момент часу t . Якщо кількість риб достатньо велика, то ми можемо вважати, що зміна чисельності на одну рибину є незначною, а функція кількості риб є неперервною і диференційовною. Приріст числа нехижих риб пропорційний їхній кількості з коефіцієнтом пропорційності α (закон Мальтуса). У той самий час, число контактів між хижими і нехижими рибами пропорційно добутку кількостей цих риб. Таким чином, $\frac{dx}{dt} = \alpha x - \beta xy$, де β — коефіцієнт смертності нехижих риб. Аналогічно, $\frac{dy}{dt} = -\gamma y + \delta xy$, де γ — коефіцієнт смертності хижаків, а δ — їхнього розмноження (чим більше їжі та хижаків, тим більше стає хижаків). Коефіцієнти α, β, γ і δ є додатними. Отже, маємо систему нелінійних диференціальних рівнянь першого порядку

$$\begin{cases} \frac{dx}{dt} = \alpha x - \beta xy, \\ \frac{dy}{dt} = -\gamma y + \delta xy, \end{cases} \quad (6.1)$$

яку називають системою диференціальних рівнянь Лотки–Вольтерри (Альфред Лотка практично одночасно отримав цю систему для динаміки популяцій). Нехай x_0 і y_0 — початкові кількості нехижих і хижих риб, тоді отримуємо початкові умови $x(0) = x_0, y(0) = y_0$.

Для того щоб динамічна система мала положення рівноваги, необхідно і достатньо, щоб для нього праві частини системи перетворювались у нуль. Отже, система (6.1) має два положення рівноваги: $x \equiv 0, y \equiv 0$ і $\bar{x} \equiv \frac{\gamma}{\beta}, \bar{y} \equiv \frac{\alpha}{\delta}$. Перший розв'язок нецікавий, бо він означає відсутність всіх риб. Другий розв'язок означає, що чисельність риб з часом не змінюється.

Поділимо перше рівняння системи (6.1) на друге і зінтегруємо результат:

$$\begin{aligned} \frac{dx}{dy} &= \frac{(\alpha - \beta y)x}{(-\gamma + \delta x)y} \Rightarrow \frac{(-\gamma + \delta x)dx}{x} = \frac{\alpha - \beta y}{y} dy \Rightarrow \\ &-\gamma \ln x + \delta x = \alpha \ln y - \beta y + C \Rightarrow \\ &y^\alpha e^{-\beta y} x^\gamma e^{-\delta x} = C \end{aligned} \quad (6.2)$$

(тут стали C перепозначено, а модулів під знаком логарифма немає, бо $x > 0, y > 0$).

Теорема. Всі фазові траєкторії системи (6.1) є замкненими.

Доведення. Спочатку визначимо поведінку функцій $f(y) = y^\alpha e^{-\beta y}$ і $g(x) = x^\gamma e^{-\delta x}$ для додатних x і y . Бачимо, що $f(0) = 0, f(\infty) = 0$ і $f(y) > 0$ для всіх $y > 0$. Знайшовши похідну

$$f'(y) = \frac{\alpha y^{\alpha-1} - \beta y^\alpha}{e^{\beta y}} = \frac{y^{\alpha-1}(\alpha - \beta y)}{e^{\beta y}},$$

помічаємо, що функція $f(y)$ має одну критичну точку $y = \frac{\alpha}{\beta}$. Отже, $f(y)$ досягає свого максимального значення $M_y = \left(\frac{\alpha}{\beta}\right)^\alpha e^{-\alpha}$ при $y = \frac{\alpha}{\beta}$, а графік функції $f(y)$ має вигляд, показаний на рис. 6.1. Аналогічно, функція $g(x)$ досягає свого максимального значення $M_x = \left(\frac{\gamma}{\delta}\right)^\gamma e^{-\gamma}$ при $x = \frac{\gamma}{\delta}$, а графік функції $g(x)$ має вигляд, поданий на рис. 6.2.

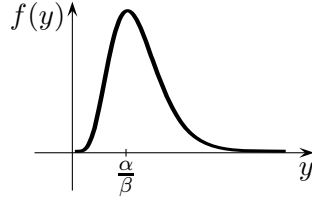


Рис. 6.1.

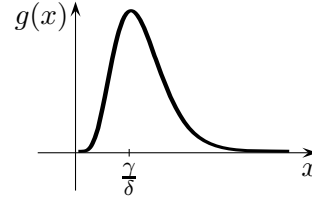


Рис. 6.2.

З попереднього аналізу робимо висновок про те, що рівняння (6.2) не має розв'язків $x, y > 0$ для $C > M_x M_y$ і має тільки єдиний розв'язок $x = \frac{\gamma}{\delta}$, $y = \frac{\alpha}{\beta}$ для $C = M_x M_y$. Таким чином, нам залишається розглянути випадок $C = \lambda M_y$, де $\lambda \in$ додатним числом, меншим за M_x . Зазначимо, що рівняння $x^\gamma e^{-\delta x} = \lambda$ має один розв'язок $x = x_m < \frac{\gamma}{\delta}$ і один розв'язок $x = x_M > \frac{\gamma}{\delta}$. Звідси випливає, що рівняння

$$f(y) = y^\alpha e^{-\beta y} = \left(\frac{\lambda}{x^\gamma e^{-\delta x}} \right) M_y$$

не має розв'язку y , коли $x < x_m$ або $x > x_M$. Воно має єдиний розв'язок $y = \frac{\alpha}{\beta}$, коли $x = x_m$ або $x = x_M$, і має два розв'язки $y_1(x)$ і $y_2(x)$ для кожного x між x_m і x_M . Менший розв'язок $y_1(x)$ завжди менший за $\frac{\alpha}{\beta}$, тоді як більший розв'язок $y_2(x)$ завжди більший за $\frac{\alpha}{\beta}$. Коли x прямує до x_m або x_M , обидві функції $y_1(x)$ і $y_2(x)$ прямують до $\frac{\alpha}{\beta}$. Отже, криві, визначені рівнянням (6.2), є замкненими для додатних x, y і мають вигляд, подібний до наведеного на рис. 6.3. Оскільки жодна з цих замкнених кривих (за винятком $\bar{x} = \frac{\gamma}{\delta}$, $\bar{y} = \frac{\alpha}{\beta}$) не містить точок рівноваги системи (6.1), то всі розв'язки $x(t), y(t)$ системи (6.1), такі, що $x(0) > 0, y(0) > 0$, є періодичними функціями часу, тобто для них виконуються рівності $x(t+T) = x(t), y(t+T) = y(t)$ для деякого додатного числа T . Теорему доведено.

З попереднього бачимо, що положення рівноваги $\bar{x}, \bar{y} \in$ центром. Коливання чисельності відбуваються не в фазі: максимальному значенню $x(t)$ відповідає середнє значення $y(t)$ і навпаки (на рис. 6.4 наведено графіки одного з розв'язків системи (6.1), побудовані з допомогою математичного пакета для конкретних значень параметрів).

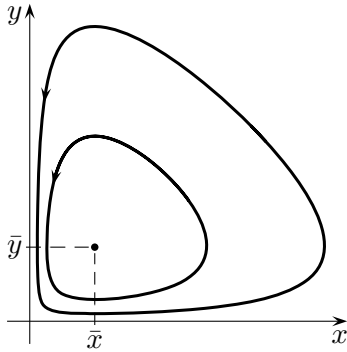


Рис. 6.3.

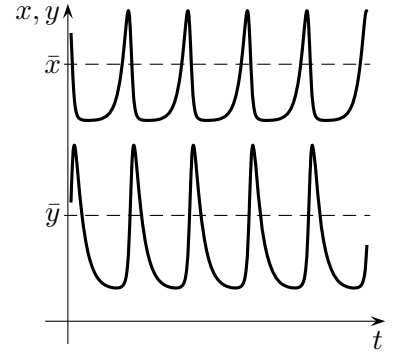


Рис. 6.4.

Якщо тепер припустити, що має місце промисловий вилов риби, то отримаємо систему диференціальних рівнянь

$$\begin{cases} \frac{dx}{dt} = \alpha x - \beta xy - cx, \\ \frac{dy}{dt} = -\gamma y + \delta xy - cy, \end{cases}$$

де c — додатний коефіцієнт вилову риби. Оскільки ловляться всі риби підряд, то коефіцієнт є тим самим в обидвох рівняннях. Для останньої системи положенням рівноваги буде точка $\bar{x} = \frac{\gamma+c}{\delta}, \bar{y} = \frac{\alpha-c}{\beta}$, що означає зростання $\bar{x}$ і зменшення $\bar{y}$ (бо $c > 0$). Отже, збільшення вилову риби приводить до зростання кількості промислової риби і зменшення кількості хижаків. Цей принцип називають принципом Вольтерри.

Аналогічну модель можна скласти і для інших систем взаємодії між хижаками і жертвами, наприклад, для систем коти — миші, вовки — зайці тощо.

У 1868 році з Австралії в США випадково завезли комаху-шкідника, який мало не знищив виробництво цитрусових. Оскільки в Австралії у нього є природний ворог — комаха, що його їсть, довелося завести в Америку і цього хижака, внаслідок чого чисельність комах врегулювалась. У XX столітті проти шкідників застосували хімічні засоби, наслідком чого стало зростання кількості шкідників згідно з принципом Вольтерри. Тому від застосування отрутохімікатів довелося відмовитись.

2. Модифікована система Лотки–Вольтерри

Запропонована модель Лотки–Вольтерри лише наближено описує реальну систему. Про це свідчить, зокрема, те, що в реальних системах коливання чисельності хижаків і жертв звичайно не відбувається.

Узагальнена модель Лотки–Вольтерри має вигляд

$$\begin{cases} \frac{dx}{dt} = \alpha x - \beta xy - ax^2, \\ \frac{dy}{dt} = -\gamma y + \delta xy - by^2, \end{cases} \quad (6.3)$$

де a і b — коефіцієнти, які відповідають конкурентній боротьбі всередині популяції. Розв'язок системи (6.3) наближається до положення рівноваги по спіралі (рис. 6.5), тобто точка $\bar{x}, \bar{y}$ — фокус.

Звичайно, і ця модель не описує всі випадки. По-перше, деколи (в унікальних, експериментальних випадках) хижаки можуть з'їсти всіх жертв, а потім померти з голоду. А такого розв'язку у цієї системи рівнянь бути не може. По-друге, якщо жертви можуть десь заховатись, то хижаки їх не з'їдять, поки ті не вийдуть зі свого сховища. Такі системи вже не є детермінованими і велику роль тут відіграє випадковий фактор (наприклад, можливість і вміння заховатись). Для моделювання таких систем потрібно використовувати інші методи. У наступній лекції розглядається моделювання систем взаємодії між хижаками і жертвами з допомогою клітинних автоматів, причому результати моделювання є подібними до отриманих засобами диференціальних рівнянь.

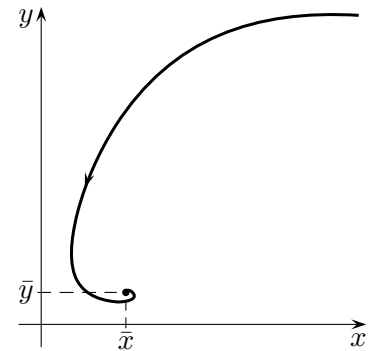


Рис. 6.5.

Лекція 7. Основи аналітичного моделювання систем масового обслуговування

План

1. Поняття системи масового обслуговування.
2. Характеристики потоку подій і вхідний потік вимог.
3. Моделювання стаціонарного пуассонівського потоку.
4. Правила обслуговування, організація черги і типи моделей систем масового обслуговування.
5. Формула Літтла.

1. Поняття системи масового обслуговування

У різних сферах людської діяльності часто виникає масовий попит на певні послуги. Системи, спрямовані на задоволення цього попиту, називають *системами масового обслуговування*. З такими системами нам доводиться зустрічатись щодня. Це процеси обслуговування в магазині, кафе, бібліотеці, на автозаправній станції, транспортні системи, служби ремонту тощо. Виробничі процеси — теж системи масового обслуговування. У теорії і практиці моделювання важливе місце посідають моделі систем масового обслуговування.

Кожна система масового обслуговування містить один або декілька обслуговуючих пристроїв, до яких можуть формуватись черги з вимог, що потребують обслуговування. Вимоги можуть називатись по-різному — повідомленнями, клієнтами, замовленнями, викликами тощо. Правила або алгоритми взаємодії пристроїв і вимог називають *правилами обслуговування й організації черги*.

Системи масового обслуговування вивчає теорія масового обслуговування (фахівці західних країн називають її теорією черг). Важливим типом завдань у теорії масового обслуговування є визначення місць накопичування вимог у системі обслуговування, наприклад, визначення необхідної кількості місць на стелажах на складі, кількості місць у палатах госпіталю та ін.

Найчастіше моделювання систем масового обслуговування здійснюють для визначення наступних показників:

- середньої кількості вимог, які система може обслужити за одиницю часу;
- імовірності того, що вимозі буде відмовлено в обслуговуванні;
- середнього часу очікування вимоги в черзі;
- середньої кількості вимог у черзі;
- коефіцієнту завантаження пристрою для обслуговування;
- середньої кількості пристроїв, зайнятих обслуговуванням;
- закону розподілу часу перебування вимог у системі;
- закону розподілу часу очікування;
- закону розподілу числа вимог у черзі.

Моделі систем масового обслуговування бувають аналітичними й імітаційними. Аналітичне моделювання систем масового обслуговування можливе лише в окремих випадках і описується у наступних параграфах цього розділу. Імітаційне моделювання цих систем буде розглянуто у двох наступних розділах.

Перші теоретичні результати для систем масового обслуговування було отримано датським вченим, співробітником телефонної компанії Копенгагена Агнером Крарупом Ерлангом на початку ХХ століття.

Система масового обслуговування характеризується вхідним потоком вимог, правилами постановки вимог у черги і вибору з них, правилами обслуговування і режимами роботи системи.

2. Характеристики потоку подій і вхідний потік вимог

Потоком подій називають послідовність однотипних ситуацій, які відбуваються одна за одною в які-небудь моменти часу. Прикладом потоку подій є поява покупців у магазині.

Найважливішою характеристикою потоку подій є *інтенсивність потоку* λ — середня кількість подій, які відбуваються за одиницю часу. Інтенсивність може бути як сталою ($\lambda = \text{const}$), так і залежною від часу t .

Потік подій, в якому інтервали часу τ_k між сусідніми подіями набувають заздалегідь відомих значень, називають *детермінованим*. Якщо при цьому інтервали однакові ($\tau_k = \tau$ для всіх $k = 1, 2, \dots$), то потік подій називають *регулярним*.

Потік подій, в якому інтервали часу τ_k між сусідніми подіями є випадковими величинами, називають *випадковим*. Потік подій називають *стаціонарним*, якщо його ймовірнісні характеристики не залежать від часу. Іншими словами, потік є стаціонарним, якщо ймовірність здійснення певної кількості подій за деякий проміжок часу τ залежить тільки від довжини цього проміжку τ і інтенсивності λ , і не залежить від місця цього проміжку на осі часу. Зокрема, інтенсивність стаціонарного потоку є сталою.

Потік подій є *потоком без післядії*, якщо ймовірність виникнення певної кількості подій за деякий проміжок часу τ не залежить від кількості і часу настання подій, які відбулись у системі раніше, тобто не залежить від передісторії. Прикладом потоку подій без післядії може бути потік моментів часу реєстрації елементарних частинок лічильником Гейгера.

Потік подій називають *ординарним*, якщо події в ньому з'являються поодиночі, а не групами по кілька подій одразу. Наприклад, ординарним є потік автомобілів повз спостерігача по дорозі з однорядним рухом. У той самий час, через турнікети в метро може зайти одночасно кілька людей. Цей потік не є ординарним.

Стаціонарним пуассонівським або найпростішим потоком називають потік, який має властивості стаціонарності, ординарності і відсутності післядії.

Якщо подія в потоці полягає в появі вимоги, то маємо потік вимог. Для визначення вхідного потоку вимог потрібно задати моменти часу їхнього надходження до системи (закон надходження) і кількість вимог, які можуть надійти одночасно. Закон надходження може бути детермінованим (вимоги надходять у систему у фіксовані моменти часу) або ймовірнісним (проміжки часу між вимогами є випадковими величинами з деяким законом розподілу).

Для визначення вхідного потоку вимог, крім закону надходження, потрібно задати кількість вимог, які приходять одночасно. Системи, в яких вимоги з'являються не поодиночі, а групами (пакетами), називають системами з груповим надходженням вимог.

Стаціонарний пуассонівський (найпростіший) потік вимог відіграє серед інших потоків особливу роль. Це пов'язано з тим, що при накладанні достатньо великої кількості незалежних потоків отримується потік, близький до пуассонівського. На практиці вважають, що вхідний потік є пуассонівським, якщо вимоги надходять до системи від великої кількості незалежних джерел. Прикладами таких потоків можуть бути дзвінки абонентів у телефонній мережі або проїзд автомобілів повз спостерігача по одній смузі руху.

3. Моделювання стаціонарного пуассонівського потоку

Покажемо тепер, як при моделюванні систем масового обслуговування можна задати стаціонарний пуассонівський потік вимог.

Ймовірність того, що за час t надійде k вимог, які утворюють стаціонарний пуассонівський потік з інтенсивністю λ , є випадковою величиною з законом розподілу Пуассона і дорівнює

$$P_k(t) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}, \quad k \geq 0, \quad t \geq 0. \quad (7.1)$$

Визначимо, який розподіл мають проміжки часу τ між моментами надходження двох сусідніх вимог у стаціонарному пуассонівському потоці.

Функція розподілу $F(t)$ проміжків часу між подіями в стаціонарному пуассонівському потоці визначає ймовірність того, що випадкова величина τ набуде значення, меншого від t , тобто $F(t) = P(\tau < t)$.

Тоді $F(t) = 1 - P_0$, де P_0 — ймовірність протилежної події, тобто ймовірність того, що за проміжок часу t до системи не надійде жодна вимога. Значення ймовірності P_0 знаходимо за формулою (7.1) для $k = 0$:

$$P_0 = \frac{(\lambda t)^0}{0!} e^{-\lambda t} = e^{-\lambda t}.$$

Отже, функція розподілу випадкової величини τ має вигляд:

$$F(t) = 1 - e^{-\lambda t}, \quad t \geq 0.$$

Ми отримали функцію розподілу експоненціального закону розподілу. Відповідна функція щільності $f(t)$ знаходиться диференціюванням:

$$f(t) = \lambda e^{-\lambda t}, \quad t \geq 0.$$

Таким чином, приходимо до висновку: *проміжки часу між подіями в стаціонарному пуассонівському потоці розподілені за експоненціальним законом розподілу.*

4. Правила обслуговування, організація черги і типи моделей систем масового обслуговування

Правила обслуговування вимог характеризуються часом (розподілом часу) обслуговування, кількістю вимог, які обслуговуються одночасно, і дисципліною обслуговування.

У системі масового обслуговування можуть використовуватись *одноканальні* і *багатоканальні* пристрої. В одноканальному пристрої може обслуговуватись не більше однієї вимоги у кожен конкретний момент часу. Натомість у багатоканальному пристрої може одночасно обслуговуватись кілька вимог. Час обслуговування може бути детермінованим або визначатись випадковою величиною з деяким законом розподілу.

Розрізняють безпріоритетні і пріоритетні дисципліни обслуговування. Пріоритет буває відносним і абсолютним. Відносний пріоритет визначає лише порядок перебування вимог у черзі. Вимоги з однаковими пріоритетами можуть утворювати черги. У випадку абсолютного пріоритету вимога з вищим пріоритетом перериває обслуговування вимоги з нижчим пріоритетом. Вимога, обслуговування якої було перерване, може дообслуговуватись на тому самому пристрої після завершення обслуговування вимоги з вищим пріоритетом, проте вона може також втрачати право на дообслуговування.

Система масового обслуговування називається *багатофазною*, якщо вимоги послідовно обслуговуються в кількох пристроях (наприклад, конвеєр). У протилежному разі її називають *однофазною*.

Перед пристроями обслуговування можуть формуватись черги, які можуть функціонувати за різними принципами.

Класична черга функціонує за принципом «*першим прийшов — першим обслуговують*», в англійській літературі його часто називають FIFO (First In First Out). Прикладом такої черги є звичайна черга в магазині.

Існує також інший спосіб організації черги за принципом «*останнім прийшов — першим обслуговують*», його ще називають LIFO (Last In First Out). Таку чергу часто називають стеком. Прикладом такої черги є пором: автомобіль, який заїхав на пором першим, виїде з нього останнім.

Черга може функціонувати з урахуванням пріоритету: вимоги з одним пріоритетом формують звичайну чергу, вимоги з вищим пріоритетом розміщуються в голові черги. Пріоритет вимог може з часом змінюватись. У цьому випадку його називають динамічним на відміну від статичного, який не змінюється.

Крім того, черга може функціонувати за деяким нестандартним принципом, наприклад, першими можуть обслуговуватись вимоги, які потребують найменшого часу обслуговування, або вимоги з найбільшим часом перебування в багатофазній системі масового обслуговування. Вимоги з черги можуть також вибиратись випадковим чином, або з черги будуть вибиратись лише ті вимоги, які задовольняють певні умови. Зокрема, вимоги можуть вибиратись за певним розкладом (планом). Наприклад,

якщо вимоги належать до трьох класів, розклад може мати вигляд: (1, 2, 3, 1, 2, 1, 3), тобто спочатку береться одна вимога першого класу, потім — другого і т. д. Розклад може бути прив'язаним до часу.

Черга може мати обмеження на довжину. Наприклад, якщо у черзі перебуває п'ять вимог при такій максимально можливій довжині черги, то при появі нової вимоги, їй буде відмовлено в обслуговуванні, бо вже досягнута максимально можлива довжина черги. Прикладом системи з обмеженою довжиною черги є комп'ютерна система з обмеженим буфером для зберігання вхідних повідомлень.

Може також накладатись обмеження на час перебування вимоги в черзі. Наприклад, якщо час перебування вимоги в черзі обмежений десятьма хвилинами, то при досягненні вимогою цього часу, вона взагалі вилучається з черги і отримує відмову в обслуговуванні.

Черга в системі масового обслуговування може не створюватись. У цьому випадку частина вимог отримує відмову в обслуговуванні.

Для позначення моделей систем масового обслуговування часто використовують запис $X/Y/n/m$, де X — розподіл часу надходження вимог, Y — розподіл часу обслуговування, n — кількість каналів для обслуговування в багатоканальному пристрої, а m — максимальна можлива довжина черги (відсутність m означає, що черга може мати необмежену довжину). Замість букв X і Y найчастіше використовують наступні позначення: D — детермінований потік вимог або детермінований час обслуговування, M — стаціонарний пуассонівський потік вимог або час обслуговування, заданий експоненціальним законом розподілу, G — довільний вхідний потік вимог або будь-який закон розподілу часу обслуговування.

З допомогою теорії систем масового обслуговування аналітичне моделювання можна здійснити лише у випадках $M/M/1/m$, $D/D/1/m$, $M/G/1/m$ і $M/M/n/m$, де $n = 1, 2, \dots$, $m = 0, 1, 2, \dots, \infty$. В інших випадках для визначення показників функціонування систем масового обслуговування треба використовувати імітаційне моделювання.

5. Формула Літла

У теорії масового обслуговування важливе значення має формула Літла, яку ще називають *законом стаціонарної черги*. Для виведення формули Літла розглянемо довільну систему масового обслуговування з очікуванням.

Нехай $\alpha(t)$ — вхідний потік вимог, який розглядається на проміжку $[0, t]$. Через $\beta(t)$ позначимо вихідний потік вимог на цьому ж проміжку. Побудуємо графіки кусково сталих функцій $\alpha(t)$ і $\beta(t)$ (рис. 7.1).

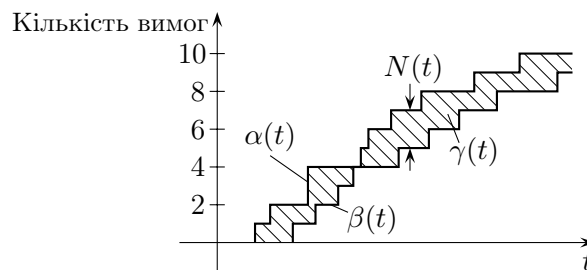


Рис. 7.1.

Кількість вимог, які знаходяться в системі у конкретний момент часу t , можна знайти за формулою $N(t) = \alpha(t) - \beta(t)$. Площа $\gamma(t)$ заштрихованої фігури між лініями $\alpha(t)$ і $\beta(t)$ визначає загальний час, проведений всіма вимогами в системі.

Інтенсивність надходження вимог до системи масового обслуговування за час спостереження $[0, t]$ визначається формулою

$$\lambda_t = \frac{\alpha(t)}{t}, \quad (7.2)$$

середній час перебування вимог у системі за той самий проміжок часу

$$T_t = \frac{\gamma(t)}{\alpha(t)}, \quad (7.3)$$

а середня кількість вимог, що перебували в системі за проміжок часу $[0, t]$,

$$\bar{N}_t = \frac{\gamma(t)}{t}. \quad (7.4)$$

Використовуючи вирази (7.2) – (7.4), отримуємо формулу

$$\bar{N}_t = \lambda_t T_t. \quad (7.5)$$

Припустимо, що існують границі $\lambda = \lim_{t \rightarrow \infty} \lambda_t$ і $T = \lim_{t \rightarrow \infty} T_t$, де λ — інтенсивність надходження вимог, а T — середній час перебування вимог у системі. У такому разі існує також границя для середньої кількості вимог, що знаходяться у системі, тобто $\bar{N} = \lim_{t \rightarrow \infty} \bar{N}_t$. Тоді з формули (7.5) отримуємо *формулу Літтла*:

$$\bar{N} = \lambda T.$$

Отже, для будь-якого вхідного потоку вимог і будь-якого розподілу часу їхнього обслуговування середня кількість вимог, що перебуває в системі, дорівнює добутку інтенсивності вхідного потоку вимог і середнього часу перебування їх у системі.

Формула Літтла має місце також для черги.

Лекція 8. Показники функціонування систем масового обслуговування

План

1. Багатоканальні системи масового обслуговування з відмовами.
2. Багатоканальні системи масового обслуговування з чергою довільної довжини.

1. Багатоканальні системи масового обслуговування з відмовами

Розглянемо багатоканальну систему масового обслуговування типу $M/M/n/0$. Ця система має n каналів, якщо зайняті всі канали, то вимога отримує відмову в обслуговуванні, а черга не формується. Прикладом такої системи є автостоянка. Якщо на ній всі місця зайняті, то водій поїде шукати поблизу іншу стоянку. Інший приклад — телефонна система: якщо всі лінії зайняті, то телефонний дзвінок не обслуговується.

Зробимо наступні припущення:

- вхідний потік є стаціонарним пуассонівським з інтенсивністю λ ;
- всі канали є однотипними;
- час обслуговування вимоги в каналі є випадковим і утворює стаціонарний пуассонівський потік з інтенсивністю μ .

Система масового обслуговування має $n + 1$ станів: S_0 — всі канали вільні, S_1 — зайнято один канал (не важливо, який саме), S_2 — зайнято два канали (не важливо, які саме) і т. д., S_n — всі канали зайнято. Якщо система перебуває в стані S_n , то чергова вимога не буде обслуговуватись.

Оскільки вхідний потік є пуассонівським, а час обслуговування утворює пуассонівський потік, то система може переходити з одного стану в інші послідовно: з S_0 в S_1 , з S_1 в S_0 або в S_2 і т. д. (рис. 8.1).

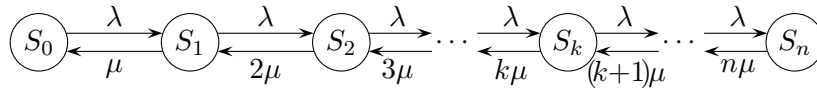


Рис. 8.1.

Зі стану S_k в стан S_{k-1} існує потік з інтенсивністю $k\mu$ ($k = 2, 3, \dots, n$), бо є k зайнятих каналів і кожен з них може звільнитись відповідно до потоку з інтенсивністю μ , причому ці події незалежні. Отже, вихідний потік також є стаціонарним пуассонівським з інтенсивністю $\lambda^* = \min\{\lambda, n\mu\}$.

Для моделювання систем з пуассонівськими потоками використовують системи диференціальних рівнянь Колмогорова, в яких шукані функції, залежні від часу t , є ймовірностями станів.

Нехай $p_i(t)$ — ймовірність i -го стану, тобто ймовірність того, що в момент часу t система перебуває в стані S_i , $i = 0, 1, 2, \dots, n$. Для ймовірностей станів має виконуватись умова

$$\sum_{i=0}^n p_i(t) = 1. \quad (8.1)$$

Розглянемо спочатку ймовірність $p_0(t)$ того, що в момент часу t система перебуває в стані S_0 (тобто всі канали вільні). Надамо t малий приріст Δt і знайдемо $p_0(t + \Delta t)$ — ймовірність того, що в момент часу $t + \Delta t$ система буде в стані S_0 . Це можливо, очевидно, при реалізації одного з двох сценаріїв: або в момент часу t система вже була в стані S_0 і за час Δt не вийшла з нього, або в момент часу t система була в стані S_1 , а за час Δt перейшла з нього в стан S_0 . Оскільки проміжок часу Δt дуже малий (потім ми навіть спрямуємо його до нуля), а всі потоки є пуассонівськими (зокрема, ординарними), то за цей час система не могла перейти зі стану S_2 в стан S_0 .

Знайдемо ймовірність розвитку подій по першому сценарію. Система знаходиться в стані S_0 . У цьому випадку можливий перехід у стан S_1 , ймовірність якого $\lambda\Delta t$. Тоді ймовірність того, що система залишиться в стані S_0 дорівнює $1 - \lambda\Delta t$. Оскільки ймовірність опинитись у стані S_0 у момент часу t дорівнює $p_0(t)$, а наступні переходи не залежать від того, яким чином система прийшла до цього стану, то ймовірність першого сценарію дорівнює добутку ймовірностей $p_0(t)(1 - \lambda\Delta t)$.

Міркуючи аналогічно, отримуємо ймовірність другого сценарію $p_1(t)\mu\Delta t$. Оскільки перший і другий сценарії є несумісними (реалізується тільки один з них), то ймовірність того, що система знаходиться у момент часу $t + \Delta t$ в стані S_0 , дорівнює

$$p_0(t + \Delta t) = p_0(t)(1 - \lambda\Delta t) + p_1(t)\mu\Delta t,$$

звідки

$$\frac{p_0(t + \Delta t) - p_0(t)}{\Delta t} = \mu p_1(t) - \lambda p_0(t).$$

Спрямовуючи Δt до нуля і переходячи до границі, отримуємо диференціальне рівняння для $p_0(t)$:

$$\frac{dp_0}{dt} = \mu p_1 - \lambda p_0.$$

Розглянемо тепер $p_1(t)$ — ймовірність того, що система в момент часу t перебуває в стані S_1 . Знайдемо ймовірність того, що система перебуватиме в стані S_1 у момент часу $t + \Delta t$. Це може відбутись, якщо система перебувала в цьому стані і в час t . Крім того, це може трапитись, якщо система перебувала в момент часу t в стані S_0 чи S_2 . Оскільки зі стану S_1 система могла перейти в стан S_0 чи S_2 (що є несумісними подіями), то ймовірність того, що система була і залишається в стані S_1 , дорівнює $p_1(t)(1 - (\lambda + \mu)\Delta t)$. Ймовірність переходу за час Δt зі стану S_0 в стан S_1 дорівнює $p_0(t)\lambda\Delta t$, а ймовірність переходу за той самий час зі стану S_2 в стан S_1 дорівнює $p_2(t) \cdot 2\mu\Delta t$. Отже, ми маємо:

$$\begin{aligned} p_1(t + \Delta t) &= p_1(t)(1 - (\lambda + \mu)\Delta t) + p_0(t)\lambda\Delta t + 2p_2(t)\mu\Delta t \Rightarrow \\ \frac{p_1(t + \Delta t) - p_1(t)}{\Delta t} &= p_0(t)\lambda + 2p_2(t)\mu - (\lambda + \mu)p_1(t) \Rightarrow \\ \frac{dp_1}{dt} &= \lambda p_0 + 2\mu p_2 - (\lambda + \mu)p_1. \end{aligned}$$

Аналогічно складаються й інші рівняння, внаслідок чого отримуємо систему диференціальних рівнянь Колмогорова

$$\begin{cases} \frac{dp_0}{dt} = \mu p_1 - \lambda p_0, \\ \frac{dp_1}{dt} = \lambda p_0 + 2\mu p_2 - (\lambda + \mu)p_1, \\ \frac{dp_2}{dt} = \lambda p_1 + 3\mu p_3 - (\lambda + 2\mu)p_2, \\ \dots \\ \frac{dp_{n-1}}{dt} = \lambda p_{n-2} + n\mu p_n - (\lambda + (n-1)\mu)p_{n-1}, \\ \frac{dp_n}{dt} = \lambda p_{n-1} - n\mu p_n. \end{cases} \quad (8.2)$$

Замість будь-якого з рівнянь у цій системі можна покласти умову (8.1) (наприклад, замість останнього).

Початкові умови для системи диференціальних рівнянь (8.2) легко складаються, якщо відомий початковий стан системи масового обслуговування. Нехай, наприклад, початковим був стан S_0 . Тоді $p_0(0) = 1$, $p_1(0) = 0$, $p_2(0) = 0$, $\dots$, $p_n(0) = 0$.

Отриману задачу Коші для лінійної системи диференціальних рівнянь зі сталими коефіцієнтами (8.2) завжди можна однозначно розв'язати і знайти ймовірності станів у кожен момент часу. Типові графіки для ймовірностей станів для $n = 2$ показані на рис. 8.2.

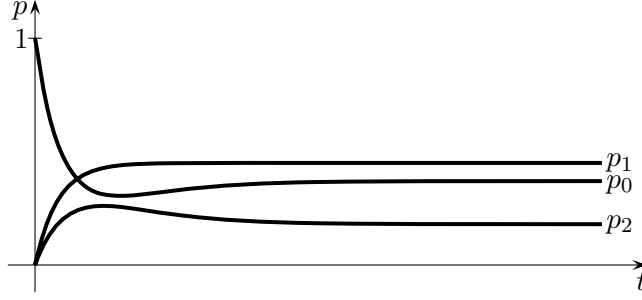


Рис. 8.2.

У теорії випадкових процесів доводиться, що якщо кількість станів є скінченною і з кожного з них можна перейти в інший за скінченну кількість кроків, то існують границі ймовірностей $p_k(t)$ при $t \rightarrow \infty$, які не залежать від початкового стану. Їх називають *фінальними ймовірностями*¹.

Фінальні ймовірності позначатимемо тими самими буквами $p_1, p_2, \dots, p_n$, що й ймовірності станів, але тепер їх трактуватимемо як сталі числа.

Для фінальних ймовірностей з системи диференціальних рівнянь (8.2) і умови (8.1) отримуємо систему лінійних алгебричних рівнянь

$$\begin{cases} \mu p_1 - \lambda p_0 = 0, \\ \lambda p_0 + 2\mu p_2 - (\lambda + \mu)p_1 = 0, \\ \lambda p_1 + 3\mu p_3 - (\lambda + 2\mu)p_2 = 0, \\ \dots \\ \lambda p_{n-2} + n\mu p_n - (\lambda + (n-1)\mu)p_{n-1} = 0, \\ p_0 + p_1 + \dots + p_n = 1. \end{cases} \quad (8.3)$$

З перших n рівнянь цієї системи, використовуючи метод математичної індукції, знаходимо:

$$\begin{aligned} p_1 &= \frac{\lambda}{\mu} p_0, \\ p_2 &= \frac{(\lambda + \mu) \frac{\lambda}{\mu} p_0 - \lambda p_0}{2\mu} = \frac{\lambda^2}{2\mu^2} p_0, \\ p_3 &= \frac{(\lambda + 2\mu) \frac{\lambda^2}{2\mu^2} p_0 - \lambda \frac{\lambda}{\mu} p_0}{3\mu} = \frac{\lambda^3}{3 \cdot 2\mu^3} p_0, \\ &\dots \\ p_{k-1} &= \frac{\lambda^{k-1}}{(k-1)! \mu^{k-1}} p_0, \\ p_k &= \frac{(\lambda + (k-1)\mu) \frac{\lambda^{k-1}}{(k-1)! \mu^{k-1}} p_0 - \lambda \frac{\lambda^{k-2}}{(k-2)! \mu^{k-2}} p_0}{k\mu} = \frac{\lambda^k}{k! \mu^k} p_0, \\ &\dots \\ p_n &= \frac{\lambda^n}{n! \mu^n} p_0. \end{aligned}$$

Нехай $\alpha = \frac{\lambda}{\mu}$ — *зведена інтенсивність вимог*, вона характеризує середню кількість вимог, що надійшли протягом середнього часу обслуговування вимоги. Підставляючи отримані вирази для фінальних ймовірностей в останнє рівняння системи (8.3) (тобто умову (8.1)), отримуємо:

$$p_0 \left(1 + \alpha + \frac{\alpha^2}{2!} + \frac{\alpha^3}{3!} + \dots + \frac{\alpha^n}{n!} \right) = 1,$$

¹В деяких книгах фінальні ймовірності називають граничними ймовірностями.

звідки

$$p_0 = \left(1 + \alpha + \frac{\alpha^2}{2!} + \frac{\alpha^3}{3!} + \dots + \frac{\alpha^n}{n!}\right)^{-1} = \left(\sum_{k=0}^n \frac{\alpha^k}{k!}\right)^{-1}. \quad (8.4)$$

Імовірності решти станів подаються формулами:

$$p_k = \frac{\alpha^k}{k!} p_0, \quad k = 1, 2, \dots, n. \quad (8.5)$$

Співвідношення (8.4), (8.5) називають *формулами Ерланга*.

Якщо всі канали зайняті, то чергова вимога не буде обслуговуватись системою й отримає відмову. Тому *ймовірність відмови системи*

$$P_{\text{відм}} = p_n = \frac{\alpha^n}{n!} p_0. \quad (8.6)$$

Імовірність обслуговування вимоги

$$P_{\text{обсл}} = 1 - P_{\text{відм}} = 1 - \frac{\alpha^n}{n!} p_0. \quad (8.7)$$

Пропускна здатність A — це середня кількість вимог, обслужених системою за одиницю часу:

$$A = \lambda P_{\text{обсл}}. \quad (8.8)$$

Знайдемо *середню кількість зайнятих каналів*:

$$\begin{aligned} N_3 &= p_1 + 2p_2 + \dots + np_n = \\ &= \sum_{k=1}^n k \frac{\alpha^k}{k!} p_0 = \alpha p_0 \sum_{k=1}^n \frac{\alpha^{k-1}}{(k-1)!} = \alpha p_0 \sum_{k=0}^{n-1} \frac{\alpha^k}{k!} = \\ &= \alpha \left(\sum_{k=0}^n \frac{\alpha^k}{k!} \right)^{-1} \cdot \left(\sum_{k=0}^n \frac{\alpha^k}{k!} - \frac{\alpha^n}{n!} \right) \Rightarrow \\ N_3 &= \alpha(1 - p_n). \end{aligned} \quad (8.9)$$

Коефіцієнт завантаження одного каналу

$$K_3 = \frac{N_3}{n}. \quad (8.10)$$

Зокрема, для $n = 1$ маємо:

$$P_{\text{відм}} = N_3 = K_3 = \frac{\alpha}{1 + \alpha}.$$

Приклад. У перукарні є три перукарі. Час обслуговування клієнта кожним з них є випадковою величиною з експоненціальним законом розподілу з середнім значенням 10 хвилин. Клієнти часом своєї появи утворюють стаціонарний пуассонівський потік з інтенсивністю $\lambda = 1/5$. Якщо всі перукарі зайняті, то клієнт іде в іншу перукарню, яка знаходиться поруч. Знайти ймовірність того, що клієнт піде в іншу перукарню, пропускну здатність системи, середню кількість зайнятих перукарів, коефіцієнт завантаження перукаря.

Розв'язання. З курсу теорії ймовірностей відомо, що інтенсивність потоку подій λ для експоненціального закону розподілу проміжків між подіями є оберненою величиною до середнього значення.

Тому в цій задачі $\lambda = 1/5$, $\mu = 1/10$, $n = 3$. Отже, зведена інтенсивність $\alpha = 2$, а за формулами (8.4)–(8.10) маємо:

$$\begin{aligned} p_0 &= \left(1 + 2 + \frac{2^2}{2!} + \frac{2^3}{3!}\right)^{-1} = \frac{3}{19}, \\ p_1 &= \frac{6}{19}, \quad p_2 = \frac{6}{19}, \quad p_3 = \frac{4}{19} \Rightarrow \\ P_{\text{відм}} &= \frac{4}{19} \approx 0,21, \quad A = \frac{3}{19} \approx 0,16, \\ N_3 &= \frac{6}{19} + 2 \cdot \frac{6}{19} + 3 \cdot \frac{4}{19} = \frac{30}{19} \approx 1,58, \quad K_3 = \frac{10}{19} \approx 0,53. \blacksquare \end{aligned}$$

2. Багатоканальні системи масового обслуговування з чергою довільної довжини

Розглянемо багатоканальну систему масового обслуговування типу $M/M/n/\infty$ з безліччю позицій для очікування в черзі. Якщо зайняті всі n каналів, то вимога стає у хвіст черги. Прикладом системи може бути торговельна точка або частина виробництва.

Зробимо наступні припущення:

- вхідний потік є стаціонарним пуассонівським з інтенсивністю λ ;
- всі канали є однотипними;
- час обслуговування вимоги в каналі є випадковим і утворює стаціонарний пуассонівський потік з інтенсивністю μ .

Система масового обслуговування може мати, взагалі кажучи, безмежну, але зліченну кількість станів: S_0 — всі канали вільні, S_1 — зайнято один канал (не важливо, який саме), S_2 — зайнято два канали (не важливо, які саме) і т. д., S_n — всі канали зайнято, але черга є порожньою, S_{n+k} — всі канали зайнято, а в черзі перебуває k вимог.

Оскільки вхідний потік є пуассонівським, а час обслуговування утворює пуассонівський потік, то система може переходити з одного стану в інші послідовно: з S_0 в S_1 , з S_1 в S_0 або в S_2 і т. д. (рис. 8.3).

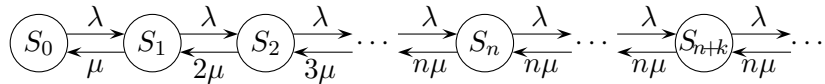


Рис. 8.3.

Вихідний потік тут також є стаціонарним пуассонівським з інтенсивністю $\lambda^* = \min\{\lambda, n\mu\}$.

Нехай $p_i(t)$ — ймовірність i -го стану, тобто ймовірність того, що в момент часу t система перебуває в стані S_i , $i = 0, 1, 2, \dots$. Для ймовірностей станів має виконуватись умова

$$\sum_{i=0}^{\infty} p_i(t) = 1. \quad (8.11)$$

Аналогічно попередньому випадку складається система диференціальних рівнянь Колмогорова

$$\begin{cases} \frac{dp_0}{dt} = \mu p_1 - \lambda p_0, \\ \frac{dp_1}{dt} = \lambda p_0 + 2\mu p_2 - (\lambda + \mu)p_1, \\ \frac{dp_2}{dt} = \lambda p_1 + 3\mu p_3 - (\lambda + 2\mu)p_2, \\ \dots \\ \frac{dp_n}{dt} = \lambda p_{n-1} + n\mu p_{n+1} - (\lambda + n\mu)p_n, \\ \dots \\ \frac{dp_{n+k}}{dt} = \lambda p_{n+k-1} + n\mu p_{n+k+1} - (\lambda + n\mu)p_{n+k}, \\ \dots \end{cases} \quad (8.12)$$

Якщо початковим станом є стан S_0 , то маємо початкові умови: $p_0(0) = 1$, $p_k(0) = 0$, $k = 1, 2, 3, \dots$

Припустимо, що ймовірності станів $p_k(t)$ при $t \rightarrow \infty$ прямують до *фінальних ймовірностей* p_k , $k = 0, 1, 2, \dots$

Для фінальних ймовірностей з системи диференціальних рівнянь (8.12) отримуємо систему лінійних алгебричних рівнянь

$$\begin{cases} \mu p_1 - \lambda p_0 = 0, \\ \lambda p_0 + 2\mu p_2 - (\lambda + \mu)p_1 = 0, \\ \lambda p_1 + 3\mu p_3 - (\lambda + 2\mu)p_2 = 0, \\ \dots \\ \lambda p_{n-1} + n\mu p_{n+1} - (\lambda + n\mu)p_n = 0, \\ \dots \\ \lambda p_{n+k-1} + n\mu p_{n+k+1} - (\lambda + n\mu)p_{n+k} = 0, \\ \dots \end{cases} \quad (8.13)$$

Нехай $\alpha = \frac{\lambda}{\mu}$ — *зведена інтенсивність вимог*. Тоді, враховуючи, що система (8.13) відрізняється від системи (??) лише тим, що має більшу кількість однотипних рівнянь, і повторюючи міркування § ??, отримуємо:

$$p_k = \frac{\alpha^k}{k!} p_0, \quad k = 1, 2, \dots, n. \quad (8.14)$$

$$p_{n+k} = \frac{\alpha^k}{n^k} \frac{\alpha^n}{n!} p_0, \quad k = 1, 2, \dots \quad (8.15)$$

Підставляючи отримані вирази для фінальних ймовірностей в умову нормування (8.11), маємо:

$$p_0 \left(1 + \alpha + \frac{\alpha^2}{2!} + \frac{\alpha^3}{3!} + \dots + \frac{\alpha^{n-1}}{(n-1)!} + \frac{\alpha^n}{n!} \left(1 + \frac{\alpha}{n} + \frac{\alpha^2}{n^2} + \dots \right) \right) = 1. \quad (8.16)$$

Безмежна сума в дужках у формулі (8.16) утворює геометричну прогресію зі знаменником $\frac{\alpha}{n}$. Геометрична прогресія має скінченну суму, якщо її знаменник

$$\frac{\alpha}{n} < 1. \quad (8.17)$$

Тоді

$$\begin{aligned} p_0 &= \left(1 + \alpha + \frac{\alpha^2}{2!} + \dots + \frac{\alpha^{n-1}}{(n-1)!} + \frac{\alpha^n}{n!} \frac{1}{1 - \frac{\alpha}{n}} \right)^{-1} \Rightarrow \\ p_0 &= \left(1 + \alpha + \frac{\alpha^2}{2!} + \dots + \frac{\alpha^{n-1}}{(n-1)!} + \frac{\alpha^n}{(n-1)!(n-\alpha)} \right)^{-1}. \end{aligned} \quad (8.18)$$

Припустимо, що виконується умова (8.17). Тоді *середня кількість зайнятих каналів*

$$\begin{aligned} N_3 &= p_1 + 2p_2 + \dots + np_n + np_{n+1} + \dots + np_{n+k} + \dots \Rightarrow \\ N_3 &= p_0 \left(\alpha + 2\frac{\alpha^2}{2!} + 3\frac{\alpha^3}{3!} + \dots + n\frac{\alpha^n}{n!} + \right. \\ &\quad \left. + n\frac{\alpha^n}{n!} \frac{\alpha}{n} \left(1 + \frac{\alpha}{n} + \frac{\alpha^2}{n^2} + \dots \right) \right) \Rightarrow \\ N_3 &= p_0 \alpha \left(1 + \alpha + \frac{\alpha^2}{2!} + \dots + \frac{\alpha^{n-1}}{(n-1)!} + \frac{\alpha^n}{(n-1)!(n-\alpha)} \right) \Rightarrow \\ N_3 &= \alpha. \end{aligned} \quad (8.19)$$

Коефіцієнт завантаження одного каналу

$$K_3 = \frac{\alpha}{n}. \quad (8.20)$$

Для виведення середньої кількості вимог у черзі скористаємось формулою

$$\sum_{k=1}^m kq^k = \frac{q^{m+1}(mq - m - 1) + q}{(1 - q)^2}, \quad q \neq 1,$$

з якої для $|q| < 1$ випливає, що

$$\sum_{k=1}^{\infty} kq^k = \frac{q}{(1 - q)^2}.$$

Тоді

$$\begin{aligned} L_{\text{ч}} &= p_{n+1} + 2p_{n+2} + \dots + kp_{n+k} + \dots \Rightarrow \\ L_{\text{ч}} &= \frac{\alpha^n}{n!} \left(\frac{\alpha}{n} + 2\frac{\alpha^2}{n^2} + 3\frac{\alpha^3}{n^3} + \dots + k\frac{\alpha^k}{n^k} + \dots \right) p_0 = \\ &= \frac{\alpha^n}{n!} \frac{\frac{\alpha}{n}}{\left(1 - \frac{\alpha}{n}\right)^2} p_0 \Rightarrow \\ L_{\text{ч}} &= \frac{\alpha^{n+1}}{(n-1)!(n-\alpha)^2} p_0. \end{aligned} \quad (8.21)$$

Згідно з формулою Літгла середній час очікування в черзі

$$W_{\text{ч}} = \frac{L_{\text{ч}}}{\lambda}. \quad (8.22)$$

Зокрема, для $n = 1$ при виконанні умови $\alpha < 1$ маємо:

$$p_0 = 1 - \alpha, \quad N_3 = K_3 = \alpha, \quad L_{\text{ч}} = \frac{\alpha^2}{1 - \alpha}.$$

Якщо $\frac{\alpha}{n} \geq 1$, то з формули (8.16) випливає, що $p_0 = 0$ і, отже, $p_k = 0$ для кожного скінченного k . Це означає, що у випадку $\frac{\alpha}{n} \geq 1$ довжина черги і середній час очікування в черзі при $t \rightarrow \infty$ прямують до нескінченності, коефіцієнт завантаження одного каналу прямує до одиниці, а середня кількість зайнятих каналів — до n .

Приклад. У перукарні є три перукарі. Час обслуговування клієнта кожним з них є випадковою величиною з експоненціальним законом розподілу з середнім значенням 10 хвилин. Клієнти часом своєї появи утворюють стаціонарний пуассонівський потік з інтенсивністю $\lambda = 1/5$. Знайти середню кількість зайнятих перукарів, коефіцієнт завантаження перукаря, середню довжину черги і середній час очікування в черзі.

Розв'язання. З курсу теорії ймовірностей відомо, що інтенсивність потоку подій λ для експоненціального закону розподілу проміжків між подіями є оберненою величиною до середнього значення. Тому в цій задачі $\lambda = 1/5$, $\mu = 1/10$, $n = 3$. Отже, зведена інтенсивність $\alpha = 2$, а оскільки $\frac{\alpha}{n} = \frac{2}{3} < 1$, то черга є скінченною і за формулами (8.18)–(8.22) маємо:

$$\begin{aligned} p_0 &= \left(1 + 2 + \frac{2^2}{2!} + \frac{2^3}{2!(3-2)} \right)^{-1} = \frac{1}{9}, \\ N_3 &= 2, \quad K_3 = \frac{2}{3} \approx 0,667, \\ L_{\text{ч}} &= \frac{2^4}{2!(3-2)^2} \frac{1}{9} = \frac{8}{9} \approx 0,889, \quad W_{\text{ч}} = 5L_{\text{ч}} = \frac{40}{9} \approx 4,44. \quad \blacksquare \end{aligned}$$

Лекція 9. Основи операційного аналізу мереж систем масового обслуговування

План

1. Поняття про операційний аналіз мереж систем масового обслуговування.
2. Операційні змінні.
3. Операційні залежності.

1. Поняття про операційний аналіз мереж систем масового обслуговування

У попередніх параграфах розглядалися моделі простих систем масового обслуговування. Проте зустрічаються складніші структури — мережі систем масового обслуговування, вузлами яких є прості системи масового обслуговування. Найпростіша мережа утворюється шляхом послідовного з'єднання кількох систем масового обслуговування. Таку мережу називають багатофазною. Разом з тим, існують більш складні мережі. На рис. 9.1 наведено приклад мережі, що складається з чотирьох вузлів. Зовнішнє середовище позначено нулем. Стрілки вказують напрям руху вимог.

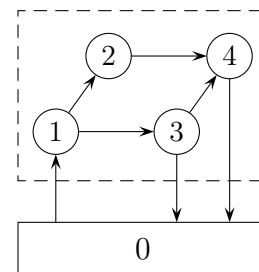


Рис. 9.1.

Операційний аналіз мереж систем масового обслуговування можна застосовувати до мереж, для яких існує баланс потоків вимог: кількість вимог, які залишають мережу протягом певного проміжку часу, має дорівнювати кількості вимог, що потрапили в мережу за цей самий проміжок часу. Це означає, що операційний аналіз можна застосовувати для дослідження мереж, які працюють з перевантаженням, тобто в них завжди є черга вимог. У цьому разі можна вважати, що замість вимоги, яка залишила мережу, до неї надходить інша вимога.

Крім того, для застосування операційного аналізу до мереж систем масового обслуговування припускають, що процес надходження вимог від одного вузла до іншого не повинен залежати від довжини черг у вузлах і часу закінчення обслуговування вимог у них.

Операційний аналіз часто використовують для визначення коефіцієнтів завантаження вузлів мережі та аналізу вузьких місць у ній.

2. Операційні змінні

Введемо основні операційні змінні, значення яких можливо отримати шляхом безпосереднього вимірювання параметрів реальної системи:

- q_{0j} — імовірність надходження вимог з зовнішнього середовища до j -го вузла мережі, $j = 1, 2, \dots, n$, де n — загальна кількість вузлів;
- q_{kj} — імовірність надходження вимог від k -го до j -го вузла мережі, $k, j = 1, 2, \dots, n$;
- q_{k0} — імовірність того, що після закінчення обслуговування у k -му вузлі вимоги залишать мережу, $k = 1, 2, \dots, n$;
- A_k — кількість вимог, які надійшли до k -го вузла за час спостереження за системою, $k = 1, 2, \dots, n$;
- A_{0j} — кількість вимог, які надійшли з зовнішнього середовища до j -го вузла за час спостереження за системою, $j = 1, 2, \dots, n$;
- C_{kj} — кількість вимог, які залишили k -й вузол і перейшли до j -го вузла мережі за час спостереження за системою, $k, j = 1, 2, \dots, n$;
- C_{k0} — кількість вимог, які залишили k -й вузол і перейшли до зовнішнього середовища за час спостереження за системою, $k = 1, 2, \dots, n$;
- T — загальний час спостереження за системою;
- B_k — загальний час обслуговування вимог у k -му вузлі, $k = 1, 2, \dots, n$ (вузол вважається зайнятим, якщо в ньому перебуває хоч одна вимога).

Як видно, зовнішнє середовище тут позначається нулем. Введемо додаткові операційні змінні:

$$C_k = \sum_{j=0}^n C_{kj}, \quad A_0 = \sum_{j=1}^n A_{0j}, \quad C_0 = \sum_{k=1}^n C_{k0},$$

де C_k — кількість вимог, які залишили k -й вузол, A_0 — кількість вимог, які надійшли до вузлів мережі із зовнішнього середовища, C_0 — кількість вимог, які залишили мережу.

Крім того, розглядають наступні додаткові операційні змінні:

– інтенсивність надходження вимог до k -го вузла

$$\lambda_k = \frac{A_k}{T};$$

– коефіцієнт використання k -го вузла

$$U_k = \frac{B_k}{T}; \quad (9.1)$$

– середній час обслуговування у k -му вузлі

$$S_k = \frac{B_k}{C_k}; \quad (9.2)$$

– інтенсивність вихідного потоку вимог з k -го вузла

$$X_k = \frac{C_k}{T}. \quad (9.3)$$

3. Операційні залежності

Основні співвідношення операційного аналізу формуються у вигляді операційних залежностей. Розглянемо деякі з них.

Ймовірність переміщення вимог між вузлами можна подати формулою

$$q_{kj} = \begin{cases} \frac{C_{kj}}{C_k}, & k = 1, 2, \dots, n, \quad j = 0, 1, \dots, n, \\ \frac{A_{0j}}{A_0}, & k = 0, \quad j = 1, 2, \dots, n. \end{cases} \quad (9.4)$$

Крім того, для ймовірностей переходів між вузлами мають справджуватись формули:

$$\sum_{k=0}^n q_{kj} = 1, \quad \sum_{j=0}^n q_{kj} = 1.$$

Використовуючи формули (9.1)–(9.3), маємо:

$$U_k = X_k S_k. \quad (9.5)$$

Останній вираз — це закон коефіцієнта використання вузла.

Рівність $A_0 = C_0$ є ознакою балансу вхідного і вихідного потоків вимог для мережі. Рівності $A_j = C_j$, $j = 1, 2, \dots, n$, є ознаками балансів вхідних і вихідних потоків вимог для вузлів мережі.

У припущенні, що $A_j = C_j$, $j = 0, 1, \dots, n$, має місце *система балансу потоків вимог*

$$X_j = \sum_{k=0}^n X_k q_{kj}, \quad j = 0, 1, \dots, n. \quad (9.6)$$

Доведемо систему (9.6). З рівностей

$$A_j = A_{0j} + \sum_{k=1}^n C_{kj}, \quad j = 0, 1, \dots, n,$$

враховуючи, що $A_j = C_j$, маємо:

$$C_j = A_{0j} + \sum_{k=1}^n C_{kj}.$$

З формули (9.4) випливає, що $C_{kj} = C_k q_{kj}$, $A_{0j} = A_0 q_{0j}$. Отже,

$$C_j = A_0 q_{0j} + \sum_{k=1}^n C_k q_{kj} \Rightarrow C_j = \sum_{k=0}^n C_k q_{kj}, \quad j = 0, 1, \dots, n.$$

Поділивши всі рівняння останньої системи на T , отримаємо систему (9.6), що й потрібно було довести.

Приклад. На рис. 9.2 подано мережу систем масового обслуговування, яка працює з перевантаженням. Вузли мережі позначено квадратами з їхніми номерами, для позначення черг перед вузлами використовуються символи $\equiv$. Середній час (у секундах) обслуговування вимог у вузлах мережі є наступним: $S_1 = 0,05$, $S_2 = 0,03$, $S_3 = 0,09$, $S_4 = 0,01$. Проаналізувати вузькі місця мережі і визначити найбільшу кількість вимог, яку мережа зможе опрацювати за одиницю часу, тобто максимальну інтенсивність вхідного потоку. Обчислити коефіцієнти використання всіх вузлів мережі і вихідні потоки з усіх вузлів для знайденого вхідного потоку.

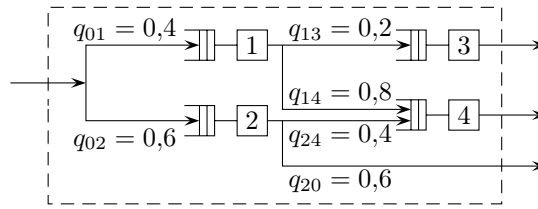


Рис. 9.2.

Розв'язання. Складаємо систему балансу (9.6) для мережі з рис. 9.2:

$$\begin{cases} X_0 = X_3 + X_4 + 0,6X_2, \\ X_1 = 0,4X_0, \\ X_2 = 0,6X_0, \\ X_3 = 0,2X_1, \\ X_4 = 0,8X_1 + 0,4X_2 \end{cases} \Rightarrow \begin{cases} X_1 = 0,4X_0, \\ X_2 = 0,6X_0, \\ X_3 = 0,08X_0, \\ X_4 = 0,56X_0. \end{cases}$$

За формулами (9.5) визначаємо коефіцієнти використання вузлів через інтенсивність вхідного потоку X_0 :

$$U_1 = 0,02X_0, \quad U_2 = 0,018X_0, \quad U_3 = 0,0072X_0, \quad U_4 = 0,0056X_0.$$

Найбільшим є коефіцієнт використання першого вузла, тому саме перший вузол є вузьким місцем даної мережі. Оскільки коефіцієнт використання вузла не може бути більшим від 1, то з рівності $0,02X_0 = 1$ отримуємо $X_0 = 50$ — максимальну інтенсивність вхідного потоку. Тепер можна обчислити відповідні інтенсивності вихідних потоків і коефіцієнти використання всіх вузлів:

$$X_1 = 20, \quad X_2 = 30, \quad X_3 = 4, \quad X_4 = 28, \\ U_1 = 1, \quad U_2 = 0,9, \quad U_3 = 0,36, \quad U_4 = 0,28. \quad \blacksquare$$

Лекція 10. Основи роботи з середовищем GPSS World, категорії і типи об'єктів мови GPSS, системні числові атрибути, формат рядка, арифметичні і логічні вирази

План

1. Основи роботи з середовищем GPSS World.
2. Категорії і типи об'єктів мови GPSS.
3. Системні числові атрибути.
4. Формат рядка.
5. Арифметичні і логічні вирази.

1. Основи роботи з середовищем GPSS World

Імітаційне моделювання можна здійснювати з допомогою програм, написаних на універсальних мовах програмування, таких як Delphi, C++ та інші. Але набагато зручніше для цього використовувати спеціальні мови програмування моделей. Однією з найпоширеніших є мова GPSS, яка розшифровується як General Purpose System Simulation — система моделювання загального призначення. Ця мова була розроблена ще на початку 60-х років XX-го століття Джеффері Гордоном. Зараз використовується GPSS World — сучасна реалізація цієї мови, розроблена у 2000 році фірмою Minuteman Software для операційної системи Windows. Вільну версію середовища GPSS World для студентів можна завантажити з офіційного сайту фірми Minuteman Software (www.minutemansoftware.com).

Мову імітаційного моделювання GPSS World найчастіше використовують для моделювання систем масового обслуговування. Проте її успішно можна використовувати і для розв'язування ряду інших задач імітаційного моделювання. Можливість інтегрування диференціальних рівнянь паралельно з імітаційним моделюванням робить мову GPSS World придатною для застосування комбінованих методів математичного моделювання.

Модель на мові GPSS записується у вигляді програми — послідовності операторів. Великі і малі латинські букви в тексті програми не розрізняються. Написаний текст моделі (програми) транслюється у внутрішній код, а потім запускається на виконання. Відповідно можуть з'являтися повідомлення про помилки трансляції чи часу виконання. Якщо помилок немає, то відображається статистика змодельованого процесу, яка містить інформацію про час роботи, зайняті комірки пам'яті тощо. Ця статистика і є результатом роботи моделі. Її детальність визначається використаними в тексті програми спеціальними операторами. Як і в інших мовах програмування, є можливість покрокового виконання програми, створення точок зупинки, перегляду змінних і т. п.

Робота з середовищем GPSS World суттєво не відрізняється від роботи з іншими сучасними програмами. Створення тексту моделі здійснюється в окремому вікні. Для трансляції моделі і запуску її на виконання використовується команда меню **Command►Create Simulation**, цю команду можна також виконати, натиснувши комбінацію клавіш **Ctrl+Alt+S**. При кожній трансляції моделі створюється окреме вікно журналу з повідомленнями про результати трансляції. При кожному запуску процесу моделювання у вікні журналу відображаються повідомлення про хід процесу моделювання. Крім того, кожний успішний запуск моделі генерує ще одне вікно із статистичним звітом. Переходити між вікнами можна звичайним способом з використанням меню **Window**.

При наявності помилок у моделі процес моделювання може продовжуватись необмежено довго. Перервати процес моделювання можна з допомогою команди меню **Command►Halt**. Цю команду можна також викликати, натиснувши клавішу **F4**.

Всі повідомлення про помилки виводяться у вікні журналу з вказуванням номерів рядка і стовпця, назви помилки і відображенням рядка, який призвів до помилки. Команда меню **Search►Next Error** відображає вікно з текстом моделі і виділяє помилковий (з точки зору транслятора) фрагмент рядка. Однак треба мати на увазі, що помилка насправді могла бути допущена у попередньому рядку програми. Якщо є кілька помилок, то команда **Search►Next Error** (або комбінація клавіш **Ctrl+Alt+N**) дозволяє переміщуватись між помилковими рядками.

Потрібно розрізняти модельний і реальний час моделювання. Зміст одиниці модельного часу визначає розробник моделі. Цією одиницею може бути секунда, хвилина, година тощо. Реальний час моделювання залежить від швидкодії комп'ютера.

Текст програми складається з послідовності операторів, якими є блоки, команди опису і команди керування. Блоки є найважливішими складовими елементами програми. Вони формують саму модель і використовуються протягом всього процесу моделювання. Команди опису описують різні об'єкти мови GPSS і можуть розміщуватись у будь-якому місці програми і в будь-якому порядку. По аналогії з іншими мовами програмування їх часто наводять на початку програми. У тексті програми також можуть бути присутніми деякі команди керування. Їх краще розміщувати в кінці програми. Команди опису і команди керування виконуються лише по одному разу в тому порядку, в якому вони записані в тексті програми.

При роботі з GPSS World можна користуватись довідковою системою середовища на англійській мові, до якої отримують доступ через меню **Help** або натисненням клавіші **F1**. Крім того, можна знайти електронну книгу **Reference Manual** у розділі **GPSS World Student Version** (меню **Пуск**). У книзі [10] вся довідкова система GPSS World перекладена на російську мову. Можна також користуватись книгами [4, 14].

2. Категорії і типи об'єктів мови GPSS

Моделі на мові GPSS є сукупностями окремих елементів, які називають об'єктами. Основні категорії об'єктів мови GPSS — операційна, динамічна, обчислювальна, запам'ятовуюча, апаратна і статистична.

До операційної категорії належить єдиний тип об'єктів — *блоки*. Кожен блок має входи й виходи, з допомогою яких здійснюється зв'язок у моделі. Існує два особливих блоки: **GENERATE**, який має лише вихід, і **TERMINATE**, який має тільки вхід.

До динамічної категорії належать *транзакти*. Функціонування об'єкта відбувається в моделі у вигляді переміщення транзактів від блока **GENERATE** у блок **TERMINATE** через проміжні блоки. Транзакти (повідомлення) є абстрактними рухомими об'єктами, які, переміщуючись між блоками, викликають різні дії і зазнають їх. Транзакти послідовно входять у всі проміжні блоки, якщо не використовуються спеціальні блоки, які змінюють маршрут руху транзактів. При русі транзактів через модель команди опису і команди керування ігноруються. Поняття транзакта встановлює розробник моделі. Наприклад, при розробці моделі великого магазину в поняття транзакта можна закласти покупця, який переміщується між відділами, може стояти в черзі, здійснювати покупки, спілкуватись з іншими покупцями, вимагати книгу скарг тощо. При моделюванні виробничого процесу зручно під транзактами розуміти деталі.

Об'єктами обчислювальної категорії є змінні, функції, арифметичні вирази і PLUS-процедури.

Для збереження даних під час моделювання використовують об'єкти запам'ятовуючої категорії — комірки пам'яті та матриці комірок пам'яті. Вони можуть використовуватись також для обчислень.

До апаратної категорії відносять три типи об'єктів: одноканальні пристрої, багатоканальні пристрої і логічні перемикачі¹. Одноканальні і багатоканальні пристрої (прилади) моделюють об'єкти, в яких відбувається обробка транзактів. Як правило, вона пов'язана з витратами часу. Одноканальний пристрій може бути зайнятий лише одним транзактом. Багатоканальні пристрої можуть обслуговувати кілька транзактів одночасно. Логічні перемикачі можуть перебувати в одному з двох станів — 0 (вимкнений) і 1 (увімкнений). Транзакти при русі по моделі можуть вмикати, вимикати чи інвертувати перемикачі.

До статистичної категорії належать черги і таблиці. Транзакти в процесі руху можуть затримуватись перед блоками, вхід в які в заданих умовах неможливий. У цьому випадку утворюються черги. Система GPSS дозволяє зібрати статистику про них. Для відображення даних у стандартній табличній формі про різні події під час моделювання використовуються таблиці.

¹У довідковій системі GPSS World вживаються терміни: Facilities, Storages, Logicswitches. У деяких книгах багатоканальні пристрої називають пам'ятями, а логічні перемикачі — логічними ключами.

3. Системні числові атрибути

Всі об'єкти мови GPSS мають числові характеристики, які називають *системними числовими атрибутами*². Вони виконують функцію імен. Системні числові атрибути бувають двох видів. Системний числовий атрибут першого виду складається з двох частин. Перша частина вказує на клас (тип) об'єкта — групове ім'я, а друга — ідентифікує конкретного члена групи (вузьке ім'я).

Є кілька десятків класів системних числових атрибутів першого виду. Усі вони мають вигляд однієї чи двох латинських букв. Наприклад, Q — клас черг, V — клас змінних, FN — клас функцій.

Друга частина системного числового атрибута (вузьке ім'я) може бути двох видів: 1) послідовність латинських букв, цифр і символів підкреслення «_», яка починається з букви; 2) натуральне число. У деяких випадках вузьке ім'я може бути ім'ям лише першого виду, в інших — як першого, так і другого виду.

Системні числові атрибути другого виду є атомарними і не мають вузького імені. Їх є лише кілька: A1, AC1, C1, M1, PR, TG1, XN1, Z1. Наприклад, AC1 — абсолютний модельний час.

Великі і малі букви в системних числових атрибутах, як і скрізь у GPSS, не розрізняються. Конкретні класи системних числових атрибутів розглядаються у наступних параграфах, а їхній список наведено у § ???. У межах однієї моделі однакові вузькі імена можуть використовуватись для позначення об'єктів різного типу. Зарезервовані ключові слова не можна використовувати як імена. В одних випадках використовують лише вузькі імена, а в інших потрібно записувати (повні) системні числові атрибути. При порушенні правил запису виникатимуть повідомлення про помилки. Треба бути обережними також з назвами класів системних числових атрибутів. Якщо таку назву (наприклад, V) використати як вузьке ім'я, то виникнуть малозрозумілі повідомлення про помилки або модель працюватиме неправильно. Щоб гарантувати відмінність вузьких імен від класів системних числових атрибутів, рекомендується використовувати вузькі імена, що складаються не менше, ніж з трьох букв, або використовувати в іменах символ підкреслення.

Клас системного числового атрибута відокремлюється від вузького імені, що містить букви, знаком долара «\$». Число від класу системного числового атрибута нічим не відокремлюється. Приклади: Q\$otx, FN\$abcd1, P125, RN4.

4. Формат рядка

Кожен оператор програми (за винятком операторів у тілі PLUS-процедури) записується в окремому рядку, тобто в рядку може бути тільки одна команда опису, один блок або одна команда керування. Рядок може бути також порожнім або містити лише коментар. Формат рядка є наступним:

<номер> <мітка> <ключове слово> <операнди> <коментар>

Кутові дужки тут використані лише для відокремлення елементів рядка, ставити їх не потрібно.

Номер є необов'язковим елементом рядка, ніде не використовується й ігнорується. Він може використовуватись хіба що для зручності перегляду першої таблиці статистики, яка відображає кількість транзактів, що пройшли через кожен блок моделі.

Мітка теж є необов'язковим елементом рядка. Як і вузьке ім'я першого виду, мітка складається з латинських букв, цифр і символів підкреслення. Зрозуміло, що вона не повинна збігатися з ключовими словами і класами системних числових атрибутів. Мітка, яка стоїть у блоці, використовується для (умовного чи безумовного) переходу транзактів у цей блок. При оголошенні змінної, функції, таблиці тощо мітка виконує роль вузького імені цієї змінної, функції, таблиці.

Ключове слово є головним елементом рядка і визначає його зміст. Ключовим словом може бути назва блока, команди опису або команди керування.

Кількість операндів і їхній формат залежить від ключового слова. Як правило, операнди записуються через кому без пропусків (до чи після неї). В окремих випадках перший операнд є умовним

²У деяких книгах їх називають стандартними числовими атрибутами. У довідковій системі GPSS World системний числовий атрибут позначають аббревіатурою SNA — System Numerical Attribute.

і відокремлюється від решти операндів пропуском, а не комою. Операндами можуть бути числа, системні числові атрибути, вузькі імена, назви міток, арифметичні і логічні вирази, деякі ключові слова. Ціла і дробова частини числа відокремлюються дробовою крапкою. Якщо операнд має бути цілим, а отримується дробове число, то береться його ціла частина. Операнди можуть бути як обов'язковими, так і необов'язковими. Їхній порядок є строго фіксованим. Якщо котрийсь з проміжних необов'язкових операндів треба пропустити, а якийсь з наступних (можливо останній) операнд мусить бути, то всі коми до нього мають бути збережені. Кінцеві коми ігноруються.

Коментар є необов'язковим елементом рядка і починається з символу «;». Усі символи після нього в рядку ігноруються. Інколи буває зручно «закоментувати» кілька рядків моделі, поклавши на початку цих рядків символи «;». Рядок, який починається з символу «*», також розглядається як коментар.

Всі елементи рядка відокремлюються один від одного одним чи кількома пропусками або символами табуляції. На початку рядка також може знаходитись довільна кількість пропусків чи знаків табуляції.

Обсяг моделі при використанні вільної студентської версії середовища GPSS World має не перевищувати 180 блоків.

Створити рядок з блоком мови GPSS можна також командою меню **Edit►Insert GPSS Blocks**. У цьому випадку достатньо вибрати назву блока з запропонованих варіантів і ввести в спеціальному вікні потрібні операнди, мітку та коментар.

Для кращого сприйняття програм на мові GPSS World скрізь у посібнику назви блоків, команд опису, команд керування, класів системних числових атрибутів та інших зарезервованих термінів наводяться великими буквами, всі імена і мітки — малими буквами, а імена стандартних функцій записуються з використанням великих і малих букв. Читач може не дотримуватись цього правила і набирати програми, наприклад, лише малими буквами. У статистичному звіті великими буквами відображаються не лише назви блоків, але й всі імена і мітки, використані в програмі.

5. Арифметичні і логічні вирази

Замість більшості операндів у блоках мови GPSS World та в командах опису змінних можна використовувати арифметичні і логічні вирази. Вираз є розширенням стандарту мови GPSS і тому задається у круглих дужках. У командах опису змінних круглі дужки є необов'язковими. Вирази можуть містити числа, системні числові атрибути, змінні користувача, математичні функції і функції для генерації випадкових величин, поєднані знаками арифметичних і логічних операцій. Нагадаємо, що з допомогою системних числових атрибутів здійснюється доступ до змінних, функцій, комірок пам'яті та інших об'єктів.

У виразах можна використовувати наступні операції:

$\wedge$ — піднесення до степеня;

$\#$ — множення;

$/$ — ділення;

$\backslash$ — ціла частина від ділення;

$@$ — остача від ділення;

$+$ — додавання;

$-$ — віднімання;

$=$ або 'E' — дорівнює, $A=B$ дає 1, якщо A дорівнює B , інакше — 0;

$<$ або 'L' — менше, $A<B$ дає 1, якщо A менше B , інакше — 0;

$>$ або 'G' — більше, $A>B$ дає 1, якщо A більше B , інакше — 0;

$\leq$ або 'LE' — менше або дорівнює, $A\leq B$ дає 1, якщо A менше або дорівнює B , інакше — 0;

$\geq$ або 'GE' — більше або дорівнює, $A\geq B$ дає 1, якщо A більше або дорівнює B , інакше — 0;

'NE' — не дорівнює, $A\neq B$ дає 1, якщо A не дорівнює B , інакше — 0;

$\&$ або 'AND' — логічне «І», $A\&B$ дає 1, якщо A і B не дорівнюють нулю, інакше — 0;

$|$ або 'OR' — логічне «АБО», $A|B$ дає 1, якщо A або B не дорівнює нулю, інакше — 0.

Операції над елементами виразу виконуються в такій послідовності у порядку спадання пріоритетів:

- $\wedge$ — піднесення до степеня;
- $\#$, $/$, $\backslash$ — множення, ділення, ціла частина від ділення;
- $@$ — остача від ділення;
- $-$, $+$ — віднімання, додавання;
- $<$, $>$, $<=$, $>=$ — порівняння;
- $=$, $\neq$ — дорівнює, не дорівнює;
- $\&$ — логічне «І»;
- $|$ — логічне «АБО».

Для зміни порядку виконання операцій використовують круглі дужки. У виразі біля знаків арифметичних і логічних операцій дозволяється ставити пропуски.

Приклади: $((X4+V\$rrr)\#P2)$, $(P1=1 \ \& \ Q\$chr1<3)$.

Лекція 11. Блоки і команди опису, пов'язані з транзактами, пристроями і таблицями

План

1. Блоки створення, знищення, затримки, зміни параметрів і пріоритету транзактів.
2. Основні блоки і команди опису, пов'язані з пристроями.
3. Блоки і команди опису, пов'язані зі статистичною категорією.

1. Блоки створення, знищення, затримки, зміни параметрів і пріоритету транзактів

Для створення і введення транзактів у модель використовують блок **GENERATE**. Його формат є наступним:

GENERATE *A,B,C,D,E*.

Обов'язковим є лише перший операнд *A* або четвертий операнд *D*. Якщо операнд *A* є числом, а операнд *B* — відсутній, то в модель вводиться детермінований потік транзактів (через кожні *A* одиниць модельного часу в модель вводиться один транзакт). Для моделювання випадкового потоку транзактів з рівномірним законом розподілу на проміжку $[a, b]$ використовують операнди *A* і *B*. У цьому випадку операнд *A* задає середину проміжку $\frac{a+b}{2}$, а операнд *B* — половину довжини проміжку $\frac{b-a}{2}$. Якщо операндом *A* є функція, створена командою опису **FUNCTION** (див. § 1) чи одним з 35 розподілів випадкових величин (див. § 2), а операнд *B* — відсутній, то інтервали часу між створеними транзактами визначатимуться цією функцією. Якщо операнд *A* є функцією, а операнд *B* — числом, то проміжки часу між транзактами визначатимуться рівномірним розподілом на проміжку $[A - B, A + B]$. Якщо задано дві функції або функція задана тільки в операнді *B*, а операнд *A* є числом, то проміжки часу між транзактами визначатимуться за правилом $A \cdot B$.

Інші операнди блока **GENERATE** використовують зрідка. Операнд *C* задає час появи першого транзакта. Операнд *D* вказує кількість транзактів, які потрібно згенерувати. Ціле число на місці операнда *E* визначає пріоритет транзактів. Якщо операнд *E* не використовується, то створені транзакти мають пріоритет 0. Чим більшим є число, тим вищим є пріоритет.

Якщо проміжок часу між появою транзактів з якоїсь причини виявиться меншим від нуля, то з'явиться повідомлення про помилку. Транзакт не може зайти в блок **GENERATE**; при виникненні такої спроби буде згенеровано повідомлення про помилку.

Приклади:

GENERATE 100 — через кожні 100 одиниць модельного часу в модель вводиться один транзакт;

GENERATE 100,30 — інтервали часу між появами транзактів мають рівномірний розподіл на проміжку $[70, 130]$;

GENERATE FN\$func1 — інтервали часу між потраплянням транзактів у модель задаються значеннями функції *func1*;

GENERATE ,,5,2 — на початку моделювання одночасно створюється п'ять транзактів з пріоритетом 2.

Процес моделювання запускається командою керування

START *A*,

де *A* — початкове значення лічильника завершень (натуральне число). Команду **START** можна дати з меню **Command** після трансляції моделі або навести її в кінці моделі. В останньому випадку процес моделювання автоматично запускається після трансляції моделі.

Вийшовши з блока **GENERATE** і пройшовши через модель, транзакт виводиться з моделі (знищується) блоком

TERMINATE *A*,

де *A* — ціле значення, яке віднімається від значення лічильника завершень. Коли значення лічильника завершень стає рівним нулю, моделювання припиняється. Якщо операнд *A* не використовується, то

він вважається рівним нулю, тобто при знищенні транзакта відповідним блоком **TERMINATE** значення лічильника завершень не змінюється. Якщо використовується блок **TERMINATE 1**, то команда керування **START A** запускає процес моделювання руху A транзактів через модель, наприклад:

```
GENERATE 40,10
```

```
...
```

```
TERMINATE 1
```

```
START 25
```

У деяких випадках необхідно здійснювати моделювання деякого процесу протягом певного часу. Для цього на початку або в кінці моделі додається пара блоків:

```
GENERATE A
```

```
TERMINATE 1
```

де A — час, протягом якого має здійснюватись моделювання. Крім того, всі інші блоки **TERMINATE** у моделі дають з нульовим операндом (**TERMINATE 0**), а для запуску моделювання застосовують команду **START 1**.

Для затримки транзактів з метою імітації їхнього обслуговування використовується блок

ADVANCE A, B .

Зміст операндів A і B тут той самий, що й у блоці **GENERATE**. Зокрема, якщо є лише операнд A , то він визначає час затримки транзакта. Якщо є операнди A і B , причому операнд B є числом, то час затримки є випадковим числом, рівномірно розподіленим на проміжку $[A - B, A + B]$.

Приклади:

ADVANCE 5,4 — затримка транзактів на 5,4 одиниці модельного часу;

ADVANCE 5,2 — час затримки транзактів є випадковою величиною, рівномірно розподіленою на проміжку $[3, 7]$;

ADVANCE FN\$func1 — час затримки транзактів визначається функцією **func1**.

Кожен транзакт характеризується своїм номером, параметрами, пріоритетом і часом перебування в моделі. Транзакти автоматично нумеруються натуральними числами. У кожен момент часу обробляється тільки один транзакт. Такий транзакт називають *активним*. Номер активного транзакта міститься в системному числовому атрибуті **XN1**.

Параметри транзакта дозволяють зберігати і переміщувати по моделі інформацію, визначену розробником моделі. Блок **ASSIGN** змінює значення параметра транзакта, який зайшов у цей блок. Блок **ASSIGN** може працювати в трьох режимах: фіксації, збільшення і зменшення. Для цього використовуються три формати:

ASSIGN A, B присвоює параметру A значення B (режим фіксації);

ASSIGN $A+, B$ збільшує значення параметра A на B одиниць (режим збільшення);

ASSIGN $A-, B$ зменшує значення параметра A на B одиниць (режим зменшення).

Операндом A є вузьке ім'я параметра транзакта (ним може бути натуральне число або послідовність латинських букв, цифр і символів підкреслення, що починається з букви), а не системний числовий атрибут. Якщо параметру транзакта не було присвоєно значення, а він використовується в блоці **ASSIGN $A+, B$** , то вважається, що його початкове значення дорівнює нулю. Параметрам транзактів можна також надавати рядкових значень, їх задають у подвійних лапках.

Клас системного числового атрибута для параметрів транзактів позначається буквою **P**. Опитують параметри активного транзакта з допомогою системних числових атрибутів. Спроба опитування параметра транзакта, якому не присвоєно жодного значення, викликає повідомлення про помилку. Транзакт може мати довільну кількість параметрів. Різні транзакти можуть мати різні параметри, які можуть містити різні значення.

Приклади:

ASSIGN 1,10 — параметру з ім'ям 1 (номер 1) транзакта, який зайшов у блок ASSIGN, присвоюється значення 10;

ASSIGN abcd+,2.1 — параметр abcd транзакта, який зайшов у блок ASSIGN, збільшується на величину 2,1;

ASSIGN Q_,"Hello!" — параметру Q_ транзакта, який зайшов у блок ASSIGN, присвоюється значення "Hello!";

P1 — звертання до параметра з ім'ям 1 (номер 1) активного транзакта;

P\$abcd — звертання до параметра abcd активного транзакта;

ASSIGN 2,(2#P1+2.4) — параметру з номером 2 транзакта, який зайшов у блок ASSIGN, присвоюється результат обчислення виразу $2\#P1+2.4$.

Блок PRIORITY A змінює рівень пріоритету транзакта (A — новий пріоритет). Для визначення пріоритету активного транзакта використовується системний числовий атрибут PR. З врахуванням пріоритетів здійснюється моделювання роботи одноканальних і багатоканальних пристроїв.

2. Основні блоки і команди опису, пов'язані з пристроями

Досить часто виникає потреба моделювати такі ситуації, коли одночасно може обслуговуватись не більше одного чи обмеженої кількості транзактів. Для імітації таких ситуацій використовують одноканальні і багатоканальні пристрої.

Блок SEIZE A моделює вхід в *одноканальний пристрій* з ім'ям A. Ім'ям може бути послідовність латинських букв, цифр і символів підкреслення, що починається з букви, або натуральне число. Якщо одноканальний пристрій A є вільним, то транзакт, який потрапляє в блок SEIZE A, займає цей пристрій. Якщо одноканальний пристрій A вже зайнятий іншим транзактом, то нові транзакти затримуються у попередньому блоці. Черга формується з врахуванням пріоритету транзактів. При однаковому пріоритеті нові транзакти стають у хвіст черги.

Блок RELEASE A моделює вихід з одноканального пристрою з ім'ям A. Транзакт, який входить у цей блок, звільняє одноканальний пристрій.

Блок RETURN A є синонімом до блока RELEASE A.

Приклад. У магазині є одна каса. Час обслуговування одного покупця в касі є рівномірно розподіленою випадковою величиною на проміжку [10,60] секунд. Відомо, що покупці з'являються через кожні 0–80 секунд. З допомогою мови імітаційного моделювання GPSS здійснити моделювання обслуговування 100 покупців у касі, визначити коефіцієнт її завантаження і час обслуговування 100 покупців.

Розв'язання. За одиницю модельного часу в цій задачі зручно вибрати секунду. Роль покупців у моделі будуть виконувати транзакти. Інтервали часу між надходженням покупців вважаємо випадковими величинами з рівномірним розподілом на відрізку [0,80]. Взагалі, у всіх задачах, де відомо лише проміжок зміни випадкової величини, будемо вважати, що ця випадкова величина розподілена рівномірно. Для моделювання обслуговування 100 покупців використовуємо команду START з операндом 100 і блок TERMINATE з операндом 1. Програма має вигляд:

```
GENERATE 40,40
SEIZE kasa
ADVANCE 35,25
RELEASE kasa
TERMINATE 1
START 100
```

За результатами моделювання отримуємо звіт (рис. 11.1), з якого робимо висновок про те, що коефіцієнт завантаження каси приблизно дорівнює 0,84 (стовпець UTIL. таблиці FACILITY), а загальний час обслуговування 100 покупців складає близько 70 хвилин (END TIME).

Saturday, January 04, 2014 12:17:58

START TIME		END TIME	BLOCKS	FACILITIES	STORAGES
0.000		4183.558	5	1	0

NAME		VALUE
KASA		10000.000

LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT COUNT	RETRY
	1	GENERATE	100	0	0
	2	SEIZE	100	0	0
	3	ADVANCE	100	0	0
	4	RELEASE	100	0	0
	5	TERMINATE	100	0	0

FACILITY	ENTRIES	UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
KASA	100	0.841	35.167	1	0	0	0	0	0

FEC XN	PRI	BDT	ASSEM	CURRENT	NEXT	PARAMETER	VALUE
101	0	4223.466	101	0	1		

Рис. 11.1

Треба мати на увазі, що чимало характеристик у звіті залежить від випадкових чисел, а тому повторний запуск моделі командою **START 100** може дати дещо інші числа. Визначити довжину черги до каси з використанням наведеної моделі неможливо. ■

Блок

PREEMPT A,B,C,D,E

також моделює вхід в *одноканальний пристрій* з ім'ям *A*. Обслуговування транзакта в такому пристрої може бути перерване іншим транзактом. Обов'язковим є лише операнд *A*. Якщо операнд *B* є відсутнім, то новий транзакт перериває обслуговування транзакта, який вже зайняв пристрій, при умові, що той транзакт сам не є «загарбником» (*режим переривання*). Робота «загарбника» не переривається. Якщо операндом *B* є системний числовий атрибут *PR*, то перебування транзакта в пристрої переривається лише тоді, коли новий транзакт має більш високий пріоритет (*режим пріоритетів*).

Якщо немає операндів *C*, *D*, *E*, то транзакт автоматично відновить обслуговування у пристрої після завершення обслуговування «загарбника». Час дообслуговування дорівнює часу, який залишався транзакту до завершення обслуговування в одноканальному пристрої.

Операнд *C* може містити мітку або номер блока, куди потрібно відправити транзакт, обслуговування якого було перерване. В операнді *D* може вказуватись ім'я або номер параметра транзакта, в якому зберігатиметься залишок часу, що залишався транзакту до завершення обслуговування у пристрої. Якщо операнд *E* містить ключове слово **RE**, то транзакт, обслуговування якого було перерване, не буде дообслуговуватись у пристрої. Якщо використовується операнд *E*, то одночасно має використовуватись операнд *C*. Якщо операнд *C* використовується без операнда *E*, то транзакт зберігає право на дообслуговування у пристрої. Тому цей транзакт не може бути знищений блоком **TERMINATE** перш, ніж транзакт увійде в блок **RELEASE** або **RETURN** для пристрою з ім'ям *A*.

Для моделювання виходу з одноканального пристрою, вхід в який моделювався блоком **PREEMPT**, має використовуватись блок **RETURN A** або **RELEASE A**.

Приклад моделювання роботи одноканального пристрою, обслуговування транзакта в якому може бути перерване транзактом з вищим пріоритетом:

```
PREEMPT pryl,PR
ADVANCE 1
RETURN pryl
```

Для використання *багатоканального пристрою* в моделі його спочатку потрібно описати командою

N STORAGE A,

де N — ім'я багатоканального пристрою, A — його ємність, тобто кількість каналів. Ім'ям може бути послідовність латинських букв, цифр і символів підкреслення, що починається з букви. Команду **STORAGE**, як і інші команди опису, зазвичай записують на початку програми.

Блок

ENTER A, B

моделює вхід у багатоканальний пристрій з ім'ям A , описаний командою **STORAGE**. Необов'язковий операнд B вказує кількість каналів, які займає транзакт при вході в багатоканальний пристрій. За замовчуванням транзакт претендує тільки на один канал. Якщо у багатоканальному пристрої немає достатньої кількості вільних каналів, то транзакти затримуються в попередньому блоці. Черга формується з урахуванням пріоритету транзактів. При однаковому пріоритеті нові транзакти стають у хвіст черги.

Блок

LEAVE A, B

моделює вихід із багатоканального пристрою з ім'ям A . Необов'язковий операнд B вказує кількість каналів, які звільняє транзакт при виході з багатоканального пристрою. За замовчуванням транзакт звільняє один канал.

Приклад моделювання роботи п'ятиканального пристрою:

bkr STORAGE 5

ENTER bkr

ADVANCE 2

LEAVE bkr

3. Блоки і команди опису, пов'язані зі статистичною категорією

Частина результатів моделювання автоматично наводиться у статистичному звіті, який формується після завершення моделювання. Однак для збору статистики часу виконання деякого процесу або інформації про черги доводиться використовувати спеціальні команди опису і блоки, які розглядаються у цьому параграфі.

Команда **TABLE** використовується для опису *таблиць*, які збиратимуть і відображатимуть статистичну інформацію про результати моделювання. Цю команду застосовують у наступному форматі:

N TABLE $A, B, C, D,$

де N — ім'я таблиці, A — системний числовий атрибут, який визначає те, що буде відображатись у таблиці, B — верхня межа першого інтервалу таблиці, C — ширина всіх інтервалів, крім першого і останнього, D — кількість інтервалів (натуральне число). Якщо число D виявиться надто великим, то в таблиці буде відображено інформацію лише для тих інтервалів, яким реально відповідатимуть транзакти. Якщо число D буде надто малим, то в таблиці буде наведено детальну інформацію лише про частину транзактів. У таблиці відзначається, скільки транзактів (та який їхній відсоток) за деяким критерієм відповідає кожному з інтервалів $(-\infty, B]$, $(B, B+C]$, $(B+C, B+2C]$, $(B+2C, B+3C]$, $\dots$, $(B+C(D-3), B+C(D-2)]$, $(B+C(D-2), \infty)$, а також виводиться середнє значення і середньоквадратичне відхилення.

Найчастіше створюють таблиці, в яких відображається інформація про розподіл часу руху транзактів через модель або її частину. Системний числовий атрибут **M1** забезпечує доступ до часу перебування транзакта в моделі. Його використовують замість операнда A у команді **TABLE** для збору інформації в таблиці про розподіл часу руху транзактів у моделі від моменту їхнього створення до потрапляння у блок **TABULATE A** , де A — ім'я таблиці.

Системний числовий атрибут **MP α** використовують для збору інформації в таблиці про розподіл часу руху транзактів через частину моделі від їхнього входження у блок **MARK α** до потрапляння у блок **TABULATE A** , де A — ім'я таблиці.

Можна також використовувати на місці операнда *A* у команді **TABLE** системний числовий атрибут з будь-яким параметром транзакта. У цьому випадку таблиця відображатиме частотний розподіл транзактів різного типу (відповідно до вибраного параметра).

Приклад 1. Доповнимо програму для прикладу з § 2 (с. 43) кількома операторами для збору статистики розподілу часу перебування покупців біля каси і часу обслуговування покупців касиром.

Розв’язання. Створимо дві таблиці — **tab1** і **tab2**. У таблиці **tab1** будемо збирати статистику розподілу часу, необхідного покупцям для перебування в черзі до каси і здійснення оплати, а в таблиці **tab2** — лише для здійснення оплати. Програма може бути, наприклад, такою:

```
tab1 TABLE M1,10,10,100
tab2 TABLE MP1,10,10,100
GENERATE 40,40
SEIZE kasa
MARK 1
ADVANCE 35,25
RELEASE kasa
TABULATE tab1
TABULATE tab2
TERMINATE 1
START 100
```

Отримані таблиці наведені на рис. 11.2. З них, зокрема, видно, що середній час перебування покупця біля каси складає 70 секунд (що на 10 секунд більше, ніж максимальний можливий час обслуговування покупця касиром), а один з покупців на перебування біля каси витратив понад три хвилини (останній рядок першої таблиці). ■

TABLE	MEAN	STD.DEV.	RANGE	RETRY	FREQUENCY	CUM. %
TAB1	70.000	38.643		0		
			10.000 - 20.000	8	8.00	
			20.000 - 30.000	9	17.00	
			30.000 - 40.000	10	27.00	
			40.000 - 50.000	9	36.00	
			50.000 - 60.000	10	46.00	
			60.000 - 70.000	8	54.00	
			70.000 - 80.000	6	60.00	
			80.000 - 90.000	9	69.00	
			90.000 - 100.000	10	79.00	
			100.000 - 110.000	4	83.00	
			110.000 - 120.000	3	86.00	
			120.000 - 130.000	8	94.00	
			130.000 - 140.000	2	96.00	
			140.000 - 150.000	0	96.00	
			150.000 - 160.000	3	99.00	
			160.000 - 170.000	0	99.00	
			170.000 - 180.000	0	99.00	
			180.000 - 190.000	1	100.00	
TAB2	35.167	15.230		0		
			10.000 - 20.000	24	24.00	
			20.000 - 30.000	15	39.00	
			30.000 - 40.000	17	56.00	
			40.000 - 50.000	25	81.00	
			50.000 - 60.000	19	100.00	

Рис. 11.2

У § 2 говорилося про те, що при використанні одноканальних чи багатоканальних пристроїв черга створюється автоматично. Однак зі звіту для прикладу § 2 (с. 43) взагалі не видно черги, а з прикладу 1 § 3 видно, що у звіті є лише опосередкована інформація про чергу. Для того щоб побачити у звіті характеристики черги, треба використовувати спеціальні блоки.

Блок

QUEUE *A,B*

моделює вхід у чергу з ім'ям A , при входженні транзакта в цей блок її довжина збільшується на B одиниць. Операнд B можна не вказувати, в такому разі довжина черги збільшується на одиницю.

Блок

DEPART A, B

моделює вихід з черги з ім'ям A , при входженні транзакта в цей блок її довжина зменшується на B одиниць. При відсутності операнда B довжина черги зменшується на одиницю. Транзакт може знаходитись у кількох чергах одночасно. Для правильної роботи блок QUEUE зазвичай розміщують перед блоком SEIZE, PREEMPT або ENTER, а блок DEPART — після. При використанні блоків QUEUE і DEPART у звіті виводиться інформація про середню і максимальну довжину черги, середній час перебування в черзі, кількість транзактів, які очікували в черзі тощо. Для того щоб отримати детальну статистику розподілу часу, проведеного транзактами в черзі, використовують команду опису таблиці для черги

N QTABLE A, B, C, D ,

де N — ім'я таблиці для черги, A — ім'я черги, B — верхня межа першого інтервалу таблиці, C — ширина всіх інтервалів, крім першого і останнього, D — кількість інтервалів. Операнди B , C і D задаються аналогічно відповідним операндам таблиці TABLE.

Для доступу до поточної довжини черги в моделі використовується системний числовий атрибут класу Q.

Приклад 2. Доповнимо програму для прикладу з § 2 (с. 43), крім операторів для збору статистики розподілу часу перебування покупців біля каси і часу обслуговування покупців касиром (приклад 1 § 3, с. 46), також операторами для визначення характеристик і детальної статистики черги.

Розв'язання. Отримаємо модель:

```
tab1 TABLE M1,10,10,100
tab2 TABLE MP1,10,10,100
tab3 QTABLE cherga,10,10,100
GENERATE 40,40
QUEUE cherga
SEIZE kasa
DEPART cherga
MARK 1
ADVANCE 35,25
RELEASE kasa
TABULATE tab1
TABULATE tab2
TERMINATE 1
START 100
```

Отримані додаткові таблиці наведені на рис. 11.3. З них, зокрема, видно, що максимальна довжина черги становила 4, середня довжина черги наближено дорівнює 0,8, середній час перебування в черзі — 35 секунд, але один покупець перебував у черзі понад дві хвилини. ■

QUEUE	MAX	CONT.	ENTRY	ENTRY(0)	AVE.CONT.	AVE.TIME	AVE.(-0)	RETRY
CHERGA	4	0	100	31	0.833	34.833	50.483	0
TABLE	MEAN	STD.DEV.		RANGE		RETRY	FREQUENCY	CUM.%
TAB3	34.833	35.299				0		
			-	-	10.000		37	37.00
			10.000	-	20.000		9	46.00
			20.000	-	30.000		5	51.00
			30.000	-	40.000		9	60.00
			40.000	-	50.000		5	65.00
			50.000	-	60.000		12	77.00
			60.000	-	70.000		7	84.00
			70.000	-	80.000		3	87.00
			80.000	-	90.000		5	92.00
			90.000	-	100.000		1	93.00
			100.000	-	110.000		1	94.00
			110.000	-	120.000		5	99.00
			120.000	-	130.000		1	100.00

Рис. 11.3

Лекція 12. Блоки і команди опису, пов'язані зі змінними, логічними перемикачами, комітками пам'яті і зміною маршруту руху транзактів

План

1. Використання змінних, логічних перемикачів і коміток пам'яті.
2. Блок TRANSFER для зміни маршруту руху транзактів.
3. Блоки TEST, GATE і LOOP для зміни маршруту руху транзактів.

1. Використання змінних, логічних перемикачів і коміток пам'яті

Для опису дійсної змінної з ім'ям N використовується команда

N FVARIABLE A .

Операнд A — це вираз для обчислення змінної, який будується за правилами побудови арифметичних виразів. Оточувати його круглими дужками необов'язково. У GPSS World команда опису VARIABLE є синонімом до команди FVARIABLE. У попередніх версіях вона була призначена для здійснення обчислень у множині цілих чисел. Для звертання до змінної використовується клас системних числових атрибутів V .

Команда опису

N BVARIABLE A

визначає логічну (булеву) змінну з ім'ям N . Така змінна може набувати лише значень 0 і 1. Операнд A — логічний вираз. Для звертання до логічної змінної використовується клас системних числових атрибутів BV .

Змінні дозволяють виконувати проміжні обчислення за певними, наперед заданими формулами. Під час моделювання ці формули не можуть змінюватись. Значення змінної обчислюється при звертанні до неї активного транзакта. У стандартному звіті значення змінних не виводяться.

Приклади:

aaaa FVARIABLE 10#X\$qwerty+P1 — створюється дійсна змінна з ім'ям aaaa;

bbbb BVARIABLE P1>3 & X1<3 — створюється логічна змінна з ім'ям bbbb;

V\$aaaa — звертання до дійсної змінної з ім'ям aaaa;

BV\$bbbb — звертання до логічної змінної з ім'ям bbbb.

Для збереження даних змінні не використовують, а здійснення обчислень в операндах блоків з допомогою арифметичних виразів дозволяє взагалі відмовитись від використання змінних. Логічні перемикачі і комітки пам'яті, на відміну від змінних, дозволяють зберігати інформацію, яка може бути довільним чином змінена в процесі моделювання.

Логічні перемикачі можна трактувати як змінні, які можуть набувати лише двох значень — 0 (вимкнено) і 1 (увімкнено). Блок

LOGIC X A

використовують для перемикання логічного перемикача. Операнд X є умовним оператором і може набувати наступних значень: S — увімкнути перемикач, R — вимкнути перемикач, I — інвертувати значення перемикача. Операнд A має бути ім'ям логічного перемикача. Значення логічного перемикача змінюється при входженні будь-якого транзакта в блок LOGIC.

Для визначення стану, в якому перебуває логічний перемикач, використовують системний числовий атрибут перемикача. Клас цього системного числового атрибута позначають буквами LS .

Логічний перемикач автоматично створюється при використанні блока LOGIC або при опитуванні системного числового атрибута перемикача, причому спочатку перемикач є неувімкненим (дорівнює нулю). Якщо логічний перемикач на початку моделювання має бути увімкненим, його описують командою

INITIAL A ,

де A — системний числовий атрибут перемикача.

Приклади:

INITIAL LS\$abracadabra — початкове значення логічного перемикача з ім'ям **abracadabra** є рівним 1;

LOGIC I 15 — при входженні транзакта в блок значення логічного перемикача з ім'ям **15** змінюється на протилежне.

Комірки пам'яті призначені для збереження числових або рядкових величин, їх можна трактувати як змінні і використовувати в обчисленнях. Блок **SAVEVALUE** змінює вміст комірки пам'яті. Його можна використовувати у трьох форматах:

SAVEVALUE A,B присвоює комірці пам'яті A значення B (режим фіксації);

SAVEVALUE A+,B збільшує значення комірки пам'яті A на B одиниць (режим збільшення);

SAVEVALUE A-,B зменшує значення комірки пам'яті A на B одиниць (режим зменшення).

Операндом A є вузьке ім'я комірки пам'яті (ним може бути натуральне число або послідовність латинських букв, цифр і символів підкреслення, яка починається з букви). Значення комірки пам'яті змінюється при входженні довільного транзакта в блок **SAVEVALUE**. Рядкові значення задають у подвійних лапках.

Клас системного числового атрибута для комірки пам'яті позначається буквою **X**. Опитують комірки пам'яті, як і інші об'єкти, з допомогою системних числових атрибутів. Комірка пам'яті автоматично створюється при використанні блока **SAVEVALUE** або при опитуванні системного числового атрибута комірки пам'яті, причому в останньому випадку її початкове значення дорівнює нулю. Якщо початкове значення комірки пам'яті має не дорівнювати нулю, то її описують командою

INITIAL A,B,

де A — системний числовий атрибут комірки пам'яті, B — початкове значення комірки пам'яті.

Приклади:

SAVEVALUE 2,20 — комірці пам'яті з ім'ям **2** (номер 2) присвоюється значення 20;

SAVEVALUE aaa+,2.3 — значення комірки пам'яті **aaa** збільшується на величину 2,3;

SAVEVALUE xy1,"Yes" — комірці пам'яті **xy1** присвоюється значення "Yes";

INITIAL X\$bbb,1 — початкове значення комірки пам'яті **bbb** дорівнює 1;

X2 — звертання до комірки пам'яті з ім'ям **2** (номер 2);

X\$aaa — звертання до комірки пам'яті **aaa**;

SAVEVALUE kin,(V\$xyz#P10-X12) — комірці пам'яті **kin** присвоюється результат обчислення арифметичного виразу (змінна **xyz** множиться на параметр номер 10 транзакта, який потрапив у цей блок, і від отриманого результату віднімається вміст комірки пам'яті номер 12).

2. Блок **TRANSFER** для зміни маршруту руху транзактів

Для зміни маршруту руху транзактів використовуються чотири види блоків. Спочатку розглянемо блок **TRANSFER**. Транзакт, що потрапляє в блок

TRANSFER A,B,C,D,

переходить у деякий інший блок відповідно до режиму переходу, визначеного операндом A . Розглянемо основні сім режимів.

1. Режим безумовного переходу. Якщо операнд A пропущено, то блок працює в режимі безумовного переходу. У цьому режимі транзакт переходить у блок, визначений міткою або номером, що міститься в операнді B . Інші операнди не використовуються. Блоки нумеруються послідовними натуральними числами, порожні рядки, рядки з командами або коментарями не нумеруються. Приклад:

TRANSFER ,mit — безумовний перехід на мітку **mit**.

2. Режим статистичної передачі. Якщо операнд A є числом, то блок працює в режимі статистичної передачі. Якщо $A \in (0,1)$, то транзакт з імовірністю A переходить у блок, визначений міткою або

номером, що міститься в операнді C , а з імовірністю $1 - A$ — у блок, визначений міткою або номером в операнді B . Якщо ціла частина операнда A є натуральним числом, меншим від 1000, то вона інтерпретується як кількість тисячних часток імовірності передачі. Якщо $A \leq 0$, то транзакт переходить у блок, визначений операндом B . Якщо $A \geq 1000$, то транзакт переходить у блок, визначений операндом C . Якщо операнд B пропущено, то транзакт з імовірністю $1 - A$ переходить у наступний блок. Якщо операнд C пропущено, то транзакт з імовірністю A переходить у наступний блок. Операнд D не використовується. Приклади:

TRANSFER 0,3,mit1,mit2 — транзакт з імовірністю 0,3 переходить у блок з міткою mit2, а з імовірністю 0,7 — у блок з міткою mit1;

TRANSFER 10,12 — транзакт з імовірністю 0,01 переходить у наступний блок, а з імовірністю 0,99 — у дванадцятий блок.

3. Режим BOTH. Якщо операнд A є ключовим словом BOTH, то блок працює в режимі BOTH. У цьому режимі транзакт намагається зайти в блок, визначений міткою або номером з операнда B . Якщо зайти у цей блок не вдається, бо він є входом у зайнятий одноканальний чи багатоканальний пристрій, то транзакт намагається зайти в блок, визначений міткою або номером з операнда C . Якщо не вдається зайти в жоден з двох блоків, то транзакт затримується у блоці TRANSFER до тих пір, доки він не зможе зайти в один з них. Якщо один з операндів B або C є відсутнім, то вважається, що він вказує на наступний блок. Операнд D не використовується. Приклади:

TRANSFER BOTH,rob1,rob2 — транзакт намагається зайти в один з двох блоків, визначених мітками rob1 і rob2;

TRANSFER BOTH,,rob2 — транзакт намагається зайти у наступний блок або у блок, визначений міткою rob2.

4. Режим ALL. Якщо операнд A є ключовим словом ALL, то блок працює в режимі ALL. У цьому режимі перевіряється блок, вказаний в операнді B . Якщо цей блок не може прийняти транзакт, то послідовно перевіряються всі блоки до тих пір, поки не буде досягнутий блок, визначений операндом C , або один з перевірених блоків не прийме транзакт. Розташування кожного блока, що послідовно перевіряється, обчислюється шляхом додавання до розташування останнього перевіреного блока операнда D . Блоки з більшим розташуванням, ніж блок, визначений операндом C , не розглядаються. Якщо операнд D не використовується, то послідовно перевіряються всі блоки між блоками, визначеними операндами B і C . Якщо не заданий операнд C , то перевіряється лише один блок. Транзакт скеровується в перший блок, який його приймає. Якщо жоден блок не приймає транзакт, то він затримується в блоці TRANSFER до тих пір, доки він не зможе зайти в один із цих блоків. Приклад:

TRANSFER ALL,b1_b,b1_e,2 — послідовно перевіряються всі блоки через один, починаючи з блока з міткою b1_b і закінчуючи блоком з міткою b1_e або попереднім до нього (в залежності від розташування цих блоків); якщо жоден блок не приймає транзакт, то він залишається у блоці TRANSFER до тих пір, доки його не зможе прийняти один з них.

5. Режим PICK. Якщо операнд A є ключовим словом PICK, то блок працює в режимі PICK. У цьому режимі місце призначення вибирається випадковим чином з-поміж блоків від блока, визначеного операндом B , до блока, визначеного операндом C . Операнд D не використовується.

Приклад рівноймовірного переходу на одну з п'яти міток mit1, mit2, mit3, mit4, mit5:

```
TRANSFER PICK,b1_b,b1_e
b1_b TRANSFER ,mit1
TRANSFER ,mit2
TRANSFER ,mit3
TRANSFER ,mit4
b1_e TRANSFER ,mit5
```

Рівноймовірний перехід на одну з кількох міток можна реалізувати акуратним використанням меншої кількості блоків TRANSFER в режимі статистичної передачі. Тому попередній приклад можна подати коротше:

TRANSFER 0.2,mit1
TRANSFER 0.25,mit2
TRANSFER 0.333,mit3
TRANSFER 0.5,mit4,mit5

3. Блоки TEST, GATE і LOOP для зміни маршруту руху транзактів

Для переходу транзактів при виконанні певних умов використовуються блоки TEST і GATE. Блок

TEST $X \ A, B, C$

порівнює операнди A і B відповідно до умовного операнда X . Операндами A і B можуть бути системні числові атрибути, числа, арифметичні або логічні вирази. Умовний операнд X може набувати наступних значень:

E — значення операнда A дорівнює значенню операнда B ;
 NE — значення операнда A не дорівнює значенню операнда B ;
 L — значення операнда A менше від значення операнда B ;
 LE — значення операнда A менше або дорівнює значенню операнда B ;
 G — значення операнда A більше від значення операнда B ;
 GE — значення операнда A більше або дорівнює значенню операнда B .

Якщо умова виконується, то транзакт, який потрапив у блок TEST, переходить у наступний блок. Необов'язковий операнд C може містити мітку або номер блока для альтернативного переходу. Якщо умова не виконується і заданий операнд C , то транзакт, який потрапив у блок TEST, переходить у блок, визначений операндом C . Якщо умова не виконується і відсутній операнд C , то транзакт, який потрапив у блок TEST, затримується у цьому блоці до тих пір, доки умова не буде виконана.

Для перевірки виконання складної умови у блоці TEST в одному з операндів A або B розміщують логічний вираз і перевіряють його на рівність 0 чи 1.

Приклади:

TEST E P1,2,mit — якщо значення параметра номер 1 транзакта, який потрапив у цей блок, дорівнює числу 2, то транзакт переходить у наступний блок, інакше — у блок з міткою mit;

TEST L X\$cin, (2#(X1+P\$cin)) — якщо вміст комірки пам'яті з ім'ям cin є меншим, ніж подвоєна сума вмісту комірки пам'яті номер 1 і параметра транзакта з ім'ям cin, то транзакт, який потрапив у цей блок, переходить у наступний блок, інакше він затримується у блоці TEST до тих пір, доки умова не буде виконана (умова виконається тільки тоді, коли якийсь інший транзакт відповідним чином змінить вміст хоч однієї з двох використаних в умові комірок пам'яті);

TEST E (X3>=5.1 & Q\$ch_<10),1 — якщо вміст комірки пам'яті номер 3 є більшим або рівним числу 5,1 і поточна довжина черги з ім'ям ch_ є меншою від 10, то транзакт, який потрапив у цей блок, переходить у наступний блок, інакше він затримується у блоці TEST до тих пір, доки умова не буде виконана.

Блок

GATE $X \ A, B$

змінює маршрут руху транзактів залежно від стану об'єкту, заданого операндом A . Необов'язковий операнд B може містити мітку або номер блока для альтернативного переходу. Умовний операнд X може набувати наступних значень для перевірки відповідних умов:

U — одноканальний пристрій з ім'ям A є зайнятим;
 NU — одноканальний пристрій з ім'ям A є вільним;
 I — обслуговування транзакта в одноканальному пристрої з ім'ям A щойно було перерване;
 NI — обслуговування транзакта в одноканальному пристрої з ім'ям A не було щойно перерване;
 FV — одноканальний пристрій з ім'ям A є доступним;
 FNV — одноканальний пристрій з ім'ям A є недоступним;
 SF — всі канали багатоканального пристрою з ім'ям A є зайнятими;

SNF — принаймні один канал багатоканального пристрою з ім'ям A є вільним;
 SE — всі канали багатоканального пристрою з ім'ям A є вільними;
 SNE — принаймні один канал багатоканального пристрою з ім'ям A є зайнятим;
 SV — багатоканальний пристрій з ім'ям A є доступним;
 SNV — багатоканальний пристрій з ім'ям A є недоступним;
 LS — логічний перемикач з ім'ям A увімкнений;
 LR — логічний перемикач з ім'ям A вимкнений;

M — блок MATCH, на який вказує операнд A , містить транзакт, що очікує на синхронізацію з активним транзактом;

NM — блок MATCH, на який вказує операнд A , не містить транзакта, що очікує на синхронізацію з активним транзактом.

Про доступність і недоступність одноканальних і багатоканальних пристроїв ітиметься в § ???. При синхронізацію руху транзактів з допомогою блоків MATCH ітиметься в § ???.

Якщо умова виконується, то транзакт, який потрапив у блок GATE, переходить у наступний блок. Якщо умова не виконується і заданий операнд B , то транзакт, який потрапив у блок GATE, переходить у блок, визначений операндом B . Якщо умова не виконується і відсутній операнд B , то транзакт, який потрапив у блок GATE, затримується у цьому блоці до тих пір, доки умова не буде виконана.

Приклади:

GATE U $pryl,mit$ — якщо одноканальний пристрій з ім'ям $pryl$ — зайнятий, то транзакт, який зайшов у цей блок, переходить у наступний блок, інакше — у блок з міткою mit ;

GATE LS 11 — якщо логічний перемикач з номером 11 є ввімкненим, то транзакт, який зайшов у цей блок, переходить у наступний блок, інакше він затримується у блоці GATE до тих пір, доки умова не буде виконана (умова виконається тільки тоді, коли якийсь інший транзакт змінить стан перемикача з номером 11).

Блок

LOOP A,B

призначений для організації руху транзактів у циклі. Операнд A має містити ім'я (номер) параметра транзакта, який визначатиме кількість проходів транзакта по циклу. Операнд B — мітка (номер) блока, в який переходить транзакт, якщо його параметр A не дорівнює нулю. При входженні транзакта в блок LOOP значення його параметра A зменшується на одиницю. Якщо воно є більшим від нуля, то транзакт переходить у блок, на який вказує операнд B . Якщо воно дорівнює нулю, то транзакт переходить у наступний блок. Якщо значення параметра A виявиться меншим від нуля, то виникне помилка.

Приклад створення циклу, який буде виконуватись п'ять разів (якщо в тілі циклу не змінюватиметься параметр з номером 1, який використовується як лічильник ітерацій):

```

ASSIGN 1,5
mit ...
...
LOOP 1,mit
```

Лекція 13. Функції і робота з випадковими величинами

План

1. Функції і генератори випадкових чисел.
2. Математичні функції і функції для генерації випадкових величин.
3. Приклади складання моделей.

1. Функції і генератори випадкових чисел

У мові GPSS World можна одночасно використовувати кілька різних генераторів випадкових чисел. Для доступу до них застосовують системні числові атрибути класу RN, наприклад: RN8, RN\$vyp_ch. Генератори випадкових чисел створюють цілі числа, рівномірно розподілені на проміжку $[0, 999]$. Початкове значення нумерованого генератора випадкових чисел визначається його номером.

У блоках GENERATE, ADVANCE і TRANSFER за замовчуванням для генерації випадкових чисел використовується системний числовий атрибут RN1, але його можна замінити з допомогою вкладки Random Numbers вікна, яке відкривається командою меню Edit►Settings.

Функція в мові GPSS задається таблицею. Правила пошуку значення функції за таблицею визначає команда опису FUNCTION. Вона записується принаймні у два рядки, причому другий і наступні рядки опису функції мають містити список даних функції (тобто саму таблицю):

```
N FUNCTION A,Kn  
X1,Y1/X2,Y2/.../Xn,Yn
```

Мітка N визначає ім'я функції. Операндом A є аргумент функції, ним може бути будь-який системний числовий атрибут або арифметичний вираз у дужках. Часто буває потрібним задати деяку випадкову залежність. Тоді на місці операнда A розміщують один із генераторів випадкових чисел, наприклад, RN15. У цьому випадку аргументом функції буде рівномірно розподілене випадкове число з проміжку $[0, 1)$.

Операнд Kn складається з однієї букви (C, D, E, L або M), яка вказує тип функції, і натурального числа n — кількості пар даних у таблиці значень функції. Буква і число пропуском не відокремлюються.

Пари X_i, Y_i — координати точок у списку даних функції. Вони відокремлюються символами «/». При необхідності частину пар переносять на наступні рядки, причому рядки повинні починатись з X_i . У кінці рядків символи «/» можна не ставити. Кількість пар у списку даних функції має збігатись із числом n , вказаним у операнді Kn . Рядки списку даних функції не нумеруються і не можуть містити мітки і коментарі. Величини $X_1, X_2, \dots, X_n$ мають утворювати неспадну послідовність чисел.

Для звертання до функції використовують системний числовий атрибут з класом FN, наприклад, FN\$asd.

Кожен тип функції має свої особливості використання. Буква C в операнді Kn означає, що функція є неперервною, тобто точки, задані парами X_i, Y_i , з'єднуються відрізками й утворюється ламана (рис. 13.1). Значення такої функції GPSS знаходить з допомогою лінійної інтерполяції. Для неперервної функції координати точок X_i, Y_i мають бути числами, причому перші координати X_i повинні бути невід'ємними. Якщо аргумент функції виходить за межі проміжку $[X_1, X_n]$, то функція вертає найближче крайнє значення проміжку. Якщо аргументом функції є генератор випадкових чисел, то повинні виконуватись рівності $X_1 = 0, X_n = 1$.

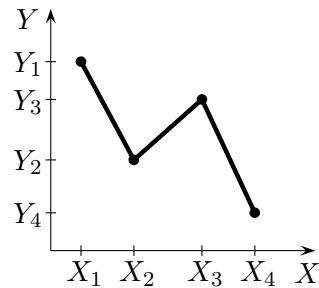


Рис. 13.1.

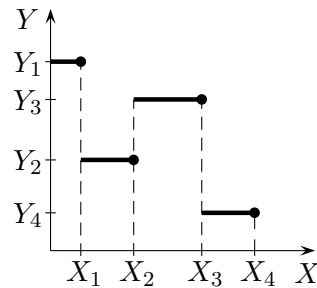


Рис. 13.2.

Приклад неперервної функції, аргументом якої є змінна **zmo**:

```
namef FUNCTION V$zmo,C5
0,1/1,2/2,1.5/4,1/6,-0.5
```

Буква **D** в операнді *Kn* означає, що функція є дискретною. Її графік є кусково-сталим (рис. 13.2). Дискретна функція дорівнює Y_i на цілому півінтервалі $(X_{i-1}, X_i]$ для $i > 1$ і дорівнює Y_1 на проміжку $(-\infty, X_1]$. Для дискретної функції координати точок X_i , Y_i мають бути числами. При обчисленні функції значення X_i у списку даних функції перебираються від найменшого до найбільшого. Для найменшого значення X_i , більшого або рівного значенню аргументу, вертається відповідне йому значення Y_i . Якщо таке значення X_i відсутнє, то отримаємо значення Y_n . Якщо аргументом функції є генератор випадкових чисел, то повинні виконуватись умови $X_1 \geq 0$, $X_n = 1$.

Приклад дискретної функції з випадковим аргументом:

```
namef1 FUNCTION RN10,D5
0.1,1/0.2,2/0.6,1.5/0.8,1/1,2
```

Приклад. Час появи замовлень у двоканальній системі масового обслуговування визначається функцією з випадковим аргументом, графік якої зображено на рис. 13.3. Час обслуговування задається функцією з випадковим аргументом, графік якої наведено на рис. 13.4. Виконати моделювання обслуговування 300 замовлень у такій системі масового обслуговування.

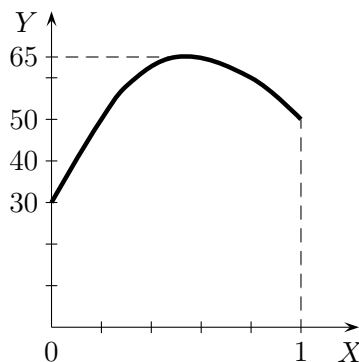


Рис. 13.3.

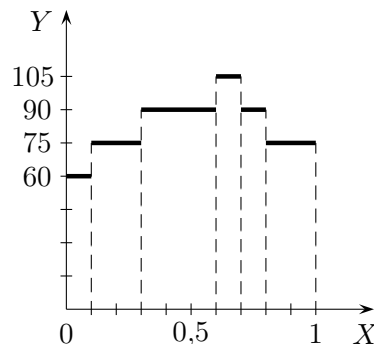


Рис. 13.4.

Розв'язання. Неперервну функцію, графік якої подано на рис. 13.3, назвемо **gen** і закодуємо шістьма точками. Дискретну функцію з рис. 13.4 назвемо **adv** і також закодуємо шістьма точками. Для моделювання системи масового обслуговування використаємо двоканальний пристрій **smo**. Для визначення характеристик черги і тривалості перебування замовлень в моделі застосуємо відповідні блоки і команду для опису таблиці. Наведемо текст моделі:

```
smo STORAGE 2
tab TABLE M1,10,10,100
gen FUNCTION RN10,C6
0,30/0.2,50/0.3,58/0.5,65/0.8,60/1,50
```

```

adv FUNCTION RN11,D6
0.1,60/0.3,75/0.6,90/0.7,95/0.8,90/1,75
GENERATE FN$gen
QUEUE cher
ENTER smo
DEPART cher
ADVANCE FN$adv
LEAVE smo
TABULATE tab
TERMINATE 1
START 300

```

На рис. 13.5 наведено частину звіту. З неї, зокрема, видно, що більшу частину часу черги не було, лише окремі замовлення перебували в черзі довжиною одне замовлення, коефіцієнт завантаження системи масового обслуговування наближено дорівнює 0,72, середній час перебування замовлення в моделі становить 82 одиниці модельного часу. Інші висновки зі звіту читачам пропонується зробити самостійно. ■

QUEUE	MAX	CONT.	ENTRY	ENTRY(0)	AVE.CONT.	AVE.TIME	AVE.(-0)	RETRY
CHER	1	0	301	289	0.005	0.255	6.397	0

STORAGE	CAP.	REM.	MIN.	MAX.	ENTRIES	AVL.	AVE.C.	UTIL.	RETRY	DELAY
SMD	2	1	0	2	301	1	1.447	0.723	0	0

TABLE	MEAN	STD.DEV.	RANGE	RETRY	FREQUENCY	CUM.%
TAB	82.106	10.530		0		
			50.000 -	60.000	26	8.67
			60.000 -	70.000	0	8.67
			70.000 -	80.000	121	49.00
			80.000 -	90.000	109	85.33
			90.000 -	100.000	42	99.33
			100.000 -	110.000	2	100.00

Рис. 13.5

2. Математичні функції і функції для генерації випадкових величин

В арифметичних виразах мова GPSS World дозволяє використовувати наступні математичні функції:

Функція	Опис
ABS(X)	модуль виразу X
SQR(X)	квадратний корінь виразу X
SIN(X)	синус виразу X (в радіанах)
COS(X)	косинус виразу X (в радіанах)
TAN(X)	тангенс виразу X (в радіанах)
ATN(X)	арктангенс виразу X (в радіанах)
EXP(X)	експонента виразу X
LOG(X)	натуральний логарифм виразу X
INT(X)	відкидання дробової частини виразу X

Приклад:

ADVANCE (2#SQR(ABS(P1))) — час затримки транзактів у блоці є подвоєним коренем квадратним від модуля параметра номер 1 транзакта, який зайшов у цей блок.

GPSS World містить 24 спеціальні функції для генерації випадкових величин з різними законами

розподілу. Їх теж можна використовувати в арифметичних виразах. У всіх цих функціях першим аргументом має бути номер генератора випадкових чисел — довільне натуральне число. Основні з функцій для генерації випадкових величин в алфавітному порядку наведені у наступній таблиці. З рештою функцій можна ознайомитись у довідковій системі GPSS World або в книгах [4, 10].

Функція	Опис
$\text{Beta}(R, a, b, \alpha_1, \alpha_2)$	бета-розподіл на проміжку $[a, b]$ з параметрами α_1, α_2
$\text{Binomial}(R, n, p)$	біноміальний розподіл для серії n незалежних випробувань з імовірністю успіху в одному випробуванні p
$\text{DUniform}(R, a, b)$	дискретний (цілочисловий) рівномірний розподіл на проміжку $[a, b]$
$\text{Exponential}(R, a, \bar{x})$	експоненціальний розподіл з середнім значенням $\bar{x}$ і зміщенням a
$\text{Gamma}(R, a, \beta, \alpha)$	гамма-розподіл з формою α , стисненням β і зміщенням a
$\text{Geometric}(R, p)$	геометричний розподіл з імовірністю p (розподіл моделює кількість невдалих спроб, див. § ??)
$\text{Lognormal}(R, a, \mu, \sigma)$	логнормальний розподіл з середнім значенням μ і середньоквадратичним відхиленням σ відповідного нормального закону розподілу, зміщений на a одиниць вправо
$\text{Normal}(R, \mu, \sigma)$	нормальний розподіл з середнім значенням μ і середньоквадратичним відхиленням σ
$\text{Poisson}(R, \lambda)$	розподіл Пуассона з середнім значенням λ
$\text{Triangular}(R, a, b, c)$	трикутний розподіл на проміжку $[a, b]$ з найбільш імовірним значенням c
$\text{Uniform}(R, a, b)$	рівномірний розподіл на проміжку $[a, b]$
$\text{Weibull}(R, a, \beta, \alpha)$	розподіл Вейбулла з формою α , стисненням β і зміщенням a

Для незміщених розподілів замість параметра a у відповідних функціях використовують число 0. Вибору розподілів випадкових величин присвячено розділ ??.

Приклади:

GENERATE ($\text{Exponential}(8, 0, 1/2, 1)$) — в модель вводиться стаціонарний пуассонівський потік транзактів з інтенсивністю 2,1;

ADVANCE ($\text{Lognormal}(9, 0, 1, 0.2)$) — час затримки транзактів є випадковою величиною з логнормальним законом розподілу з середнім значенням 1 і середньоквадратичним відхиленням 0,2 відповідного нормального закону розподілу.

3. Приклади складання моделей

Приклад 1. Телефонна система має три паралельні лінії зв'язку. Нові зовнішні виклики утворюють стаціонарний пуассонівський потік з середнім значенням 70 секунд між викликами. Якщо всі лінії зайняті, то абонент набирає номер ще раз через 5 ± 1 хвилин. Тривалість розмови складає 3 ± 1 хвилини. Потрібно скласти модель на мові GPSS World, яка б дозволила отримати: а) кількість часу, необхідного для здійснення 500 розмов; б) коефіцієнт завантаження телефонної системи; в) табулювання розподілу часу, який потрібен абоненту для встановлення зв'язку і здійснення розмови; г) середню і максимальну кількості абонентів, які чекають звільнення лінії; д) середню і максимальну кількості викликів, потрібні для здійснення абонентом телефонної розмови.

Розв'язання. Оскільки в постановці задачі для вимірювання часу використовуються як хвилини, так і секунди, потрібно весь час у хвилинах перевести в секунди або навпаки. Нехай одиницею модельного часу є секунда. У цій задачі зручно абонентів імітувати транзактами. Для формування стаціонарного пуассонівського потоку вхідних викликів використаємо функцію **Exponential** в операнді блока **GENERATE**. Для моделювання роботи телефонної системи з трьома лініями зв'язку використаємо багатоканальний пристрій **tel** з трьома каналами. Оскільки у випадку, коли зайняті всі лінії зв'язку, абонент отримує відмову і змушений чекати 5 ± 1 хвилин для здійснення повторної спроби виклику, то для перевірки зайнятості телефонної системи скористаємось блоком **GATE**, додатковим блоком затримки **ADVANCE** і блоком безумовного переходу **TRANSFER**. Таким чином, для здійснення моделювання 500 розмов і отримання відповіді на пункти а) і б) достатньо скласти таку модель:

```
tel STORAGE 3
GENERATE (Exponential(8,0,70))
again GATE SNF tel,wait
ENTER tel
ADVANCE 180,60
LEAVE tel
TERMINATE 1
wait ADVANCE 300,60
TRANSFER ,again
START 500
```

Для побудови статистики, згаданої у пункті в), створимо таблицю з ім'ям **trans** з допомогою команди опису **TABLE** і блока **TABULATE**. Часові інтервали в таблиці візьмемо рівними 60 секундам. Для відповіді на пункт г) зручно нетрадиційним способом скористатись чергою, для цього блок **ADVANCE** після мітки **wait** оточимо блоками **QUEUE** і **DEPART**.

Нарешті, для визначення середньої і максимальної кількостей спроб, потрібних для отримання зв'язку абонентом, у параметрі **vyk** кожного транзакта будемо накопичувати кількість здійснених абонентом викликів. Для цього після мітки **again** розмістимо блок **ASSIGN**, який працюватиме в режимі збільшення. Середню і максимальну кількості спроб виклику найлегше знайти з таблиці **calls**, побудованої за параметром **vyk**. Для цього скористаємось командою опису **TABLE**, першим операндом якої вкажемо системний числовий атрибут **P\$vyk**, і блоком **TABULATE**. З таблиці **calls** насправді можна отримати ще більше статистичної інформації. Нижче наводиться остаточно модель:

```
tel STORAGE 3
trans TABLE M1,60,60,200
calls TABLE P$vyk,1,1,100
GENERATE (Exponential(8,0,70))
again ASSIGN vyk+,1
GATE SNF tel,wait
ENTER tel
ADVANCE 180,60
LEAVE tel
TABULATE trans
```

```

TABULATE calls
TERMINATE 1
wait QUEUE chr
ADVANCE 300,60
DEPART chr
TRANSFER ,again
START 500

```



Приклад 2. З допомогою GPSS World здійснити моделювання роботи телефонної системи з попереднього прикладу (с. 58), якщо для абонента максимальна кількість невдалих викликів складає від трьох до шести. Визначити ймовірність того, що абонент не зможе додзвонитись.

Розв’язання. Будемо вважати, що максимальна кількість викликів має рівномірний розподіл серед цілих чисел на проміжку [3, 6]. Збережемо її у параметрі транзакта з ім’ям `maxv`. Для цього використаємо додатковий блок `ASSIGN` з викликом функції `DUniform` після блока `GENERATE` у моделі зі с. 58. Після мітки `wait` додамо перевірку виконання умови `P$vyk < P$maxv` (блок `TEST`). Для визначення кількості абонентів, які не змогли додзвонитись, використаємо блок `SAVEVALUE` в режимі збільшення. Для вилучення цих транзактів з моделі використаємо додатковий блок `TERMINATE` з нульовим операндом. Отримаємо модель:

```

tel STORAGE 3
trans TABLE M1,60,60,200
calls TABLE P$vyk,1,1,100
GENERATE (Exponential(8,0,70))
ASSIGN maxv,(DUniform(7,3,6))
again ASSIGN vyk+,1
GATE SNF tel,wait
ENTER tel
ADVANCE 180,60
LEAVE tel
TABULATE trans
TABULATE calls
TERMINATE 1
wait TEST 1 P$vyk,P$maxv,vidm
QUEUE chr
ADVANCE 300,60
DEPART chr
TRANSFER ,again
vidm SAVEVALUE vidmov+,1
TERMINATE 0
START 500

```

Після моделювання в комірці пам’яті `vidmov` міститься кількість абонентів, які не змогли додзвонитись. Імовірність того, що абонент не зможе додзвонитись, знаходимо за формулою

$$P_{\text{відм}} = \frac{\text{vidmov}}{500 + \text{vidmov}} = \frac{36}{536} = \frac{9}{134} \approx 0,07.$$

Насправді можна було б не додавати в модель блок `SAVEVALUE`, а кількість відмов знайти з таблиці блоків по кількості входів транзактів в останній блок `TERMINATE`. Обчислення ймовірності відмови можна було включити в модель з допомогою додаткового блока `SAVEVALUE` (зробіть самостійно!). Зі статистичного звіту можна отримати всі ті самі характеристики, що й у прикладі 1 зі с. 58 (зробіть це самостійно і порівняйте їх з аналогічними характеристиками з прикладу 1). ■

Лекція 14. Клітинні автомати

План

1. Поняття про клітинні автомати.
2. Модель розвитку організмів на поживних середовищах.
3. Модель розвитку організмів на поживних середовищах за наявності хижаків.

1. Поняття про клітинні автомати

Клітинні автомати є окремим випадком скінченних автоматів. Їх використовують для моделювання динамічної поведінки однорідних двовимірних і тривимірних середовищ у дискретному просторі і часі. При цьому всі фізичні характеристики теж змінюються дискретно. Існує досить розвинена теорія клітинних автоматів. В її основу лягли праці Джона фон Неймана, який у 1948 році ввів поняття клітинного автомата при розробці комп'ютерної моделі біологічної системи.

При створенні клітинних автоматів весь простір розбивається регулярною ґраткою на таблицю комірок, які називають *клітинками*. Кожна з клітинок може знаходитись в одному зі скінченної кількості станів. Кожній клітинці ставиться у відповідність *клітинний автомат*. Найважливішими особливостями клітинних автоматів є наступні:

- стан кожної клітинки оновлюється в результаті виконання послідовності дискретних кроків у часі (тактів);
- змінні стани у кожній клітинці змінюються одночасно (синхронно) відповідно до значень змінних станів цієї клітинки і клітинок деякого її околу на попередньому кроці.

Клітинки для клітинних автоматів на площині найчастіше мають форму квадрата, рідше — трикутника чи шестикутника. Сукупність клітинок утворює клітинний простір.

За оточення клітинки беруть певну кількість сусідніх клітинок. Зокрема, для квадрата оточення можуть бути чотири або вісім сусідніх клітинок (рис. 14.1). Однак можливі ситуації, коли стан клітинки залежить від стану більш віддалених клітинок.

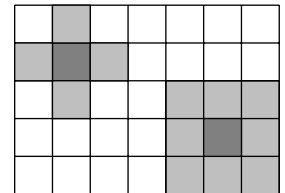


Рис. 14.1

Для того щоб виключити особливий вигляд оточення для точок, що лежать на межі області, часто вводять так звані *крайові умови циклічного типу*. Вони означають, що для крайнього лівого стовпця клітинок сусіднім зліва вважається крайній правий стовпець, а для крайнього правого — крайній лівий стовпець. Аналогічно, для крайнього верхнього рядка клітинок сусіднім зверху вважається крайній нижній рядок, і навпаки. Таким чином, можна вважати, що клітинки задаються на поверхні тора.

Функцію переходів для клітинних автоматів називають *правилами* клітинних автоматів.

Найвідомішим клітинним автоматом є автомат «Життя», розроблений у 1970 році Джоном Конвеєм. Множина правил для цього автомата полягає у наступному:

- клітинка може перебувати у двох станах — активному і пасивному;
- в якості оточення розглядається вісім сусідніх клітинок;
- якщо в оточенні пасивної клітинки є рівно три активних клітинки, то клітинка стає активною («народжується»);
- якщо в оточенні активної клітинки є дві чи три активних клітинки, то клітинка залишається активною («виживає»);
- якщо в оточенні активної клітинки є менше двох чи більше трьох активних клітинок, то вона стає пасивною («помирає» від самотності чи перенаселення).

У початковий момент часу є деякий розподіл активних і пасивних клітинок. Використовуючи наявний розподіл клітинок, визначаємо стан клітинок на наступному кроці і т. д. В залежності від вигляду початкового розподілу приходимо до різних еволюцій життя.

Розглянемо приклад. Для початкового розподілу активних клітинок автомата «Життя» з рис. 14.2 на рис. 14.3–14.5 зображено розподіли активних клітинок для четвертого, восьмого і одинадцятого кроків відповідно. На дванадцятому і всіх наступних кроках зберігатиметься розподіл активних клітинок з рис. 14.5.

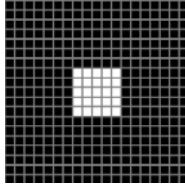


Рис. 14.2

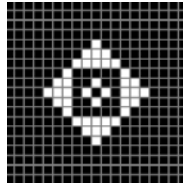


Рис. 14.3

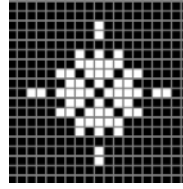


Рис. 14.4

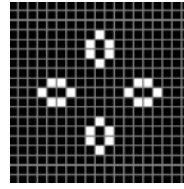


Рис. 14.5

2. Модель розвитку організмів на поживних середовищах

Поряд зі скінченними детермінованими автоматами можна розглядати скінченні ймовірнісні автомати, які вводяться подібно до детермінованих. Важливим класом скінченних ймовірнісних автоматів є клітинні автомати з випадковими процесами. Такі клітинні автомати дозволяють моделювати більш складні системи. Одним з найцікавіших застосувань цих клітинних автоматів є моделювання біологічних систем.

Розглянемо приклад моделювання біологічної системи з допомогою клітинних автоматів. Біологічною системою є двовимірне поживне середовище з одноклітинними чи іншими живими організмами, які на ньому розвиваються. Модель опишемо сукупністю правил:

1. Простір клітинних автоматів є полем $n \times n$ квадратних клітинок.
2. Окіл клітинки утворюють вісім сусідніх клітинок.
3. Кожній клітинці відповідає значення p ступеня поживності (енергоємності) розчину, яке може змінюватись від 0 до p_{max} .
4. Величина поживності (енергоємності) розчину клітинки за такт часу збільшується на величину Δp , але не може перевищувати p_{max} .
5. Клітинка може бути вільною або містити один живий організм.
6. Організм бере енергію з поживного розчину клітинки, в якій він знаходиться, зменшуючи її поживність і збільшуючи свій запас енергії на величину ΔE_{pl} за такт, але при цьому не може взяти енергії з клітинки більше, ніж її там є.
7. Максимальна можлива кількість енергії, яку може накопичити організм, не перевищує E_{max} .
8. Наприкінці кожного такту організм намагається перейти в сусідню вільну клітинку, вибираючи її довільним чином. Якщо вільної клітинки в околі немає, то організм залишається в тій самій клітинці.
9. Починаючи з віку L_{zr} тактів, організм вважається зрілим. Протягом такту до досягнення зрілості організм витрачає на життя ΔE_{mi} енергії, а після досягнення зрілості — ΔE_{zrmi} енергії.
10. Зрілий організм може створювати собі подібних діленням, витрачаючи додатково ΔE_{di} енергії при кожному поділі. При цьому батьківська особина переходить у сусідню вільну клітинку, а нова залишається у попередній клітинці з початковим запасом енергії E_0 . Якщо в околі клітинки вільних клітинок немає або запас енергії організму не перевищує ΔE_{di} , то ділення не відбувається.
11. Час життя окремої особини не перевищує L_{max} тактів. Якщо час життя організму досягає тривалості життя L_{max} або запас енергії знижується до нуля, то організм помирає.
12. Початкова поживність клітинок задається рівномірним законом розподілу в межах від 0 до p_{max} . Загальну поживність всіх клітинок на такті t позначимо через $P_A(t)$, тоді відносна поживність $P(t) = \frac{P_A(t)}{n^2 p_{max}}$.
13. Початковий закон розподілу організмів у клітинному просторі є рівномірним. Початкова кількість організмів — N_0 . Початковий запас енергії кожної особини дорівнює E_0 , а вік вважається нульовим. Загальну кількість організмів на такті t позначимо через $N_A(t)$, тоді відносна чисельність організмів $N(t) = \frac{N_A(t)}{n^2}$.

Запрограмувати ці правила будь-якою мовою програмування нескладно. У результаті можна отримати залежності P і N від часу та фазову траєкторію (залежність N від P). Наприклад, у випадку, коли $n = 1000$, $p_{max} = 10$, $\Delta p = 1$, $\Delta E_{pl} = 6$, $E_{max} = 30$, $\Delta E_{mi} = 3$, $\Delta E_{zrmi} = 2$, $\Delta E_{di} = 4$, $E_0 = 4$, $L_{max} = 20$, $L_{zr} = 4$, $N_0 = 0,3n^2$, графіки цих залежностей наведені на рис. 14.6–14.8.

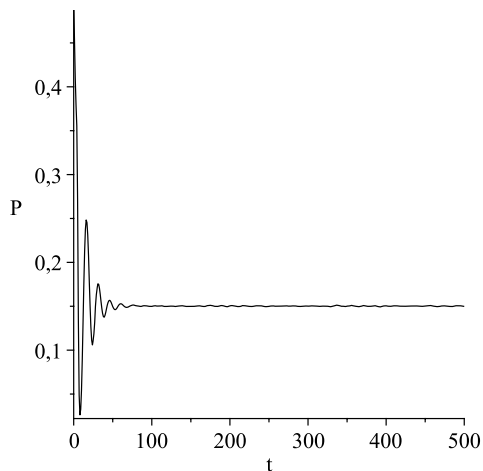


Рис. 14.6

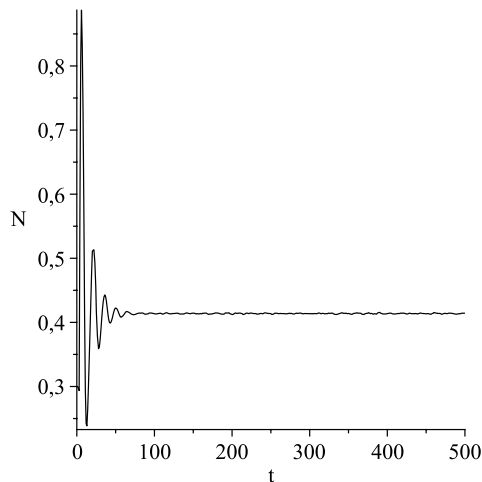


Рис. 14.7

Бачимо, що мають місце згасаючі коливання відносного рівня поживності середовища і відносної чисельності організмів навколо їхніх середніх значень $\bar{P} \approx 0,15$, $\bar{N} \approx 0,41$. Крім того, можна встановити, що середня тривалість життя організмів складає $\bar{L} \approx 4,7$ тактів, хоча є особини, які доживають до глибокої старості (20 тактів). Згасаючі коливання виявляються подібними до тих, що виникають при розв'язуванні модифікованої системи диференціальних рівнянь Лотки–Вольтерри з конкурентною боротьбою.

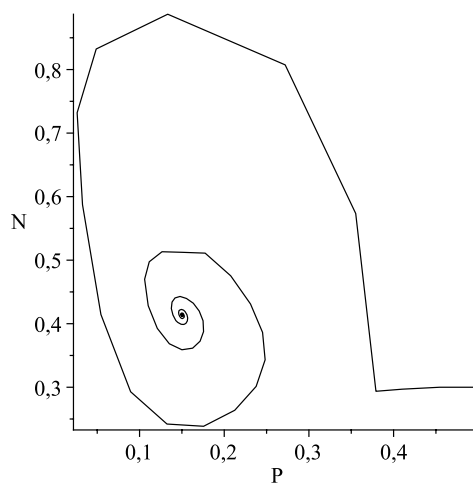


Рис. 14.8

Модель можна модифікувати наступним чином: організм завжди намагається перейти у сусідню вільну клітинку з найбільшим рівнем енергоємності. Якщо клітинки в околі мають менший запас енергії, то особина залишається у тій самій клітинці. Крім того, при діленні організм вибирає сусідню вільну клітинку з найбільшим рівнем енергоємності. У цьому випадку для тих самих числових даних отримуємо подібні графіки.

У цьому випадку спочатку відбуваються згасаючі коливання відносного рівня поживності середовища і відносної чисельності організмів навколо їхніх середніх значень $\bar{P} \approx 0,085$, $\bar{N} \approx 0,43$, після чого відносний рівень поживності середовища і відносна чисельність організмів досягають своїх середніх значень. Крім того, середня чисельність організмів у цьому випадку трохи більша, середня тривалість їхнього життя більша ($\bar{L} \approx 6,6$), а середня поживність середовища значно менша, ніж у випадку «нерозумної» поведінки організмів. Таким чином, «розумна» поведінка сприяє виживанню організмів.

3. Модель розвитку організмів на поживних середовищах за наявності хижаків

Додамо до попередньої моделі хижаків, життя яких підпорядковане наступним правилам:

1. Клітинка може бути вільною, містити один звичайний організм або одного хижака.
2. Їжею для хижаків є живі організми з попередньої моделі. Хижак може напасти на свою жертву, якщо вона знаходиться в його околі, тобто в одній з восьми сусідніх клітинок. Якщо є кілька потенційних жертв, то одна з них вибирається довільним чином. При цьому жертва гине з імовірністю P_X , а хижак переміщується в її клітинку, підвищуючи свій запас енергії на величину енергетичної цінності жертви ΔEX_{pl} . Імовірність загибелі жертви не дорівнює одиниці, бо вважається, що вона має деякі засоби захисту. Якщо жертва залишається живою, то гине хижак.
3. Якщо в околі хижака немає потенційних жертв, то хижак переміщується на сусідню вільну клітинку, вибираючи напрям довільним чином.
4. Інші характеристики життя хижаків є аналогічними до відповідних характеристик для звичайних організмів: хижак може накопичити не більше, ніж EX_{max} енергії, за один такт життя хижак до досягнення зрілості витрачає ΔEX_{mi} енергії, а після досягнення зрілості — ΔEX_{zrmi} енергії, максимальна тривалість життя хижака становить LX_{max} тактів, вік зрілості — LX_{zr} тактів, енергія, що витрачається при діленні, — ΔEX_{di} , початковий запас енергії хижака — EX_0 .
5. Якщо час життя хижака перевищує максимально можливий час LX_{max} або запас енергії зменшується до нуля, то хижак помирає.
6. Початковий розподіл хижаків у просторі є рівномірним. Початкова кількість хижаків — X_0 . Загальну кількість хижаків на такті t позначимо через $X_A(t)$, тоді відносна чисельність хижаків $X(t) = \frac{X_A(t)}{n^2}$.

Запрограмувати ці правила будь-якою мовою програмування також нескладно. У результаті можна отримати залежності P , N і X від часу та залежності N від P і X від N . Нехай, наприклад, $n = 1000$, $p_{max} = 10$, $\Delta p = 1$, $\Delta E_{pl} = 6$, $E_{max} = 30$, $\Delta E_{mi} = 3$, $\Delta E_{zrmi} = 2$, $\Delta E_{di} = 4$, $E_0 = 4$, $L_{max} = 20$, $L_{zr} = 4$, $N_0 = 0,3N$, $EX_{max} = 40$, $\Delta EX_{mi} = 3,5$, $\Delta EX_{zrmi} = 3$, $\Delta EX_{di} = 5$, $LX_{max} = 25$, $LX_{zr} = 6$, $P_X = 0,85$, $EX_0 = 5$, $X_0 = 0,05n^2$,

$$\Delta EX_{pl} = \begin{cases} 3L + \frac{E}{5}, & L < L_{zr}, \\ 10 + \frac{E}{5}, & L \geq L_{zr}, \end{cases}$$

де L — вік з'їденої жертви, E — її запас енергії. У цьому випадку також можна отримати графіки залежностей.

Тут також мають місце згасаючі коливання відносного рівня поживності середовища і відносних чисельностей нехижих і хижих організмів навколо їхніх середніх значень $\bar{P} \approx 0,40$, $\bar{N} \approx 0,33$, $\bar{X} \approx 0,07$. Крім того, можна встановити, що середня тривалість життя нехижих організмів складає $\bar{L} \approx 3,5$ тактів, хоча є особини, які доживають до глибокої старості (20 тактів), а середня тривалість життя хижаків — $\bar{L}_X \approx 3,7$ тактів, хоча зустрічаються особини, які доживають до глибокої старості (25 тактів). Згасаючі коливання виявляються подібними до тих, що виникають при інтегруванні модифікованої системи диференціальних рівнянь Лотки–Вольтерри з конкурентною боротьбою.

Модель можна модифікувати, як раніше: організм завжди намагається перейти в сусідню вільну клітинку з найбільшим рівнем енергоємності. Якщо клітинки в околі мають менший запас енергії, то особина залишається у тій самій клітинці. Крім того, при діленні організм вибирає сусідню вільну клітинку з найбільшим рівнем енергоємності. Хижаки теж вибирають собі жертву з найбільшим рівнем енергетичної цінності.

У випадку «розумної» поведінки нехижі організми утворюють фронти, які рухаються на вільні області, причому хижаки рухаються за фронтами і поїдають тих, хто відстає. Відсутність хижаків перед фронтами пояснюється великою кількістю зустрічей хижаків з нехижими організмами, які мають певні засоби захисту, що не залишає для хижаків жодних шансів.

Лекція 15. Класичні мережі Петрі, їхні властивості

План

1. Поняття класичних мереж Петрі.
2. Формальне означення мереж Петрі і їхнє матричне подання.
3. Властивості мереж Петрі.

1. Поняття класичних мереж Петрі

Мережі Петрі — це ще один з видів математичного моделювання. Їх використовують для моделювання асинхронних паралельних систем і процесів. Вперше мережі Петрі були запропоновані в докторській дисертації німецького математика Карла Адама Петрі в 1962 році.

Області застосування мереж Петрі включають дослідження і моделювання телекомунікаційних мереж, обчислювальних, виробничих і організаційних систем. Зокрема, мережі Петрі використовуються при проектуванні мобільних телефонів, систем електронних розрахунків, систем керування рухом поїздів, систем безпеки та контролю доступу, при плануванні військових операцій і навчань.

Ми лише оглядово розглянемо мережі Петрі, перш за все, у найпростішому класичному розумінні. Повний виклад теорії мереж Петрі вимагає окремої книги.

Мережа Петрі є орієнтованим дводольним графом, який має чотири базові елементи: вузли, переходи, дуги і маркери. У літературі вузли інколи називають позиціями або вершинами, дуги — ребрами, а маркери — фішками. Дводольним називається граф, який має дві множини вершин і не має ребер, які з'єднують вершини однієї множини. Вузли визначають стан, в якому може знаходитись мережа чи її частина. Переходи — це активні елементи мережі, які визначають дії, що будуть виконуватись при спрацюванні переходів. Дуги з'єднують елементи протилежних типів: вузол з переходом і перехід з вузлом. Дуги не можуть з'єднувати безпосередньо вузли чи переходи. Вузли позначають кружками, переходи — прямокутниками або рисками, дуги — лініями зі стрілками на кінцях, маркери — крапками. Маркери знаходяться всередині вузлів і переміщуються в мережі в результаті спрацювання переходів. При великій кількості маркерів у вузлі відображають не крапки, а число відповідних маркерів. Дуги можуть бути простими або кратними. Розглянемо спочатку випадок простих дуг.

У випадку простих дуг *перехід* називається *дозволеним (збудженим)*, якщо всі його вхідні вузли містять маркери. На рис. 15.1 і 15.2 наведено приклади дозволеного і недозволеного переходів.

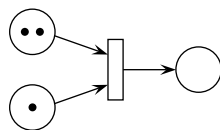


Рис. 15.1. Дозволений перехід

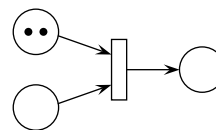


Рис. 15.2. Недозволений перехід

Дозволений перехід може спрацювати. У випадку простих дуг при *спрацюванні (запуску) переходу* вилучається по одному маркеру з кожного його вхідного вузла і розміщується по одному маркеру у всіх його вихідних вузлах (рис. 15.3).

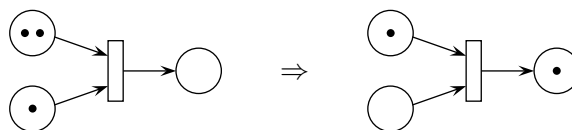


Рис. 15.3. Графічне зображення спрацювання переходу

Для моделювання реальних об'єктів буває зручно використовувати кратні дуги. Графічно кратні дуги, як правило, зображають як одну дугу, записуючи біля неї її кратність (рис. 15.4). У цьому випадку граф мережі насправді є мультиграфом.



Рис. 15.4

Для дозволеності переходу у випадку кратних дуг потрібно, щоб у кожному його вхідному вузлі було принаймні стільки маркерів, якою є кратність відповідної дуги.

При спрацюванні такого переходу з кожного його вхідного вузла вилучаються маркери в кількості, відповідній кратності дуги, а в кожному вихідному вузлі розміщується стільки маркерів, якою є кратність дуги від переходу до цього вузла (рис. 15.5).



Рис. 15.5

У класичній мережі Петрі перехід спрацьовує миттєво. Одночасно може бути кілька дозволених переходів, але спрацьовує з них лише один, який вибирається довільним чином. Довільність вибору переходу для спрацювання визначає недетермінований характер поведінки мережі. Після спрацювання переходу в мережі знову може бути один або кілька дозволених переходів, з яких на наступному кроці спрацьовує тільки один.

Мережа Петрі описує множину різних допустимих варіантів поведінки модельованих систем. Може бути, що спрацювання якогось переходу (чи послідовності переходів) приведе систему в тупикову ситуацію, коли не зможе відбутись жоден перехід. Побудова й аналіз мереж Петрі дає змогу виявити всі варіанти поведінки реальних систем і тупикові ситуації.

У мережах Петрі можуть бути петлі. Але мережу Петрі з петлею завжди можливо подати у вигляді мережі без петлі, ввівши додатковий вузол і перехід (рис. 15.6).

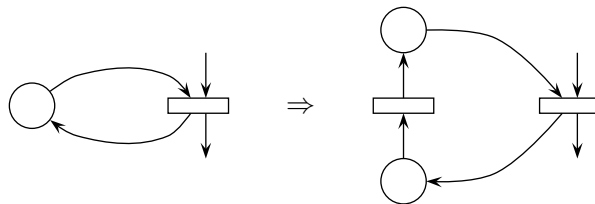


Рис. 15.6

При розумінні функціонування мережі Петрі вузли разом з маркерами в них розглядають як певні умови, а переходи — як події. Виникнення певної комбінації маркерів у вузлах приводить до настання деякої події, яка у свою чергу викликає зміну стану умов мережі. Стан мережі визначається сукупністю маркерів кожного вузла. Цю сукупність маркерів називають *розміткою*.

Класичні мережі Петрі є дискретно-ймовірнісними моделями.

Приклад. Розглянемо мережу Петрі N_1 для передачі односторонніх повідомлень і підтвердження їхнього прийняття (рис. 15.7).

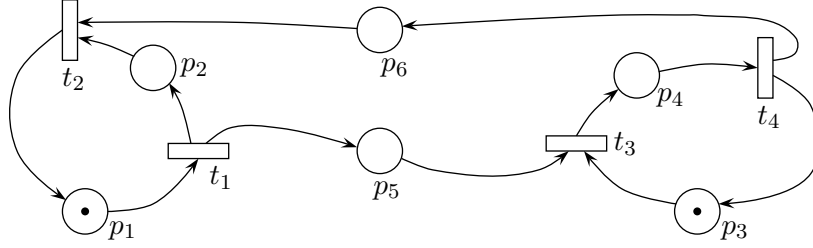


Рис. 15.7. Мережа N_1

У цій мережі перехід t_1 моделює відправлення повідомлення, перехід t_3 — прийом повідомлення, перехід t_4 — відправлення підтвердження, перехід t_2 — отримання підтвердження. Переходи спрацюють саме у цій послідовності, утворюючи цикл $t_1 t_3 t_4 t_2$.

2. Формальне означення мереж Петрі і їхнє матричне подання

Є кілька способів формального означення мереж Петрі. Наведемо один з них. *Графом мережі Петрі* називають трійку $G = (P, T, F)$, де $P = \{p_i\}$, $i = 1, 2, \dots, m$, — непорожня скінченна множина вузлів, $T = \{t_j\}$, $j = 1, 2, \dots, n$, — непорожня скінченна множина переходів, F — функція інцидентності, що задає зв'язок між елементами множин P і T , тобто $F : P \times T \cup T \times P \rightarrow N_0$, де $N_0 = \mathbb{N} \cup \{0\}$. Значення функції F визначають наявність і кратність дуг.

Розміткою (маркуванням) мережі називають відображення $M : P \rightarrow N_0$, яке задає розміщення маркерів у вузлах. Тоді *мережа Петрі* — це четвірка $N = (P, T, F, M_0)$ або $N = (G, M_0)$, де M_0 — початкова розмітка. Значення розмітки для j -го вузла називають j -м *значенням розмітки*.

Мережу Петрі N_1 з рисунка 15.7 можна формально записати так:

$$\begin{aligned} N_1 &= (P, T, F, M_0), \\ P &= \{p_1, p_2, p_3, p_4, p_5, p_6\}, \quad T = \{t_1, t_2, t_3, t_4\}, \\ F(p_1, t_1) &= 1, \quad F(p_2, t_2) = 1, \quad F(p_3, t_3) = 1, \quad F(p_4, t_4) = 1, \\ F(p_5, t_3) &= 1, \quad F(p_6, t_2) = 1, \quad F(t_1, p_2) = 1, \quad F(t_1, p_5) = 1, \\ F(t_2, p_1) &= 1, \quad F(t_3, p_4) = 1, \quad F(t_4, p_3) = 1, \quad F(t_4, p_6) = 1, \\ M_0(p_1) &= 1, \quad M_0(p_3) = 1. \end{aligned}$$

Для скорочення запису вказані лише ненульові значення функцій F і M_0 . У випадку кратних дуг відповідні значення функції F були б більшими за одиницю.

Поряд з графічним і теоретико-множинним визначенням мереж Петрі широко використовується їхнє матричне подання. Функцію інцидентності F подають матрицями B і D :

$$\begin{aligned} B &= (b_{ij}), \quad i = \overline{1, m}, \quad j = \overline{1, n}, \quad b_{ij} = F(p_i, t_j), \\ D &= (d_{ij}), \quad i = \overline{1, m}, \quad j = \overline{1, n}, \quad d_{ij} = F(t_j, p_i). \end{aligned}$$

Матрицю B називають *матрицею входів*, а матрицю D — *матрицею виходів*. Для мереж без петель замість матриць B і D можна використовувати одну матрицю $C = D - B$. Її називають *матрицею змінювань*.

Розмітку мережі подають вектором

$$\overline{M} = (M(p_1) \quad M(p_2) \quad \dots \quad M(p_m))^T,$$

який частіше заради компактності будемо записувати у вигляді $\overline{M} = (M(p_1), M(p_2), \dots, M(p_m))$, продовжуючи розуміти його як вектор-стовпець. Тоді мережу Петрі однозначно задає набір $(B, D, \overline{M}_0)$ або у випадку мережі без петель — $(C, \overline{M}_0)$, де $\overline{M}_0$ — вектор початкової розмітки.

Побудуємо матричне подання мережі Петрі N_1 , зображеної на рис. 15.7:

$$N_1 = (B, D, \overline{M}_0),$$

$$B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \quad D = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad \overline{M}_0 = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Оскільки мережа не має петель, то можливе подання $N_1 = (C, \overline{M}_0)$, де

$$C = \begin{pmatrix} -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \\ 1 & 0 & -1 & 0 \\ 0 & -1 & 0 & 1 \end{pmatrix}. \quad (15.1)$$

3. Властивості мереж Петрі

Використовуватимемо наступні допоміжні позначення: дозволений (збуджений) перехід $t \in T$ у розмітці $\overline{M}$ позначимо через $\overline{M} \xrightarrow{t}$, спрацювання переходу $t \in T$ у розмітці $\overline{M}$ позначимо через $\overline{M} \xrightarrow{t} \overline{M}'$. Аналогічні позначення будемо використовувати також для послідовностей переходів $\sigma \in T^*$ ($\sigma = t_{j_1} t_{j_2} \dots t_{j_k}$, T^* — позначення для послідовності множин T): послідовність $\sigma \in T^*$ дозволена у розмітці $\overline{M}$, якщо $\overline{M} \xrightarrow{\sigma}$, послідовність $\sigma \in T^*$ спрацює у розмітці $\overline{M}$, якщо $\overline{M} \xrightarrow{\sigma} \overline{M}'$.

Наприклад, якщо для мережі N_1 (рис. 15.7)

$$\overline{M} = (1, 0, 1, 0, 0, 0), \quad \sigma = t_1 t_3,$$

то $\overline{M} \xrightarrow{\sigma}$, $\overline{M} \xrightarrow{\sigma} \overline{M}'$, де $\overline{M}' = (0, 1, 0, 1, 0, 0)$.

Під властивостями мереж Петрі розуміють набір корисних чи небажаних характеристик, притаманних досліджуваній мережі. Тому властивості можна умовно поділити на позитивні і негативні. Крім того, розрізняють поведінкові і структурні властивості мереж. Поведінкові властивості пов'язані з конкретною початковою розміткою, а структурні властивості мають місце незалежно від початкової розмітки.

Розглянемо основні властивості мереж Петрі.

1. *Досяжність розмітки*. Розмітка $\overline{M}$ є досяжною для заданої мережі $N = (G, \overline{M}_0)$, якщо існує така дозволена у початковій розмітці $\overline{M}_0$ послідовність переходів σ , що $\overline{M}_0 \xrightarrow{\sigma} \overline{M}$. Множину досяжних розміток мережі позначають через $R(N)$, $R(N) = \{\overline{M} \mid \exists \sigma \in T^* : \overline{M}_0 \xrightarrow{\sigma} \overline{M}\}$.

2. *Потенційна живість*. Перехід $t \in T$ називають потенційно живим у розмітці $\overline{M}$, якщо існує дозволена послідовність переходів, яка містить цей перехід: $\exists \sigma \in T^* : t \in \sigma, \overline{M} \xrightarrow{\sigma}$.

3. *Живість*. Перехід називають живим, якщо він є потенційно живим у будь-якій досяжній розмітці, тобто перехід $t \in T$ живий, якщо $\forall \overline{M} \in R(N) \exists \sigma \in T^* : t \in \sigma, \overline{M} \xrightarrow{\sigma}$. Мережа Петрі жива, якщо живі всі її переходи.

4. *Термінальність*. Розмітку $\overline{M}$ мережі Петрі називають t -термінальною, якщо перехід $t \in T$ не є в ній потенційно живим. Розмітку $\overline{M}$ мережі називають термінальною, якщо вона є t -термінальною для всіх переходів мережі. Термінальну розмітку називають також тупиковою. У ній не є дозволеним жоден перехід.

5. *Обмеженість*. Вузол $p \in P$ називають обмеженим (l -обмеженим), якщо існує таке ціле число l , що для будь-якої досяжної розмітки мережі кількість маркерів у вузлі p не перевищує числа l , тобто $\exists l \in \mathbb{Z} \quad \forall \overline{M} \in R(N) : M(p) \leq l$. Мережа Петрі є обмеженою, якщо обмежені всі її вузли.

6. *Безпечність*. Вузол $p \in P$ називають безпечним, якщо для довільної досяжної розмітки мережі кількість маркерів у вузлі p не перевищує одиниці, тобто дорівнює 0 або 1. Мережа Петрі є безпечною, якщо безпечні всі її вузли.

7. *Повторюваність*. Послідовність переходів σ називають повторюваною, якщо її можна запустити довільну кількість разів.

8. *Стаціонарна повторюваність*. Послідовність переходів σ називають стаціонарно повторюваною, якщо вона є повторюваною і переводить мережу в ту саму розмітку: $\overline{M} \xrightarrow{\sigma} \overline{M}$.

9. *Оборотність*. Мережу Петрі називають оборотною, якщо з довільної досяжної розмітки досяжною є початкова розмітка, тобто $\forall \overline{M} \in R(G, \overline{M}_0) : \overline{M}_0 \in R(G, \overline{M})$.

10. *Стійкість*. Мережу Петрі називають стійкою, якщо спрацювання одного з будь-яких двох дозволених переходів не може зробити недозволеним другий перехід.

11. *Консервативність*. Консервативною називають мережу, яка зберігає зважену суму маркерів відносно певного вагового вектора $\overline{w} = (w_1 \ w_2 \ \dots \ w_m)$ з натуральними елементами, тобто $\overline{w}\overline{M} = \overline{w}\overline{M}_0 = \text{const}$, де $\overline{M}$ — довільна розмітка, $\overline{M}_0$ — початкова розмітка.

12. *Строго консервативність*. Строго консервативною називають мережу, сума маркерів якої є сталою, тобто консервативну мережу відносно одиничного вагового вектора.

Наприклад, для мережі Петрі N_1 (рис. 15.7) множина досяжних розміток має вигляд

$$R(N_1) = \{(1, 0, 1, 0, 0, 0), (0, 1, 1, 0, 1, 0), (0, 1, 0, 1, 0, 0), \\ (0, 1, 1, 0, 0, 1)\}.$$

Безпосередньо видно, що мережа N_1 є живою, обмеженою, безпечною, оборотною, стійкою і має стаціонарно повторювану послідовність переходів $t_1 t_3 t_4 t_2$. Консервативність цієї мережі буде досліджено у наступному параграфі.

Лекція 16. Матричний аналіз мереж Петрі, некласичні мережі Петрі

План

1. Система станів і фундаментальне рівняння мереж Петрі.
2. Інваріанти і їх застосування для матричного аналізу мереж Петрі.
3. Некласичні мережі Петрі.

1. Система станів і фундаментальне рівняння мереж Петрі

Індикатором переходу t_j назовемо вектор-стовпець

$$\bar{u}_{t_j} = (u_1 \quad u_2 \quad \dots \quad u_n)^T, \quad u_k = \begin{cases} 1, & k = j, \\ 0, & k \neq j. \end{cases}$$

Вхідні дуги переходу t_j можна подати як $t_j^- = B\bar{u}_{t_j}$, а вихідні — як $t_j^+ = D\bar{u}_{t_j}$, де B — матриця входів, а D — матриця виходів.

Розглянемо приклад для мережі N_1 (рис. 15.7):

$$\bar{u}_{t_2} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \quad t_2^- = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}, \quad t_2^+ = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Умову дозволеності переходу $t \in T$ у розмітці $\bar{M}$ можна подати нерівністю $\bar{M} \geq B\bar{u}_t$ (ця нерівність означає, що кожен елемент вектора-стовпця $\bar{M}$ є не меншим за відповідний елемент вектора-стовпця t^-). Спрацювання переходу $t \in T$ у розмітці $\bar{M}$ подається співвідношенням $\bar{M}' = \bar{M} - B\bar{u}_t + D\bar{u}_t$. Отже, формальний опис процесу функціонування мережі Петрі за відсутності петель дає наступна система:

$$\begin{cases} \bar{M}_{k+1} = \bar{M}_k + C\bar{u}_t, \\ \bar{M}_k \geq B\bar{u}_t, \quad t \in T, \end{cases} \quad (16.1)$$

де $\bar{M}_k$ і $\bar{M}_{k+1}$ — розмітки мережі Петрі на k -му і $(k+1)$ -му кроках, а C — матриця змінювань. Систему (16.1) називають *системою станів* мережі Петрі.

Система станів мережі Петрі формально подає динаміку функціонування мережі, однак вона містить векторну нерівність і є рекурентною. Подамо з допомогою системи станів розмітку $\bar{M}_k$ на довільному кроці k , отриману в результаті спрацювання дозволеної послідовності переходів σ , через початкову розмітку $\bar{M}_0$, $\bar{M}_0 \xrightarrow{\sigma} \bar{M}_k$:

$$\bar{M}_k = \bar{M}_0 + C \sum_{t \in \sigma} \bar{u}_t.$$

Ввівши позначення

$$\bar{\sigma} = \sum_{t \in \sigma} \bar{u}_t,$$

отримаємо рівняння

$$\bar{M}_k = \bar{M}_0 + C\bar{\sigma}, \quad (16.2)$$

яке називають *фундаментальним рівнянням* мережі Петрі. Вектор $\bar{\sigma}$ називають *вектором лічби спрацювань* послідовності σ .

При побудові фундаментального рівняння відбувається втрата інформації, проте це рівняння є лінійним, а тому його розв'язок відносно нескладно знайти. Розв'язність фундаментального рівняння в цілих невід'ємних числах є необхідною умовою досяжності розмітки: *якщо розмітка $\bar{M}$ досяжна в мережі, то фундаментальне рівняння має цілий невід'ємний розв'язок $\bar{\sigma}$, такий що $\bar{M}_0 \xrightarrow{\sigma} \bar{M}$.*

Однак існування цілого невід'ємного розв'язку рівняння (16.2) є недостатнім для досяжності розмітки. Розглянемо приклад мережі Петрі N_2 з рис. 16.1.

Знайдемо матриці входів, виходів і змінювань (§ 2):

$$B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad D = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad C = \begin{pmatrix} -1 & 0 \\ 1 & -1 \\ -1 & 1 \end{pmatrix}.$$

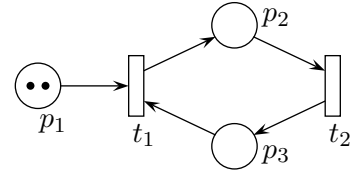


Рис. 16.1. Мережа N_2

З допомогою рівняння (16.2) перевіримо виконання необхідної умови досяжності розмітки $\overline{M}' = (1, 0, 0)$ з початкової розмітки $\overline{M}_0 = (2, 0, 0)$:

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} -1 & 0 \\ 1 & -1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} \sigma_1 \\ \sigma_2 \end{pmatrix}.$$

Ця система має розв'язок $\sigma_1 = \sigma_2 = 1$, а тому виконується необхідна умова досяжності розмітки $\overline{M}'$. У той самий час, початкова розмітка мережі Петрі N_2 є тупиковою, бо в ній недозволений жодний перехід. Отже, розмітка $\overline{M}'$ не є досяжною.

2. Інваріанти і їх застосування для матричного аналізу мереж Петрі

Інваріантом вузлів мережі або *p-інваріантом* називають ненульовий вектор-рядок $\overline{x} = (x_1 \ x_2 \ \dots \ x_m)$, який складається з невід'ємних цілих чисел і є розв'язком рівняння

$$\overline{x}C = 0. \quad (16.3)$$

Інваріантом переходів мережі або *t-інваріантом* називають ненульовий вектор-стовпець $\overline{y} = (y_1 \ y_2 \ \dots \ y_n)^T$, який складається з невід'ємних цілих чисел і є розв'язком рівняння

$$C\overline{y} = 0. \quad (16.4)$$

Інваріанти вузлів і переходів визначаються неоднозначно. Якщо існує інваріант вузлів, який складається лише з натуральних чисел, то мережу Петрі називають *p-інваріантною*. Аналогічно, *t-інваріантною* називають мережу Петрі, у якій існує інваріант переходів, що складається лише з натуральних чисел.

Теорема 1. Мережа Петрі є консервативною тоді і тільки тоді, коли вона є *p-інваріантною*.

Доведення. Нехай мережа Петрі є *p-інваріантною*, тобто існує інваріант вузлів $\overline{x}$, який складається лише з натуральних чисел. Помножимо обидві частини фундаментального рівняння (16.2) для довільної досяжної розмітки $\overline{M}$ зліва на вектор $\overline{x}$:

$$\overline{x}\overline{M} = \overline{x}\overline{M}_0 + \overline{x}C\overline{\sigma}. \quad (16.5)$$

Внаслідок рівності (16.3) маємо

$$\overline{x}\overline{M} = \overline{x}\overline{M}_0, \quad (16.6)$$

звідки випливає консервативність мережі Петрі.

Нехай тепер мережа Петрі є консервативною, тобто для довільної досяжної розмітки $\overline{M}$ виконується рівність (16.6) з ваговим вектором $\overline{x}$, всі елементи якого є натуральними. Помноживши обидві частини фундаментального рівняння (16.2) зліва на вектор $\overline{x}$, отримаємо рівняння (16.5), з якого випливає рівність $\overline{x}C\overline{\sigma} = 0$ для кожного вектора $\overline{\sigma}$. Отже, виконується рівність (16.3) для вектора $\overline{x}$ з натуральними елементами. Тому мережа Петрі є *p-інваріантною*. Теорему доведено.

Теорема 2. Кожна консервативна мережа Петрі є обмеженою.

Доведення. Для кожної консервативної мережі Петрі виконується рівність (16.6) з ваговим вектором $\bar{x}$, всі елементи якого є натуральними. Виберемо довільний вузол $q \in P$ і розпишемо ліву частину рівності (16.6):

$$x_q M(q) + \sum_{p \neq q} x_p M(p) = \bar{x} \overline{M}_0.$$

Враховуючи невід'ємність доданків, маємо:

$$x_q M(q) \leq \bar{x} \overline{M}_0,$$

звідки

$$M(q) \leq \frac{\bar{x} \overline{M}_0}{x_q}.$$

Теорему доведено.

Наслідок 1. Кожна p -інваріантна мережа Петрі є обмеженою.

Наслідок 2. Необмежена мережа Петрі не є консервативною.

Теорема 3. Якщо в мережі Петрі немає інваріанта переходів, то в ній не існує стаціонарно повторювана послідовність.

Доведення. Припустимо, що в мережі Петрі існує стаціонарно повторювана послідовність. Тоді з фундаментального рівняння (16.2) випливає, що існує така досяжна розмітка $\overline{M}$, для якої $\overline{M} = \overline{M} + C\overline{\sigma}$, звідки $C\overline{\sigma} = 0$, де $\overline{\sigma}$ — ненульовий вектор, що складається з невід'ємних цілих чисел. Існування такого вектора суперечить умові теореми. Теорему доведено.

Наявність інваріанта переходів мережі Петрі може означати існування у мережі стаціонарно повторюваної послідовності. Справді, використаємо інваріант переходів $\overline{y}$ замість вектора лічби спрацьовувань у фундаментальному рівнянні (16.2). Тоді $\overline{M}_k = \overline{M}_0$, що означає наявність у мережі стаціонарно повторюваної послідовності. Однак, оскільки фундаментальне рівняння (16.2) не враховує дозволеність переходів, ця послідовність переходів може виявитись недозволеною.

Приклад. Для мережі Петрі N_1 (рис. 15.7) у § 2 встановлено, що матриця C подається формулою (15.1). Знайдемо інваріанти мережі і зробимо висновки.

Інваріант вузлів знаходимо за формулою (16.3):

$$\begin{cases} -x_1 + x_2 + x_5 = 0, \\ x_1 - x_2 - x_6 = 0, \\ -x_3 + x_4 - x_5 = 0, \\ x_3 - x_4 + x_6 = 0 \end{cases} \Rightarrow \begin{cases} x_1 = x_2 + x_5, \\ x_4 = x_3 + x_5, \\ x_6 = x_5. \end{cases}$$

Надамо вільним невідомим x_2 , x_3 і x_5 натуральних значень, наприклад, $x_2 = x_3 = x_5 = 1$. Тоді $x_1 = 2$, $x_4 = 2$, $x_6 = 1$ і існує інваріант вузлів $\bar{x} = (2 \ 1 \ 1 \ 2 \ 1 \ 1)$, який складається з натуральних чисел. Отже, згідно з теоремою 1 і наслідком 1 з теореми 2 мережа Петрі N_1 є консервативною й обмеженою, проте не є строго консервативною, бо ваговий вектор $\bar{x}$ не є одиничним.

Інваріант переходів знаходимо за формулою (16.4):

$$\begin{cases} -y_1 + y_2 = 0, \\ y_1 - y_2 = 0, \\ -y_3 + y_4 = 0, \\ y_3 - y_4 = 0, \\ y_1 - y_3 = 0, \\ -y_2 + y_4 = 0 \end{cases} \Rightarrow y_1 = y_2 = y_3 = y_4.$$

Інваріант переходів є, наприклад, таким: $\overline{y} = (1 \ 1 \ 1 \ 1)^T$. Тому мережа Петрі N_1 може мати стаціонарно повторювану послідовність переходів. З аналізу мережі Петрі N_1 у § 1 випливає, що вона справді має стаціонарно повторювану послідовність. ■

3. Некласичні мережі Петрі

Некласичні мережі Петрі дають більше можливостей для опису і моделювання систем. Існують різні неklasичні мережі Петрі, орієнтовані на моделювання систем різних типів: імовірнісних, детермінованих та ін.

Найчастіше у неklasичних мережах Петрі розширюються можливості переходів. По-перше, переходи можуть спрацьовувати паралельно й одночасно. По-друге, перехід може спрацьовувати не миттєво, а протягом певного часу. Якщо збуджується кілька переходів, які претендують на використання тих самих маркерів, перехід, який спрацьовуватиме, вибирається у системі довільним чином. Можуть також задаватись імовірності спрацьовування переходів, які записують біля відповідних вхідних дуг (рис. 16.2). Ці ймовірності є меншими від одиниці, тому вони відрізняються від кратностей дуг, що вказуються натуральними числами. У деяких випадках задають пріоритети для переходів.

Для того щоб мати можливість моделювати динамічні системи, тривалість спрацьовування переходів має бути пов'язаною з модельним часом. У неklasичних мережах Петрі переходи бувають кількох типів — *миттєвий*, *детермінований*, *експоненціальний*, *рівномірний* тощо. Миттєвий перехід не пов'язаний з часом і спрацьовує миттєво, тобто так, як спрацьовують переходи у класичних мережах Петрі. Детермінований перехід спрацьовує протягом вказаного для нього фіксованого часу. Час спрацьовування експоненціального переходу є випадковою величиною з експоненціальним законом розподілу із заданим середнім значенням. Для рівномірного переходу задають два числа — початок і кінець проміжку, рівномірно розподілені випадкові числа з якого будуть братись як час спрацьовування переходу. Деколи тип переходу вказують певним кольором.

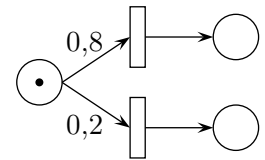


Рис. 16.2

Такі розширення роблять мережі Петрі неперервно-ймовірнісними моделями.

У неklasичних мережах Петрі можуть накладатись обмеження на максимальну кількість маркерів, які одночасно можуть перебувати у вузлі. Це може привести до того, що переходи, які містять достатню кількість маркерів у вхідних вузлах, не зможуть спрацювати, якщо хоч один з вихідних вузлів містить максимально допустиму кількість маркерів.

У деяких випадках у мережах Петрі доводиться вводити дуги заперечення та інформаційні дуги. *Дуга заперечення* (її ще називають *інгібіторною дугою*) має вигляд лінії з кружечком на кінці замість стрілки і може з'єднувати лише вхідний вузол і перехід. Перехід з дугою заперечення є дозволим лише тоді, коли у всіх вхідних вузлах, з'єднаних з ним звичайними дугами, є достатня кількість маркерів, а у вхідних вузлах, з'єднаних з ним дугами заперечення, немає маркерів. Приклад спрацювання такого переходу є на рис. 16.3. Дуга заперечення дозволяє моделювати в мережах Петрі логічний оператор заперечення.

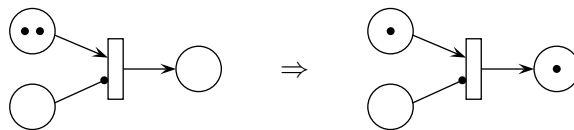


Рис. 16.3. Спрацювання переходу з дугою заперечення

Інформаційна дуга має вигляд пунктирної лінії зі стрілкою на кінці і може з'єднувати лише вхідний вузол і перехід. Перехід з інформаційною дугою є дозволим лише тоді, коли у всіх його вхідних вузлах є достатня кількість маркерів. При спрацюванні такого переходу маркери вилучаються лише з вхідних вузлів, з'єднаних з ним звичайними дугами. Маркери у вхідних вузлах, з'єднаних з переходом інформаційними дугами, лише дозволяють перехід, але не вилучаються з вузлів при його спрацюванні. Приклад спрацювання переходу з інформаційною дугою наведено на рис. 16.4.

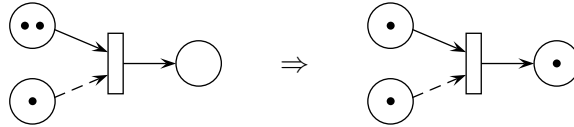


Рис. 16.4. Спрацювання переходу з інформаційною дугою

Приклад 1. В аеропорт для перевезення за тим самим маршрутом надходять вантажі у контейнерах, утворюючи стаціонарний пуассонівський потік з інтенсивністю 0,5 контейнера за хвилину. Вантажний аеропорт не має фіксованого розкладу, а літаки відправляються по мірі їхнього повного завантаження. У розпорядженні керуючого аеропортом для перевезення вантажів за даним маршрутом є три літаки з вантажопідйомністю 70 контейнерів і один літак з вантажопідйомністю 120 контейнерів. Час польоту кожного літака туди й назад, включаючи час розвантаження, розподілено рівномірно на проміжку від чотирьох до шести годин. Керуючий аеропортом намагається якомога частіше використовувати літаки меншої вантажопідйомності для зменшення часу, необхідного для транспортування вантажів. Літак, що піднімає 120 контейнерів, використовуються тільки тоді, коли інших літаків немає в наявності. Припускається, що часом завантаження літаків можна нехтувати. Скласти мережу Петрі для моделювання роботи аеропорту.

Розв'язання. Для моделювання стаціонарного пуассонівського потоку використовується експоненціальний закон розподілу. У випадку експоненціально розподіленої випадкової величини середнє значення є оберненою величиною до інтенсивності. За одиницю модельного часу вибираємо хвилину. Мережа Петрі N_5 , яка може використовуватись для моделювання цієї задачі, наведена на рис. 16.5.

У цій мережі перехід t_1 є експоненціальним з середнім значенням 2 і моделює надходження контейнерів, перехід t_2 є миттєвим і моделює завантаження малого літака, перехід t_3 є рівномірним на проміжку $[240, 360]$ і моделює політ малого літака, перехід t_4 є миттєвим і моделює завантаження великого літака, перехід t_5 є рівномірним на проміжку $[240, 360]$ і моделює політ великого літака. У процесі моделювання вузол p_2 міститиме чергу з контейнерів, p_3 — кількість вільних малих літаків, p_4 — кількість малих літаків, що перебувають у польоті, p_5 — кількість рейсів, виконаних малими літаками, p_6 — кількість вільних великих літаків, p_7 — кількість великих літаків, що перебувають у польоті, p_8 — кількість рейсів, виконаних великими літаками. ■

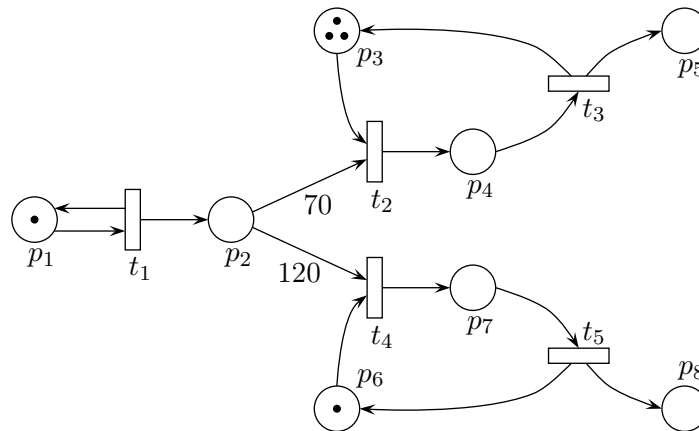


Рис. 16.5. Мережа N_5

Приклад 2. На дорозі з двостороннім рухом і однією смугою для руху в кожен бік є світлофор на пішохідному переході (рис. 16.6). Автомобілі у напрямку AB утворюють стаціонарний пуассонівський потік з інтенсивністю 5 машин за хвилину, а у напрямку BA — стаціонарний пуассонівський потік з інтенсивністю 6 машин за хвилину. На перетин пішохідного переходу автомобілі витрачають час, який є рівномірно розподіленою на проміжку $[2, 5]$ секунд випадковою величиною.

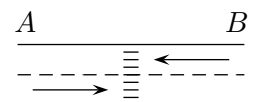


Рис. 16.6

Жовтий сигнал світлофора можна не враховувати, бо він визначає рух відповідно до попереднього сигналу і є лише попередженням про зміну руху. Тривалість зеленого сигналу для автомобілів — 1 хвилина, а червоного — 45 секунд. Скласти мережу Петрі для моделювання руху автомобілів на цьому переході.

Розв’язання. За одиницю модельного часу вибираємо секунду. Мережа Петрі N_6 , яка може використовуватись для моделювання цієї задачі, наведена на рис. 16.7. У ній перехід t_1 є експоненціальним з середнім значенням 12 і моделює появу машин у напрямі AB , перехід t_2 є рівномірним на проміжку $[2, 5]$ і моделює проїзд пішохідного переходу у напрямі AB , перехід t_3 є детермінованим зі значенням 60 і моделює тривалість зеленого сигналу світлофора для автомобілів, перехід t_4 є детермінованим зі значенням 45 і моделює тривалість червоного сигналу світлофора для автомобілів, перехід t_5 є експоненціальним з середнім значенням 10 і моделює появу машин у напрямі BA , перехід t_6 є рівномірним на проміжку $[2, 5]$ і моделює проїзд пішохідного переходу у напрямі BA . У процесі моделювання вузол p_2 міститиме чергу з автомобілів у напрямі AB , p_3 — кількість автомобілів, що проїхали через перехід у напрямі AB , p_7 — чергу з автомобілів у напрямі BA , p_8 — кількість автомобілів, що проїхали через перехід у напрямі BA . Наявність маркера у вузлі p_4 моделює зелений сигнал світлофора для автомобілів. Інформаційні дуги від вузла p_4 до переходів t_2 і t_6 дозволяють використовувати один маркер у вузлі p_4 для багатьох запусків цих переходів. ■

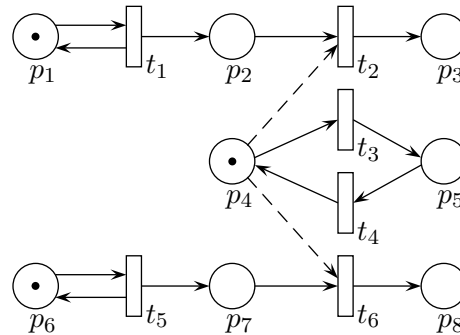


Рис. 16.7. Мережа N_6

Детальніше з класичними і некласичними мережами Петрі можна ознайомитись у посібниках [8, 13, 15].

Список рекомендованої літератури

1. Амелькин В. В. Дифференциальные уравнения в приложениях / В. В. Амелькин. — М. : Едиториал УРСС, 2003. — 208 с.
2. Бартіш М. Я. Дослідження операцій. Частина 5: Моделі з чинником часу / М. Я. Бартіш, І. М. Дудзяний. — Львів : ЛНУ імені Івана Франка, 2012. — 256 с.
3. Введение в математическое моделирование / Под ред. П. В. Трусова. — М. : Логос, 2005. — 440 с.
4. Жерновий Ю. В. Імітаційне моделювання систем масового обслуговування : Практикум / Ю. В. Жерновий. — Львів : Вид. центр ЛНУ ім. І. Франка, 2007. — 307 с.
5. Жерновий Ю. В. Марковські моделі масового обслуговування : Тексти лекцій / Ю. В. Жерновий. — Львів : Видавничий центр ЛНУ ім. Івана Франка, 2004. — 154 с.
6. Махней О. В. Лабораторний практикум з імітаційного моделювання у GPSS: методичні рекомендації до проведення лабораторних занять / Махней О. В. — Івано-Франківськ : ВДВ ЦІТ ПНУ, 2010. — 36 с.
7. Махней О. В. Математичне моделювання / О. В. Махней. — Івано-Франківськ : Супрун В. П., 2015. — 372 с.
8. Питерсон Дж. Теория сетей Петри и моделирование систем : пер. с англ. / Дж. Питерсон. — М. : Мир, 1984. — 264 с.
9. Пономарев К. К. Составление и решение дифференциальных уравнений / К. К. Пономарев. — Минск : Выш. школа, 1973. — 560 с.
10. Руководство пользователя по GPSS World / Пер. с англ. — Казань : Изд-во «Мастер Лайн», 2002. — 384 с.
11. Самарский А. А. Математическое моделирование: Идеи. Методы. Примеры / А. А. Самарский, А. П. Михайлов. — М. : Физматлит, 2001. — 320 с.
12. Смородинский С. С. Оптимизация решений на основе компьютерных имитационных методов и моделей / С. С. Смородинский, Н. В. Батин. — Ч. 2. — Мн. : БГУИР, 2005. — 104 с.
13. Стеценко І. В. Моделювання систем / І. В. Стеценко. — Черкаси : ЧДТУ, 2009. — 399 с.
14. Томашевский В. Имитационное моделирование в среде GPSS / В. Томашевский, Е. Жданова. — М. : Бестселлер, 2003. — 416 с.
15. Томашевський В. М. Моделювання систем / В. М. Томашевський. — К. : Видавнича група BVH, 2005. — 352 с.