

Лекція 8

МЕТОД ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ



МЕТОД ІМІТАЦІЙНОГО МОДЕЛЮВАННЯ

8.1.Імітаційне моделювання.

8.2.Алгоритми імітації процесів функціонування дискретних систем.

8.1.Імітаційне моделювання.

Серед великої кількості методів моделювання, що існують, виділимо такі методи:

- *аналітичне моделювання;*
- *математичне моделювання;*
- *імітаційне моделювання.*

Моделювання аналітичне, якщо представлення залежності *F* вихідних змінних *Y* від вхідних її змінних *X* має аналітичний вигляд, тобто представлений у вигляді відомих аналітичних функцій.

Нагадаємо, що функція називається аналітичною, якщо вона розкладається у ряд Тейлора. Аналітичні функції диференційовані безліч разів і тому до них можуть застосовуватись методи математичного аналізу. Перевагою цього методу моделювання є можливість отримання залежності $Y=f(X)$ в явному вигляді і застосування до неї методів класичного математичного аналізу. Якщо є можливість побудувати аналітичну модель системи, то завжди віддають перевагу цьому методу моделювання.

Етапи розробки моделей на прикладі побудови математичних моделей:

1. Постановка задачі моделювання.
2. Створення концептуальної моделі (тої, що описує).
3. Побудова математичної моделі.
4. Підготовка вхідної інформації.
5. Складання програми моделі.
6. Перевірка вірогідності моделі.
7. Керування моделі.
8. Розв’язання задачі дослідження моделі після корегування.
9. Аналіз результатів.

Деякі системи настільки складні, що не дивлячись на те, що опис їх функціонування піддається опису аналітичними функціями, знаходження залежності $Y=f(X)$ у явному вигляді виявляється неможливим. Наприклад, усі задачі математичного програмування мають досить простий аналітичний опис, але розв'язок задачі може бути знайдений тільки в результаті виконання певної кількості кроків. Іншими словами відомий алгоритм відшукання точного розв'язку задачі, але сам розв'язок не може бути записаний в аналітичній формі. Такий метод моделювання називають ***математичним моделюванням***.

Зауважимо, що алгоритм F відшукання точного розв'язку задачі може бути реалізований дослідником самостійно, за допомогою спеціального програмного забезпечення або за допомогою чисельних методів.

Існують системи, опис яких не піддається опису аналітичними функціями, але процес функціонування їх може бути описаний алгоритмом імітації.

Під ***імітацією*** розуміють відтворення за допомогою комп'ютерної програми процесу функціонування складної системи в часі.

У результаті багатократних прогонів імітаційної моделі дослідник отримує інформацію про властивості реальної системи. Такий метод моделювання називають ***імітаційним моделюванням***.

- Необхідність у використанні імітаційного моделювання може виникнути з наступних причин:

1) методи аналітичного моделювання не можуть бути використані через складність моделі;

2) застосування методів аналітичного моделювання дасть результати, які не задовольняють меті дослідження системі;

3) застосування методів імітаційного моделювання дозволяє універсалізувати рішення множини задач, що розглядається.

- ***Імітаційне моделювання*** системи передбачає, що процес функціонування системи **відтворюється** за допомогою алгоритму, який реалізується за допомогою комп'ютера.
- Англійський термін імітаційного моделювання „simulation modeling" підказує суть цього методу. Моделлю системи у разі імітації є комп'ютерна програма, яка прикидається (симулює), що вона є реальна система.
- Будь-яка реальна система функціонує в часі, змінювання якого відбувається незалежно від людини. Якщо дослідник збирається відтворювати функціонування системи, то йому доведеться, по-перше, моделювати змінювання часу, і тільки, по-друге, моделювати зміни в моделі відповідно до змінювання часу.

- Особливо корисною виявилась імітація для дослідження стохастичних систем, в яких переплітається вплив багатьох випадкових величин. Моделювання випадкових величин в комп’ютерних програмах пов’язано із побудовою генераторів випадкових величин. Тому від дослідника, який займається проблемами імітаційного моделювання систем, потрібно насамперед досконале знання методів та способів побудови генераторів випадкових величин.

Генератори випадкових величин

Відомі наступні способи генерування випадкових величин:

1) зберігання у комп'ютері таблиці випадкових чисел і отримання потім з неї даних для імітаційного моделювання;

2) використання деякого фізичного пристрою, наприклад ефект електронної лампи, для генерації випадкового шуму;

3) застосування рекурсивних формул коли на підставі і-того випадкового числа обчислюється i+1-ше випадкове число.

Недоліком першого способу є зберігання великого обсягу інформації та повільна швидкість. Недоліком другого способу - неможливість направленого експерименту з параметрами моделі. Третій спосіб не має недоліків попередніх способів і в теперішній час є найбільш прийнятним.

До генераторів випадкових чисел, що використовуються у цілях імітаційного моделювання, висуваються наступні вимоги:

1) числа рівномірно розподілені на інтервалі (0;1) і незалежні;

2) генерується достатньо велика кількість чисел, що не повторюються;

3) послідовність випадкових чисел може бути відтворена;

4) швидкодія;

5) обсяг пам’яті, що використовується, достатньо малий.

Якість генератора випадкових чисел перевіряють за допомогою машинного експерименту. Розрізняють тести двох типів – емпіричні та теоретичні.

Емпіричні тести основані на перевірці випадкових чисел, які сформував генератор, за допомогою статистичних тестів. Найбільш поширеними є перевірки *на рівномірність, на випадковість та на кореляцію*.

- Перевірку на **рівномірність** виконують методами ідентифікації закону розподілу з використанням критерію узгодження $\chi 2$.
- Весь інтервал (0;1) розбивається на m рівних частин і спостерігається кількість влучень в j-тий інтервал Nj, $j = 1,\dots m$ ($\Sigma Nj = N$). За критерієм $\chi 2$ оцінюється узгодження з теоретично очікуваною кількістю влучень в інтервал N/m. Рекомендована кількість інтервалів m=20÷50, кількість генерованих випадкових чисел N=(102÷105)m.
- Перевірку на **випадковість** виконують за *критерієм серій*. За цим критерієм перевіряється незалежність випадкових значень, але не їх рівномірність.
- При достатньо великій кількості спостережуваних значень (більше 4000) значення статистичного критерію R має бути меншим за значення критерію $\chi 2$ з 6 ступенями вільності і довірчою ймовірністю 0,9 ($\chi 2$=10,6).
- Перевірка **кореляції** полягає у визначенні кореляції між послідовностями випадкових значень, що розділені інтервалом довжиною j, тобто послідовностями $\zeta 1$, $\zeta 2$, $\zeta 3$... ζh і $\zeta 1+j$, $\zeta 2+j$, $\zeta 3+j$... $\zeta h+j$. Оцінка кореляції для кожного j=1,2...v виконується за спеціальними формулами.

- ***Теоретичні тести*** виконують перевірку генератора випадкових величин ґрунтуючись тільки на значеннях параметрів, які використовуються при генеруванні. У такому випадку дослідженню підлягають формули генерування випадкових значень.
- Рівномірність випадкових значень перевіряється у теоретичних тестах за допомогою зображення пар випадкових чисел (ζi, ζi+1) в одиничному квадраті координатної площини (або трійок чисел (ζi, ζi+1, ζi+2) в одиничному кубі). Якщо числа рівномірно заповнюють площу квадрата (простір кубу), то параметри генератора випадкових чисел задовільні.
- Наприкінці зауважимо, що не існує кінцевого набору тестів, що гарантує придатність кінцевої випадкової послідовності взагалі. Для даного набору тестів завжди існує послідовність, що задовольняє йому, але повністю непридатна для ряду окремих випадків. Однак проблем у зв’язку з цим не виникає, тому що дослідник не потребує додаткових властивостей випадковості.
- Тестуванню генераторів випадкових чисел потрібно приділити увагу рівно настільки, наскільки статична змістовність результатів імітаційного моделювання залежить від якості генератора випадкових чисел.

8.2. Алгоритми імітації процесів функціонування дискретних систем.

- Алгоритм, який відтворює функціонування системи, за допомогою комп’ютерної програми називається *алгоритмом імітації*.

Побудова алгоритму імітації складається з побудови:

1) алгоритму просування модельного часу;

2) алгоритму просування стану моделі в залежності від часу;

3) алгоритму збирання інформації про поведінку моделі у процесі імітації.

- Просування модельного часу та просування стану моделі в залежності від часу становлять єдину задачу побудови алгоритму імітації. Якщо дослідник для реалізації алгоритму імітації використовує імітаційну мову моделювання, то моделюючий алгоритм закладений в самий засіб реалізації, і дослідникові тільки корисно знати як він здійснюється, щоб не допустити помилки у застосуванні тієї або іншої імітаційної мови. Якщо ж дослідник використовує універсальну мову програмування, то задачу побудови моделюючого алгоритму йому потрібно вирішувати самостійно.

Узагальнена схема алгоритму імітації представлена на рисунку 8.1

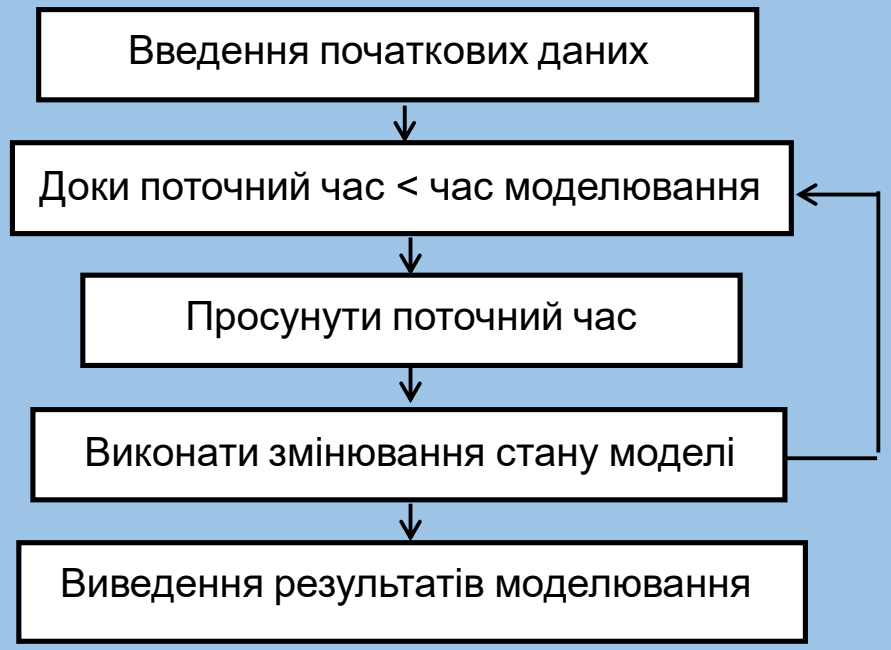


Рисунок 8.1 - Схема алгоритму імітації

Способи побудови алгоритмів імітації

Способи побудови алгоритму просування модельного часу

Існують три способи просування модельного часу:

- за принципом Δt,
- за принципом найближчої події,
- за принципом послідовного проведення об’єктів уздовж моделі.

Принцип Δt

- Весь інтервал часу, протягом якого моделюється система, поділяється на рівні інтервали довжиною $\Delta t = const$. При кожному просуванні модельного часу на величину Δt послідовно визначаються всі зміни, що відбуваються в моделі. Величина одного інтервалу Δt повинна бути настільки малою, щоб в одному інтервалі Δt відбувалось не більш однієї події. В противному випадку логіка алгоритму імітації порушується і функціонування моделі суттєво відрізняється від функціонування реальної системи.
- Принцип Δt є найбільш універсальним, але частіше він використовується для моделювання неперервних динамічних систем, оскільки вимагає у порівнянні з іншими способами більших затрат комп’ютерного часу при тій же точності моделювання.

Принцип найближчої події

- Дискретні системи, імітаційне моделювання яких розглядається, мають певну особливість: змінювання стану в таких системах відбувається тільки в деякі моменти часу, а в усі інші моменти часу система не змінюється. Змінювання стану моделі спричиняється виникненням певної події у системі. Наприклад, подія «надходження деталі до технологічного процесу» спричиняє збільшення кількості деталей у системі, подія «виникнення поломки обладнання» спричиняє відправку поламаного обладнання до ремонту і т.д. Процес функціонування системи розглядається як послідовність подій, що відбуваються у моделі.
- За принципом найближчої події модельний час просувається від моменту виникнення однієї події до моменту виникнення іншої, і після кожного просування часу реалізуються зміни стану моделі, відповідні до події, що виникла. Використання принципу найближчої події вимагає від дослідника побудови спеціальної процедури визначення моменту найближчої події, але при цьому він отримує виграш у затратах комп'ютерного часу, оскільки пропускається моделювання системи у моменти часу, коли події не відбуваються.
- Для дискретних систем просування модельного часу за принципом найближчої події є найефективнішим способом.

Принцип послідовного проведення об’єктів уздовж моделі

- За принципом послідовного проведення об’єктів уздовж моделі алгоритм просування часу не будується. Кожний об’єкт проводиться по моделі з моменту його надходження у модель до моменту виходу з моделі. Історія кожного проведення запам’ятовується, так що наступний об’єкт проводиться уздовж моделі з урахуванням історії попередніх проведень. Такий алгоритм імітації дозволяє моделювати систему тільки в моменти виникнення подій, не займаючись побудовою алгоритму просування модельного часу. Проте використовується він дуже рідко, оскільки часто призводить до складних заплутаних алгоритмів.

$$\varepsilon_\beta = \sigma_{\tilde{m}_x} t_\beta,$$

Способи побудови алгоритму просування стану моделі в залежності від часу

Процес функціонування імітаційної моделі може бути описаний з точки зору:

1) змінювання стану системи, що відбуваються в момент появи подій;

2) дій, які виконуються елементами системи;

3) процесу, який відбувається у системі.

У відповідності з означеними способами опису функціонування моделі існують три способи просування стану моделі в часі:

- орієнтований на події;
- орієнтований на дії;
- процесно-орієнтований.

Спосіб, орієнтований на події

- При підході, орієнтованому на події, дослідник визначає і описує події, які виникають у моделі. Імітація здійснюється виконанням упорядкованої у часі послідовності логічно взаємозв’язаних подій.
- Спосіб, орієнтований на події, при побудові алгоритмів імітації дискретних систем засобами універсальних мов програмування виявляється найефективнішим.

- *Спосіб, орієнтований на дії*

При іншому підході, основаному на описуванні дій, що виникають у системі, дослідник визначає і описує дії елементів системи та умови початку і кінця кожної дії. Умови початку і кінця кожної дії перевіряються після чергового просування модельного часу. Якщо умова початку дії виконується, то виконується дія. Для того, щоб була виконана кожна дія, сканування умов здійснюється для всієї множини дій при кожному просуванні модельного часу. Цей підхід найбільш ефективний, коли тривалість дії залежить від того, наскільки стан всієї системи задовольняє заданим умовам. В інших випадках більш ефективним виявляється підхід, орієнтований на події. Взагалі, оскільки сканувати умови початку і кінця дій потрібно після кожного просування модельного часу і для кожної дії, спосіб побудови моделюючого алгоритму, орієнтований на дії, має обмежене застосування в імітації дискретних систем.

- *Спосіб, орієнтований на процес*

При процесно-орієнтованому підході описують процес проходження об’єктів уздовж моделі, використовуючи кінцевий набір операторів. Послідовність операторів потім транслюється у відповідну послідовність подій і далі моделювання здійснюється як при підході, орієнтованому на події. Тому цей спосіб важко реалізувати самостійно, користуючись тільки універсальними мовами програмування. Використання його передбачає знання мов імітаційного моделювання, найбільш популярними з яких є *GPSS* та *SIMULA*. При цьому виходить компактна програма, що легко читається, зручна для коректування та налагодження.

Використання процесно-орієнтованих мов моделювання привертає дослідників простотою складання програми. Однак, оскільки набір операторів обмежений, можливості дослідника, який створює імітаційну модель засобами мови моделювання, сильно обмежені вибором пропонованих операторів.

Користування універсальними мовами програмування надає досліднику необмежені можливості при побудові імітаційного алгоритму, а студенту – можливість удосконалення навичок програмування. Гнучкість, яка досягається при користуванні універсальними мовами програмування, не може бути досягнута при користуванні спеціальним програмним забезпеченням. Тому, чим більше дослідник володіє навичками програмування, тим частіше при побудові алгоритму імітації він обирає універсальні мови програмування.